BabyLM 2026

The Second BabyLM Workshop: Accelerating Language Modeling Research with Cognitively Plausible Data

Proceedings of the Workshop

October 29, 2026

©2026 Association for Computational Linguistics

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
317 Sidney Baker St. S
Suite 400 - 134
Kerrville, TX 78028
USA
Tel: +1-855-225-1962
acl@aclweb.org

ISBN 979-8-89176-475-0

Introduction

We are excited to welcome attendees to the Second BabyLM Workshop! This follows two initial years of the BabyLM Challenge (co-hosted with CoNLL), and the first BabyLM Workshop in 2025. The Second BabyLM Workshop will be co-located with the 2026 Conference on Empirical Methods in Natural Language Processing on October 29, 2026 in Budapest, Hungary.

This year, the program includes an oral session for the winning BabyLM Challenge systems, one oral sessions for the award-winning workshop papers, and a poster session for all accepted submissions (including non-archival papers). There is also an introductory presentation from the organizers summarizing the challenge, this year's winning submissions, and trends across submissions. In addition, we have two invited talks.

We received 58 workshop submissions (many of which also included system submissions to the BabyLM Challenge) and 44 direct submissions to the BabyLM Challenge. Among these, 41 Workshop submissions and 36 Challenge submissions were accepted. We are grateful to the authors, whether in the challenge or workshop tracks, for advancing the science of language modeling. The participants' efforts are essential to advancing the state of cognitively plausible and sample-efficient language modeling.

We also extend our thanks to the organizers of EMNLP for their significant efforts in sustaining a conference of its scale, and in providing an environment for the BabyLM community.

Finally, we thank our program committee members—largely sampled from the participants of the challenge—for committing their time to help us curate an excellent program.

—The BabyLM Organizing Committee: Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox

Organizing Committee

Program Chairs

Leshem Choshen, Massachusetts Institute of Technology
Ryan Cotterell, ETH Zürich
Mustafa Omer Gul, Cornell University
Jaap Jumelet, University of Groningen
Tal Linzen, New York University
Aaron Mueller, Boston University
Suchir Salhan, University of Cambridge
Raj Sanjay Shah, Georgia Tech
Alex Warstadt, UC San Diego
Ethan Gotlieb Wilcox, Georgetown University

Program Committee

BabyLM Challenge Reviewers

Jeremias Bohn

Alexander Dazhen Cai, Leshem Choshen

Atul Dev

Lukas Edman

Dylan Gaines, Mustafa Omer Gul

Patrick Haller, Zhuojing Huang

Jaap Jumelet

Mary Kennedy

Mateusz Lango

Francois Meyer, Rafael Mosquera-Gómez, Aaron Mueller

Pierre Nugues

Juan Diego Rodriguez

Suchir Salhan, Sharath Sathish, Raj Sanjay Shah

Alex Warstadt, Ethan Gotlieb Wilcox

Qingyang Xu

BabyLM Workshop Reviewers

Shiva Ahir

Libby Barak, Lisa Beinborn, Adrian Brasoveanu, Bastian Bunzeck, Lisa Bylinina

Jon-Paul Cacioli, Andrew Caines

Justin DeBenedetto, Chenyi Ding, Jakub Dotlacil

Tamjid Hasan Fahim

Kata Gabor

Patrick Haller, Katherine Howitt, Imrul Huda

Mayank Jobanputra

Shunsuke Kando, Oscar Kemme, Mary Kennedy, Hyunho Kim, Jenny Kunz

Sharid Loáiciga

Jonas Mayer Martins, Francois Meyer, Keiji Miura, Rafael Mosquera-Gómez

Ercong Nie, Sergiu Nisioi

Francesca Padovani, Omkar Pangarkar

Vatsal Raina

Ekta Sood, Julius Steuer

Nikitas Theodoropoulos

Xiulin Yang

Noor Ul Zain, Iuliia Zaitova, Richard Zhu

Table of Contents

BabyLM Workshop

Halved CLM Exposure Mitigates Late-Training Degradation in Small Recurrent Language Models
Yash Maheshwari 1

Sentence-Level Foresight in Small Language Models: A Diagnostic Study of Learning Signals and Calibration
Ahmet Rıfat Öztürk, Yağız Ekrem Dalar, Nedim Mutlu Sezer, Feyzi Arda Salihoğlu and Ömer Faruk Aksoy 15

Hard-Wiring the Induction Head: The Surprising Effectiveness of Copy Mechanisms in Data-Efficient Language Modelling
Josh Taylor 24

HydraLLM: Adaptive Width Slicing for Elastic Text and Vision-Language Models
Oscar Kemme, Janek Haberer, Ali Hojjat and Olaf Landsiedel 33

Can Child Language Acquisition be Modeled by Linear Causal or Bidirectional Masked Models Trained on Child-Directed Input?: A Comparison of Baby Language Models and Children in Acquiring Wanna Contraction
Haerim Hwang 50

Does Typological Distance Affect Cross-lingual Transfer?
Pengfei Ren 65

What Survives the BabyLM Budget? A Controlled Study of Token Mixers and Generative Objectives
Filipe Laitenberger, Patrick Haller, Jonas Golde and Alan Akbik 75

Do Association-Reweighted BPE Merge Rules Help Data-Efficient Language Modeling?
Zy Li, Phu Han, Trinh Tran, Lam Duong and Andrew Fang 93

Retrieval Without Relevance: Prompt Dependence in Small Machine Translation Models
Heesu Kim, Shota Sato and Mamoru Komachi 105

When Does Depth Recursion Pay? A Parameter-Matched Study of Weight-Tied Hybrid Transformers on 100M Unique Words
Aditya Sasidhar 123

Parameter Golf: What Really Works?
Prashanna Mani Paudel and Shivanand Venkanna Sheshappanavar 134

Prosodic Structure as a Critical Component in Phonemic Language Modelling at BabyLM Scale
Bill K. S. Ku and Janet B. Pierrehumbert 148

Augustinian BabyLM: What Ostensive Definition Can and Cannot Teach a Small Language Model
Lisa Bylinina 165

How Far Is the Training Corpus from Real Infant Input? Auditing BabyLM Against Day-Long Home Recordings, and What an Age-of-Acquisition Curriculum Does Not Buy
Imrul Huda 179

Linguistically Inspired Multilingual Pre-Pretraining
Masato Mita, Max Müller-Eberstein, Misaki Yamaguchi and Yohei Oseki 187

BPE vs. Morphology-Aware Tokenization: A Case Study on Arabic Language Modeling at BabyLM Scale
Mohamed Badawy, Marion Di Marco and Alexander Fraser 199

Structural priors for data-efficient language learning
Yana Veitsman, Jonas Mayer Martins, Jonathan Lautenschlager and Lisa Beinborn 209

Arabic BabyLM: Effects of Diglossia and Tokenisation
Mozafar Haider and Burcu Can 227

Repetition Without Exclusivity: Scale Sensitivity of Referential Mechanisms in Child-Scale Language Models
Jon-Paul Cacioli 240

Reading Acquisition Order out of BabyLM Representations: Child-Scale Training Produces Developmentally Aligned Semantic Spaces
Chenyi Ding, Zachary Nicholas Houghton and Masoud Jasbi 250

Relational Attention for Data-Efficient Language Modeling
Adrian Brasoveanu, Ece Takmaz and Jakub Dotlacil 260

Can Translation Bridge the Data Gap for Low-Resource Languages? A Case Study of Hindi and Telugu
Venkata Naga Sai Vishnu Rohit Pulipaka and Libby Barak 279

Causal, Hybrid, and Ternary Transformers under a Restricted Data Budget
Arnav Srivastav 289

Can Randomly Permuting the LM Decoding Order Boost Data-Efficiency?
Max Ploner and Alan Akbik 299

When the Weakest Language Keeps Moving: ARMOR-DP for Multilingual BabyLM
Hanyu Liao, Yijing Chen and Tao Ji 309

Cross-Linguistic Effects in Bilingual Phoneme BabyLMs
Nikitas Theodoropoulos, Maria Lymperaiou and Giorgos Filandrianos 320

TinyLibrary: Do Age-Graded Curricula Help Small Vision-Language Models?
Dheeraj Varghese 333

Not All Irregularity Is Equal: Causally Isolating a Rare Failure Mode in Japanese Morphological Inflection
Wen Zhang 348

TRLM: Shared-Weight Recursive Reasoning for Non-Autoregressive Language Generation
Doğa Oytaç 358

Right Tool, Right Job: Native-Language Evaluation, Tokenizer Sensitivity, and Methodological Findings from a French-Only BabyLM
Adam Zachary Wasserman and David Beauchemin 369

What Helped with Huge Data Hurt with Small Data: Standardization, Representational Geometry, and Data Scale
Catherine Finegan-Dollak 388

Compact Multilingual Pre-training under the BabyLM 100M-Word Budget with Regex-Guided Tokenization and Attention Residuals
Yihan Liu, Tongjing Zhang and Pierre Nugues 403

Different Corpora, Different Skills? A Paired Study of Small Language Models
Edoardo Vaira, Georgios Chavales and Pierre Nugues 413

Modeling the Developmental Shift in Telicity Acquisition
Ellie Xia, Parisa Kordjamshidi and Alan Hezao Ke 426

Measuring Form and Function in Language Models
Hector Javier Vazquez Martinez and Charles Yang 438

Stories for the Win: Creating and Evaluating a Synthetic Dutch Children's Storytelling Dataset
Sabijn Perdijk, Gijs Wijnholds, Suzan Verberne and Max J. van Duijn 459

BabyLM Challenge

Findings of the Fourth BabyLM Challenge & Workshop: Going Beyond English
Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt and Ethan Gotlieb Wilcox 479

Exploring Auxiliary Objectives through Multi-Task Learning for Small-Data Language Model Pretraining: A BabyLM 2026 Submission
Alonso Palomino 493

MoEP+: Enhancing Modular Expert Paths for Data-Efficient Language Modeling
Noor Eidirhn Mustafa, Valdemar Melkersson and Pierre Nugues 504

Four Kilobytes of Memory: A 10M-Parameter Fixed-State Recurrent LM for BabyLM Strict-Small
Sam Furr 514

Weight-Tied Looped Depth for Data-Constrained Pretraining
Sebastian Miles 525

OLMo2 with Hybrid MLM-CLM Training for the BabyLM 2026 Multilingual Challenge
Zichao Wei 539

Loop Before You Leap: What Recurrent Depth Can and Cannot Buy in Data-Limited Pretraining
Daichi Hotta, Yohei Kobashi, Takeshi Kojima, Yusuke Iwasawa and Yutaka Matsuo 543

URCHIN: A Horizontal Spiking Language Model for Data-Constrained Pretraining
Po-Han Chiang 563

The Impact of Preference Optimization Data Selection: Teaching Practical Life Skills to BabyLM
Barbora Štěpánková, Vojtěch Lanz and Jakub Szymanik 575

Subword Segmental BabyLMs: Learning to Tokenise for Sample-Efficient Pretraining
Francois Meyer 585

Optimizing Multilingual BabyLM: The Roles of Tokenization, Data Packing, and Curriculum Learning
Fuming Yang, Kai-hao Yang, Hsuan Lien, Lukas Edman and Alexander Fraser 598

Pairing Original Text with Meaning-Preserving Rewrites for Sample-Efficient Small-Data Language Model Pretraining
Shaoxiang Wu, Alexander Fraser and Lukas Edman 611

Qwenfant: Train Big Character-level BabyLMs on Concise Parallel Text
Lukas Edman and Alexander Fraser 624

Does a Data2vec Auxiliary Help Small Masked Language Models? A BabyLM-2026 Strict-Small Study
Martin Benedikt, Ivan Dostál and Robin Klubarski 631

Recursive Transformers for BabyLM 2026
Serdar Gülbahar, Alexander Fraser and Lukas Edman 639

Spend the Wasted Epochs: Preference Post-Training from the Saturation Checkpoint for Sample-Efficient Language Modeling
Ujjwal Mishra and Sudipta Roy 646

Exploring Linguistic Parameters in LMs
Cristiano Chesi, Achille Fusco, Guido Formichi, Sarah Rossi, Hanna B. Cygan, Tommaso Sgrizzi and Asya Zanollo 659

Scaling Compute Instead of Parameters: Exploring Iterative Transformers for Data-Efficient Language Modeling
Wiktor Kamzela, Ondrej Dusek and Mateusz Lango 678

Learning from Correction: Data-Efficient Language Model Training Through Adaptive Teacher-Model Feedback
Sadri Islam, Tagore Rao Kosireddy, Kevin Li and Dylan Gaines 689

Miss Universal Grammar: Pretraining BabyLMs with Linguistic and Non-Linguistic Sources
Erika Varis Doggett and Mary Kennedy 702

From Puzzle-Solver to Language Model: Recurrent Inductive Bias, Data Scale, and Benchmark Validity in BabyLM
Patrick Haller, Filipe Laitenberger, Jonas Golde and Alan Akbik 722

Language Allocation and Targeted Data Design for a Multilingual BabyLM
Naoki Mori, Seiji Kyo and Koki Goda 735

Feeding BabyLMs Macaroni: Code-Switching Curricula Cause Cross-Lingual Convergence
Dries Rooryck, Alexander Dazhen Cai, Yonatan Belinkov, David Alvarez-Melis and Kianté Brantley 750

XpertGPT: Multi-Scale Sparse Expert Routing for Data-Constrained Language Modeling
Soham Jain, Harsh Singh, Divija Dewan and Atul Dev 767

BabyConceptLM: Exploring Dynamic Concept Computation under Restricted Language Exposure
Yizhe Xu, Ming Song, Danni He, Jianghao Liu, Yitong Wang and Qing Cai 779

Introducing Code-Switched Contexts to Cognitively-Inspired Bilingual Model Training
Zhuojing Huang, Luise Pohlmann and Lisa Beinborn 805

BabySteps at BabyLM 2026: Stronger Optimization and Causal Baselines for Data-Constrained Pretraining
Siddhi Satheesh, Jorge C Murillo and Yubei Chen 823

Learning What Can Combine and What Can Substitute: Relational Inductive Biases for BabyLM Pretraining
Yuwei Yang, Shihao Pu, Gefei Feng and Yiming Yang 831

Domain-Based Routing in Mixture-of-Experts Models for BabyLMs
Jeremias Bohn, Sebastian Feiertag, Nicolas Raube, Emil Schwenger and Georg Groh 843

Dense Supervision, Gentle Decay: Curriculum-Guided Rebalancing of ELECTRA-Style RTD and MLM for Small-Data Pretraining
Aftab Hossain Mikat and Tamjid Hasan Fahim 859

Tokens of Meaning: Semantically-Informed Morphological Tokenization in Low-Resource Language Modeling
Juan Diego Rodriguez 875

What Does Input-Dependent Depth Routing Buy in Data-Limited Language Modeling?
Yupei Wang 892

Language Exposure Matters: Distribution and Order in Multilingual BabyLM Pre-training
JiWoo Hwang, Lukas Edman and Alexander Fraser 904

Looped GPT-BERT: Trading Parameters for Computation in Small Language Modeling
Tingshuo Fan, Hongtao Mu, Tianyu Zhou, Hansen Liu and Tao Ji 912

Program

Thursday, October 29, 2026

09:15 - 09:30 *Opening Remarks*

09:30 - 10:30 *BabyLM Challenge Orals*

Qwenfant: Train Big Character-level BabyLMs on Concise Parallel Text
Lukas Edman and Alexander Fraser

BabySteps at BabyLM 2026: Stronger Optimization and Causal Baselines for Data-Constrained Pretraining
Siddhi Satheesh, Jorge C Murillo and Yubei Chen

Recursive Transformers for BabyLM 2026
Serdar Gülbahar, Alexander Fraser and Lukas Edman

Miss Universal Grammar: Pretraining BabyLMs with Linguistic and Non-Linguistic Sources
Erika Varis Doggett and Mary Kennedy

10:30 - 11:00 *Break*

11:00 - 12:00 *Invited Talk 1: Mitja Nikolaus*

12:00 - 12:30 *BabyLM Workshop Orals*

TinyLibrary: Do Age-Graded Curricula Help Small Vision-Language Models?
Dheeraj Varghese

Relational Attention for Data-Efficient Language Modeling
Adrian Brasoveanu, Ece Takmaz and Jakub Dotlacil

12:30 - 14:00 *Lunch*

14:00 - 15:30 *Poster Session*

Thursday, October 29, 2026 (continued)

15:30 - 16:00 *Break*

16:00 - 17:00 *Invited Talk 2: Pontus Stenetorp*

17:00 - 17:25 *Town Hall*

17:25 - 17:30 *Awards and Closing Remarks*

Halved CLM Exposure Mitigates Late-Training Degradation in Small Recurrent Language Models

Yash Maheshwari
Independent Researcher
yashmahe2018@gmail.com

Abstract

Small recurrent language models can degrade late in training under standard causal language modeling (CLM). On the BabyLM Strict-Small corpus, vanilla RWKV-7 peaks near 69% BLiMP early, then loses 2.57 to 3.85 pp by step 18k across all three seeds we ran. Halving how often CLM updates the weights mitigates this decline, via two mechanisms: alternating a backward MLM objective every other step (BiRWKV-7), or skipping every other gradient step (vanilla-skip). Vanilla-skip carries no auxiliary objectives yet beats BiRWKV-7 on peak BLiMP in every seed, so the auxiliaries are not necessary for stability. What halving changes is coupled: it reduces the CLM update count and doubles the learning-rate decay in update space. A constant-rate control still degrades at the two higher rates we tested, so a low terminal rate is not necessary for the decline; but at fixed update count, flooring the rate earlier recovers most of it. The update-space rate trajectory therefore also modulates the decline; it stays coupled to update count, and our design does not separate the two. On the official BabyLM 2026 pipeline our submission scores 68.70% BLiMP, above the GPT-2 baseline (65.23%, 98.4M) at about 28% as many parameters.

1 Introduction

The BabyLM challenge asks one central question in computational language acquisition: how much grammatical knowledge can a model extract from the quantity of data a human child encounters? We train on the official BabyLM 2026 Strict-Small corpus (Choshen et al., 2026; BabyLM Community, 2026). After applying our 32K BPE tokenizer, this gives 14.87M tokens drawn from children's books and other text sources (Warstadt et al., 2023; Choshen et al., 2024). In this tight budget, sample efficiency matters more than raw compute.

Transformer architectures dominate large-scale language modeling but impose $O(T^2)$ attention costs at training time and KV-cache memory proportional to sequence length at inference. State space models (Gu et al., 2022; Gu and Dao, 2023), retention networks (Sun et al., 2023), and RWKV-style architectures (Peng et al., 2023, 2025) share $O(T)$ training complexity while offering fixed-size recurrent state at inference. Both training and evaluation ran on consumer hardware (Apple M-series, MLX framework (Hannun et al., 2023); details in Appendix A.1).

RWKV-7 (Peng et al., 2025) combines data-dependent decay with in-context correction for expressivity close to full attention while keeping recurrent inference fast. We extend it to BiRWKV-7: a variant that trains a forward (L→R) causal pass and a backward (R→L) masked reconstruction pass over the same shared block weights. A learnable fusion gate scales how much backward-pass signal enters the combined representation. N-gram embedding features and multi-token prediction (MTP) heads add further training objectives. At inference, only the forward pass is used.

During BiRWKV-7 training, we observed a consistent failure: vanilla RWKV-7 with full CLM exposure degrades by 2.57 to 3.85 pp across all three seeds, while BiRWKV-7 stays stable. We therefore asked: is that stability due to BiRWKV-7's auxiliary objectives (MLM, MTP, n-grams), or just to the halved CLM update count that comes from alternating objectives? A vanilla-skip model (same RWKV-7 architecture, no auxiliary objectives, CLM updates halved by step-skipping) separates the two. It exceeds BiRWKV-7 on peak BLiMP across all three seeds (+0.83 pp seed 42, +2.58 pp seed 123, +1.81 pp seed 777) and drops only 0.33 to 0.83 pp from its peak.

Key finding. Full CLM training degrades BLiMP late in training for RWKV-7 across all

Proceedings of the Second BabyLM Workshop, pages 1–14
October 29, 2026 ©2026 Association for Computational Linguistics

three seeds, while halving CLM updates, by objective alternation (BiRWKV-7) or step-skipping (vanilla-skip), mitigates it. Since the auxiliary-free vanilla-skip still outperforms BiRWKV-7, the auxiliary objectives are not necessary for the stability. Halving also makes the learning rate decay twice as fast in update space, so update count and that trajectory stay coupled in our design (Limitations).

Contributions. (1) We identify and characterize a consistent late-training BLiMP degradation under full CLM exposure in small recurrent LMs, across three seeds (Section 4). (2) We show that halving updates by either objective alternation (BiRWKV-7) or step-skipping (vanilla-skip) consistently mitigates the degradation across all three seeds; halving is sufficient for the stability, but since an early learning-rate floor at full exposure also mitigates the decline (seed 42), it is not necessary. Two controls probe the schedule. A constant-learning-rate control reproduces the decline with no annealing, and a fixed-update anneal-timing probe shows the update-space learning-rate trajectory also affects the outcome. Reduced CLM update count and that trajectory are coupled in our design, so neither is isolated from the other and neither is established as the sole cause (Section 4). (3) We document that the best-validation-loss and peak-BLiMP checkpoints differ in all three BiRWKV-7 seeds; the BLiMP advantage is resolvable in seeds 42 and 123, while seed 777's 0.17 pp gap is within the measured $\sim$0.3 pp rerun tolerance (Section 4). (4) As a secondary result, n-gram features add +1.77 pp at inference. We also quantize recurrent KV states to int4; because the BLiMP scoring inputs fit in a single recurrence chunk, this reports infrastructure feasibility rather than compression quality (Appendix B.5).

2 Related Work

Prior BabyLM work has explored many strategies for extracting more grammatical knowledge from limited data. On the architecture side, ELC-BERT (Charpentier and Samuel, 2023) and LTG-BERT (Samuel et al., 2023) redesign the encoder, and modular expert paths add capacity along a different axis (Tapaninaho, 2025). On the objective side, GPT-BERT mixes masked and causal prediction (Charpentier and Samuel, 2024), adaptive masking decays the mask rate during training (Edman and Fraser, 2025), and curriculum methods reorder the data. These approaches generally ask which architecture or objective improves sample efficiency. Recurrent state space models (S4 (Gu et al., 2022), S5 (Smith et al., 2023), Mamba (Gu and Dao, 2023), and RWKV variants (Peng et al., 2023, 2025)) share $O(T)$ training complexity and fixed-size recurrent state at inference. BLaLM (Haller et al., 2025), which uses the mLSTM cell from xLSTM (Beck et al., 2024), is the closest prior linear-time BabyLM entry we are aware of; on the 2025 Strict-Small track it reports 67.0% BLiMP and 33.7% Entity Tracking. Because that result is from a different year's corpus and pipeline, we do not draw a head-to-head comparison and note it only to place linear-time models in the BabyLM setting. Our BLiMP evaluation protocol follows Warstadt et al. (2020).

Repeated-data overfitting under a fixed corpus is itself studied. Muennighoff et al. (2023) find that repeating data for up to about four epochs matches fresh data in the compute-optimal regime, with returns diminishing rapidly past that. Closer to our small-data scale, Chen et al. (2026) report that a 150M-parameter model on 75M tokens reaches its validation-loss minimum near epoch sixteen and then degrades monotonically, with training-time augmentation delaying the decline. Within BabyLM, Yamamoto and Miura (2025) counter over-training by repeating short learning-rate intervals with optimizer-state resets between them, and Roque and Velasco (2025) substitute simplified text rather than repeat it under a two-pass budget. Early optima and repetition-driven degradation are thus known. Our contribution is narrower: we document a broad late BLiMP degradation in small RWKV-7 and show that halving the CLM update count mitigates it without any auxiliary objectives, though update count stays entangled with the update-space learning-rate schedule (Section 4).

3 Model and Training

Terminology. Throughout, "CLM exposure" means the number of optimizer updates driven by the CLM objective, not total token or batch exposure. BiRWKV-7 matches vanilla in data steps and optimizer-update events, but not necessarily in parameter-space movement: equal update counts do not imply equal movement when the update gradients differ.

Architecture. BiRWKV-7 uses shared RWKV-7 blocks with two training passes.

Forward pass (L→R): produces causal CLM logits, plus lightweight multi-token prediction (MTP) heads for positions t+2 and t+3. The MTP heads are delta projections that add multi-step prediction signal without touching the primary recurrence.

Backward pass (R→L): the token sequence is masked (15%, BERT-style (Devlin et al., 2019)) and reversed, then fed through the same block weights to produce masked language modeling (MLM) reconstruction logits. The forward and backward hidden states are combined through a learnable sigmoid gate that controls how much backward signal enters the joint representation. This backward pass is an auxiliary training objective only: its gradients flow through shared weights during training, but it is not used at inference. "Bidirectional" here means weight-sharing across two causal passes (L→R and R→L), not full masked bidirectionality where both directions are simultaneously visible at each position. All BLiMP scores in this paper use the forward CLM head only. Entity Tracking results using the fused representation are reported as a secondary finding in Section 4.

N-gram embedding projection: a (2048 × n_embd) linear projection blends character-level 2/3/4-gram features (MD5 hash, 2048 vocab) into token embeddings through a learnable scalar alpha (initialized at 0.10). Alpha rises to ∼0.20 early in training and stabilizes near 0.13 across all three seeds.

Configuration. The shared RWKV-7 backbone has 27,369,984 parameters (∼27.4M), which is also the exact count of the vanilla and vanilla-skip models. BiRWKV-7's auxiliary components (n-gram projection, MTP heads, fusion gate) add 787,585 parameters for 28,157,569 in total (∼28.2M); only the forward CLM path runs at inference. The model uses 8 layers, hidden dim 384, 6 heads (head size 64), 32K BPE vocabulary, and context length 512. Training uses a chunked WKV-7 kernel (chunk_size=64, diagonal-only intra-chunk approximation), which affects both training gradient flow and inference; exact-WKV effects are not measured and are noted in the Limitations section.

Evaluation protocol. BLiMP accuracy is computed as the fraction of minimal pairs where the grammatical sentence has higher log-probability under the forward CLM head. Checkpoints are selected by validation loss (best-loss) and by BLiMP sweeps for peak-BLiMP, on a 2k-step grid (2k through 18k, nine checkpoints per seed) for vanilla and BiRWKV-7 and on a denser 500-step grid for the submitted vanilla-skip model (30 candidates for seed 42); the unequal grid inflates vanilla-skip's peak margin over BiRWKV-7 and is quantified in Limitations. The trajectory BLiMP scores in Figure 1 and Tables B.1, 1, B.2, B.3, B.5, B.6, and B.7 use this internal MLX evaluator; the official BabyLM pipeline (Table 2) reads about 1.25 pp lower at the same checkpoint (vanilla-skip step 18k: 69.95% internal vs 68.70% official). Every within-model degradation comparison in this paper is computed on the internal evaluator alone, so this internal-to-official offset does not enter those comparisons; it bears only on the absolute official submission numbers. Repeated evaluation of an identical checkpoint varies by up to ∼0.23 pp at a single point, from nondeterministic GPU reduction order in the MLX/Metal backend (measured by re-scoring a single near-peak seed-42 checkpoint three times: 68.08, 68.31, 68.21). We therefore treat ∼0.23 pp as the observed single-point rerun variation, and ∼0.3 pp as the corresponding tolerance on a quantity that differences two such measurements (a peak-to-final drop, or a checkpoint-to-checkpoint gap). This tolerance is measured at stable, near-peak checkpoints. The degraded step-18k vanilla checkpoints vary more: independent evaluation sweeps of the same checkpoint can differ by up to ∼1.5 pp (for instance seed 777 reads 67.55% and 66.10% in two separate sweeps), so the vanilla peak-to-final drop magnitudes in Table 1 are accurate only to about that range. The direction of the vanilla decline is corroborated independently by rising held-out validation loss (Appendix B.9), so the degradation itself is robust even where its exact per-seed magnitude is not tightly pinned.

Training schedule. Training alternates between two objectives at each step:

- **CLM + MTP** (even steps): forward pass only. Cross-entropy on next-token targets (weight 0.60) + MTP delta heads for t+2, t+3 (weight 0.15).
- **MLM** (odd steps): masked forward + backward passes over shared weights. Reconstruc-

tion from 15%-masked context (weight 0.25).

Combined loss: $L = 0.60 \times L_{\text{CLM}} + 0.25 \times L_{\text{MLM}} + 0.15 \times L_{\text{MTP}}$. This alternation yields 9k effective CLM gradient updates over 18k total steps, exactly half of vanilla's full exposure, with the remaining odd steps computing MLM reconstruction through both forward and backward passes.

Design principle. BiRWKV-7 differs from vanilla RWKV-7 in two ways: it adds auxiliary objectives (MLM, MTP, n-grams) and halves CLM exposure through objective alternation. These two factors are perfectly confounded inside BiRWKV-7. The vanilla-skip model separates them: it keeps the halved CLM update count while removing all auxiliary objectives, n-gram features, and the backward pass entirely. Its stability is therefore consistent with reduced CLM-objective updates rather than bidirectional auxiliary training, though the update-space learning-rate trajectory is coupled to that update count and is discussed in Section 4.

Baselines. *Vanilla RWKV-7*: same architecture and hyperparameters as BiRWKV-7, CLM-only (no MLM, MTP, n-grams, or backward pass), 18k CLM gradient updates over 18k steps. *Vanilla-skip*: same as vanilla but with CLM updates halved by step-skipping; odd steps advance the LR schedule with no gradient update, giving 9k effective CLM updates over 18k total steps.

Warm-start reference (non-competitive). A confounded warm-start run using Common Crawl initialization and an SGDR (stochastic gradient descent with warm restarts) schedule is documented in Appendix A. It does not qualify for the competitive track and is not used in any primary claims.

4 Results

4.1 Late-training degradation and its mitigation

Vanilla RWKV-7 (27,369,984 parameters, CLM-only over 18k steps) degrades late in training. It peaks between steps 4k and 6k, then loses 2.57 to 3.85 pp of BLiMP by step 18k across all three seeds (Figure 1; per-seed values in Table 1). Held-out validation loss rises alongside the decline (Figure 2), which argues against a BLiMP-specific scoring artifact. On seed 42, validation loss bottoms out near step 4k and then climbs about 1.3 nats to step 18k while training loss keeps falling, the classic overfitting signature.

Halving how often the CLM objective updates the weights mitigates the decline. BiRWKV-7 alternates CLM on even steps and MLM on odd steps, giving 9k effective CLM updates over 18k total, and stays stable through training. Its seed-42 trajectory holds near 69% from step 10k on, reaching 69.45% at step 16k (Appendix B.1, Table B.1), in the range of Baby Llama (58M parameters, 69.8% BLiMP; Timiryasov and Tastet, 2023, an earlier evaluation round, reported for context only, not as a controlled parameter-efficiency claim).

Auxiliary objectives and halved CLM exposure are confounded inside BiRWKV-7 (Section 3). The vanilla-skip model separates them. It shares the RWKV-7 backbone and the same per-scheduled-step LR schedule, carries no auxiliary objectives, and halves CLM exposure purely through step-skipping: odd steps advance the schedule but apply no gradient update, giving 9k effective CLM updates over 18k total. With no auxiliary objectives, it still exceeds BiRWKV-7 on peak BLiMP across all three seeds (+0.83, +2.58, +1.81 pp). On seed 42 it peaks at 70.28% (step 7,500) and drops only 0.33 pp by step 18k, about 1 pp above vanilla's peak, with the dominant contrast in the post-peak trajectory (Figure 1a). Table 1 reports all three model types across all three seeds.

Vanilla-skip has no n-gram features, which add +1.77 pp at inference for BiRWKV-7 (Appendix B.3), yet it outperforms BiRWKV-7 on all three seeds. Halving CLM-objective updates is therefore enough to produce stability with no auxiliary objectives.

Reduced data exposure is not necessary for that stability either. Step-skipping halves how much data vanilla-skip sees ($\sim$5 epochs versus $\sim$10 for vanilla; Table A.1), so less overfitting is a candidate explanation. BiRWKV-7 removes it with no extra run: it samples a fresh batch on every step, so it sees all 18k data steps ($\sim$10 epochs, as vanilla) and makes 18k optimizer updates (9k CLM plus 9k MLM), yet applies only 9k CLM-objective updates and stays stable. Data exposure alone therefore cannot explain the contrast. BiRWKV-7 matches vanilla in data steps and optimizer-update events, but the differing CLM and MLM gradients need not produce the same parameter-space movement. The two stable arms share one more thing

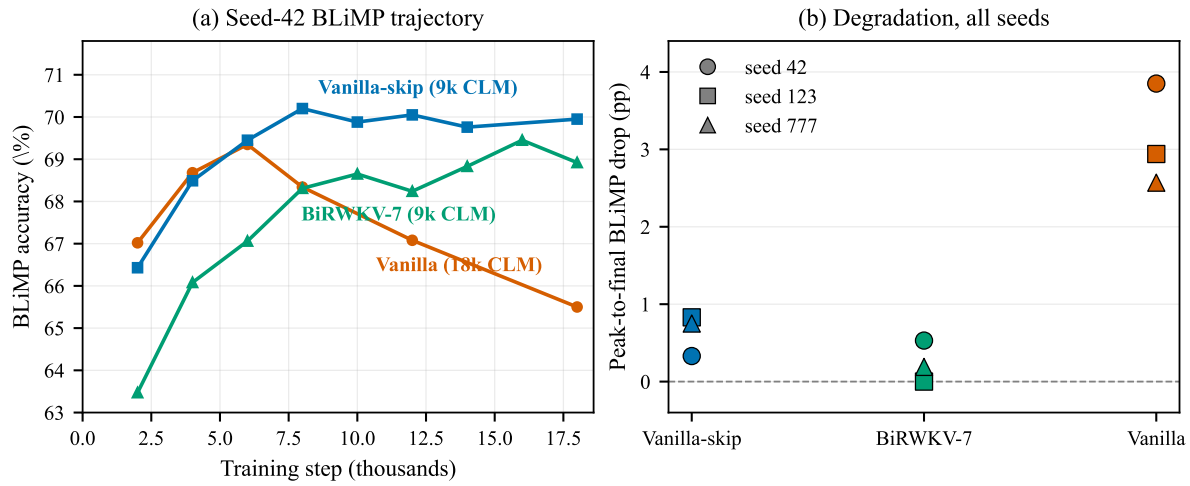


Figure 1: Halving CLM updates mitigates late-training BLiMP degradation. (a) Seed-42 BLiMP trajectory for vanilla RWKV-7 (18k CLM updates), vanilla-skip (9k, step-skipping), and BiRWKV-7 (9k, objective alternation). Vanilla peaks near step 6k and then declines; vanilla-skip peaks early and holds, while BiRWKV-7 rises and plateaus near 69% from step 10k on, broadly stable with only a small late decline. Each arm is drawn from a single evaluation sweep on the 2k-step grid; vanilla-skip's true peak (70.28% at step 7,500) is located off this grid on a denser 500-step sweep. (b) Peak-to-final BLiMP drop for all three conditions across seeds 42, 123, and 777 (plotted as positive magnitudes; larger is worse). The two halved-update conditions sit near zero (0.00 to 0.83 pp); full-exposure vanilla drops 2.57 to 3.85 pp. Exact per-seed values are in Table 1.

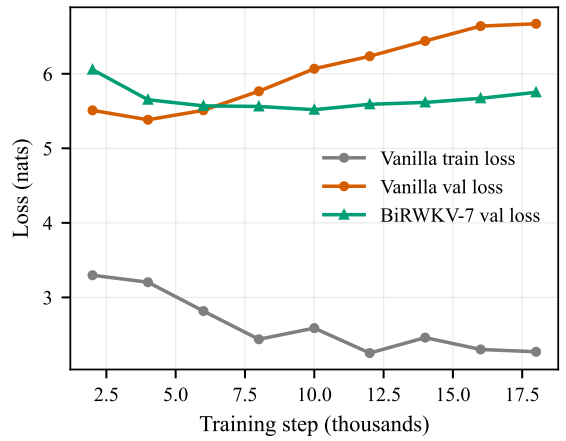


Figure 2: Seed-42 loss trajectory over the 2k-step grid. Vanilla RWKV-7's validation loss reaches its minimum near step 4k and then rises about 1.3 nats while training loss keeps falling. BiRWKV-7, with half the CLM-objective updates, instead reaches its validation-loss minimum near step 10k and rises only about 0.2 nats thereafter. The per-category BLiMP breakdown of the same degradation is in Appendix B.9.

beyond halved CLM updates. Both spread their CLM updates across a cosine that reaches its floor in 9k CLM updates rather than 18k, so both incur a faster decay in CLM-update space, which the next subsection probes.

4.2 The learning-rate schedule confound

Halving CLM updates changes two things at once, and our design does not separate them. Skipping every other step leaves the per-scheduled-step learning rate unchanged: the schedule is indexed by scheduled step, so vanilla and vanilla-skip use the same rate at each scheduled step, and their mean per-update rate is essentially identical ($\sim$2.15e-4 for both). What differs is the update-space trajectory. Because vanilla-skip traverses the cosine over 9k surviving updates rather than 18k, its rate decays about twice as fast per update, and at a matched update index it sees a lower rate than vanilla outside the brief warmup. Reduced CLM update count and the shape of the update-space schedule are thus entangled. Two probes bound what each contributes.

The first isolates the schedule's terminal rate. We trained vanilla RWKV-7 at a constant learning rate (warmup, then held flat) for the full 18k updates, with no cosine decay, at three rates in the band where the cosine model is still declining (1e-4, 2e-4, and 3e-4; seed 42). At the two higher rates it peaks early and then sheds 3.0 to 4.0 pp by step 18k; at 1e-4 it loses 1.2 pp. These rates sit above the cosine schedule's terminal value, so the control never reaches the cosine model's final low-rate regime. The narrow claim is that reaching a

Model Type	Seed	Peak BLiMP	Peak Step	Final (18k)	Degradation
Vanilla-skip (9k CLM, 0 aux)	42	70.28%	7,500	69.95%	−0.33 pp
Vanilla-skip (9k CLM, 0 aux)	123	70.55%	8,500	69.72%	−0.83 pp
Vanilla-skip (9k CLM, 0 aux)	777	69.96%	10,500	69.21%	−0.75 pp
BiRWKV-7 (9k CLM + 9k MLM)	42	69.45%	16,000	68.92%	−0.53 pp
BiRWKV-7 (9k CLM + 9k MLM)	123	67.97%	18,000	67.97%	0.00 pp
BiRWKV-7 (9k CLM + 9k MLM)	777	68.15%	14,000	67.96%	−0.19 pp
Vanilla (18k CLM, 0 aux)	42	69.35%	6,000	65.50%	**−3.85 pp**
Vanilla (18k CLM, 0 aux)	123	68.78%	4,000	65.84%	**−2.94 pp**
Vanilla (18k CLM, 0 aux)	777	68.67%	6,000	66.10%	**−2.57 pp**

Table 1: All three model types across all three seeds. Halved-exposure models (vanilla-skip and BiRWKV-7) maintain stability in every case; full-exposure vanilla degrades 2.57 to 3.85 pp consistently. Vanilla-skip exceeds BiRWKV-7 on peak BLiMP by +0.83 pp (seed 42), +2.58 pp (seed 123), and +1.81 pp (seed 777); these peak margins are measured on an unequal checkpoint grid (denser for vanilla-skip) and are upper bounds, and the stability contrast is conservative for the same reason (Limitations).

low terminal rate is not necessary for the decline: sustained constant-rate CLM training degrades on its own. Repeating the 3e-4 control on seeds 123 and 777 gives peak-to-18k drops of 3.99, 2.66, and 3.56 pp for seeds 42, 123, and 777 (constant-LR figures, separate from the cosine degradation of Table 1), so the constant-rate degradation holds across all three seeds.

The second probe holds update count fixed and moves only the schedule. At full 18k-update exposure (seed 42) we vary when the cosine reaches its floor, over the first N updates (Figure 3). Flooring the rate earliest, over 4500 updates, ends at 68.43% BLiMP against 65.50% for the full vanilla schedule, a 2.9 pp gap at fixed update count. That gap is about twice the up-to-$\sim$1.5 pp same-checkpoint evaluation spread at degraded step-18k checkpoints (Section 3), so the extreme contrast is robust; the two intermediate horizons (N=9000, 13500) end only 0.51 pp apart, inside that spread, so we do not read their ordering as resolved (Appendix B.7). The update-space learning-rate trajectory therefore also modulates the decline at fixed update count.

Together the two probes place the effect within late-phase training dynamics. The two halved-CLM arms differ from vanilla in different subsets of architecture, auxiliary objectives, data exposure, and total optimizer updates, and no single one of those is shared by both, yet both stay stable. What both share with each other, and not with vanilla, is the halved CLM update count together with the faster CLM-update-space learning-rate decay that comes with it. The anneal probe shows the decline also moves at fixed update count, so update count is not the sole lever; it stays coupled to the update-space schedule, and we do not decompose their individual shares. Vanilla-skip carries no MLM and still holds within 0.83 pp of its peak in every seed, so MLM gradient diversity is not necessary to prevent the decline. All arms share the RWKV-7 architecture, so these results speak to CLM exposure within one recurrent family, not to other architectures or transformers.

4.3 Official evaluation and secondary findings

On the official BabyLM 2026 pipeline, the GPT-2 baseline (GPT-2-base retrained on the Strict-Small data) scores 65.23% BLiMP on the shared leaderboard (BabyLM Organizers, 2026). Our primary submission (vanilla-skip seed 42, step 18k) scores 68.70% BLiMP, 3.47 pp above the baseline, at the submitted model's 27,369,984 parameters against the baseline's 98.4M, about 28% as many parameters. The comparison is uncontrolled and only situates our result in the BabyLM 2026 framework. The advantage is specific to BLiMP: on the rest of the suite the model is weak (Text Average 32.8 in Table 2), so we frame the central claim as one about grammatical acceptability under CLM training, not broad task competence.

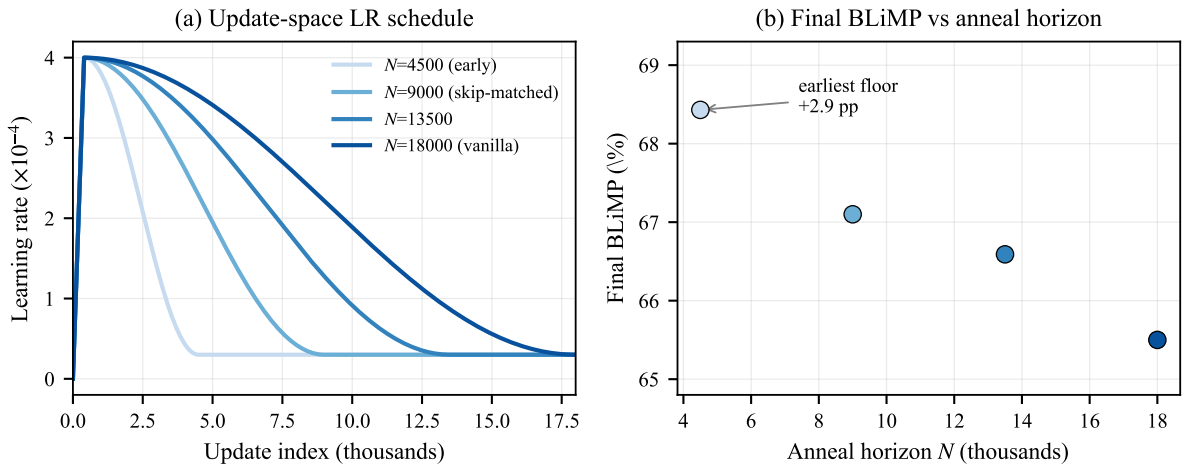


Figure 3: Anneal-timing probe (seed 42, full 18k-update exposure throughout). (a) Update-space learning-rate schedules: the cosine reaches its floor over the first N updates and is then held flat, for N=4500, 9000, 13500, and 18000; the N=18000 curve is the full vanilla schedule and the N=9000 curve reaches the floor at the same rate as vanilla-skip. All start at peak 4e-4 and end at 3e-5 (Table A.1). (b) Final BLiMP against anneal horizon N. Flooring earlier lifts final BLiMP: the earliest floor (N=4500) ends 2.9 pp above the full vanilla schedule (N=18000). The two intermediate horizons (N=9000, 13500) differ by only 0.51 pp, within the ∼1.5 pp same-checkpoint evaluation spread at degraded step-18k checkpoints (Section 3), so their ordering is not resolved.

Task	Score
BLiMP (filtered)	**68.70%**
BLiMP Supplement	57.24%
COMPS	52.55%
Entity Tracking	19.98%
EWoK	1.53%
Reading (Eye-Track. 1.68, Self-Paced 0.05)	0.86
GLUE (avg, Appendix B.4)	59.03%
AoA	2.85
Text Average (8-col)	**32.8**
BoolQ (GLUE component)	66.06%

Table 2: Official BabyLM 2026 evaluation for vanilla-skip seed 42 at step 18k. Tasks are zero-shot except the GLUE fine-tuning average; the table reports the official task scores and the resulting Text Average. Evaluation and aggregation details are given in Appendix A.1.

Three secondary findings round out the evaluation, each detailed in the appendix. First, the best-validation-loss and peak-BLiMP checkpoints differ in all three BiRWKV-7 seeds; a resolvable BLiMP advantage appears in seeds 42 and 123, whereas seed 777's 0.17 pp gap is within the measured ∼0.3 pp rerun tolerance (Appendix B.2, Table B.2). Thus validation loss alone may miss peak grammatical performance in this setting. Second, disabling n-gram features at inference on seed 42 costs 1.77 pp of BLiMP (Appendix B.3, Table B.3), larger than either checkpoint-selection difference. Third, GLUE fine-tuning of the submitted model averages 59.03% across seven tasks (Appendix B.4, Table B.4), below typical transformer baselines at this parameter count, as expected for causal recurrent models.

Entity Tracking is the one task where internal and official scores warrant separate comment. For the submitted model they agree closely: vanilla-skip at step 18k scores 18.56% internally and **19.98%** officially, a ∼1.4 pp gap with the official score above the internal one (the reverse of the BLiMP offset for that checkpoint in Section 3, where the internal sweep read higher). Both are low in absolute terms and near chance for this task, so this is a task the model does not solve. The one large divergence is the fused BiRWKV-7 representation, which peaks at 40.20% internally but scores only **22.45%** officially; we treat the official value as authoritative and do not claim the internal peak (Appendix B.8).

5 Conclusion

In small recurrent language models on a Strict-Small budget, halving CLM exposure mitigates late-training BLiMP degradation, and auxiliary objectives are not necessary for it. Under full CLM exposure, vanilla RWKV-7 peaks early and then sheds several points of BLiMP by step 18k across seeds 42, 123, and 777. Halving that exposure mitigates the decline, whether by alternating a backward MLM objective (BiRWKV-7) or by skip-

ping every other gradient step (vanilla-skip); the auxiliary-free vanilla-skip stays within 0.83 pp of its peak in every seed, so the auxiliaries are not what buys the stability. Two effects stay coupled, though. A constant-rate control degrades with no annealing on all three seeds, so a decaying schedule is not necessary; yet both halving arms also decay twice as fast in CLM-update space, and an anneal-timing probe shows an earlier floor lifts final BLiMP from 65.50% to 68.43% (seed 42; Appendix B.7). Update count and the update-space schedule co-vary in every arm, so we bound rather than decompose their shares. A clean decoupling, for instance 9k CLM updates run over an 18k-update cosine so the rate ends mid-schedule rather than at the floor, is the key open experiment. Finally, the best-validation-loss and peak-BLiMP checkpoints differ in all three BiRWKV-7 seeds. The BLiMP advantage is resolvable in seeds 42 and 123, while seed 777's 0.17 pp gap is within the measured $\sim$0.3 pp rerun tolerance; validation loss alone may therefore miss peak grammatical performance in this setting.

Availability. Training and evaluation code, the submitted checkpoint, per-table reproduction commands, and result provenance are prepared for public release under an open-source license, available from the author and to be deposited in a public code and model repository. Each headline number reads from a single named checkpoint and evaluation command.

Limitations

Several constraints limit how far these findings generalize:

1. **Three seeds only.** Effect-size estimates are uncertain. Vanilla-skip is the better-separated claim: all three seeds show clear stability (degradation 0.33 to 0.83 pp, against vanilla's 2.57 to 3.85 pp) and beat BiRWKV-7 on peak BLiMP (+0.83, +2.58, +1.81 pp). For BiRWKV-7 itself the evidence is thinner. Only seed 42 (-0.53 pp) shows a peak-to-final drop above the $\sim$0.3 pp rerun tolerance, and even that is small; seed 123 (0.00 pp) and seed 777 (-0.19 pp) both sit within the tolerance, so they are better read as no measurable change than as exact stability. For seed 123 the peak is the terminal (18k) checkpoint, so its 0.00 pp drop is definitional and that trajectory may still be non-decreasing at 18k rather than shown to have plateaued. Replication across more seeds and larger model sizes would help.

2. **Multiple comparisons and evaluation asymmetry.** Searching 27 peak candidates (9 checkpoints $\times$ 3 seeds) may inflate the apparent checkpoint-divergence gap. The peak-BLiMP checkpoint is also selected from a denser sweep for vanilla-skip (a 500-step grid, 30 candidates for seed 42) than for vanilla and BiRWKV-7 (a 2k-step grid, 6 to 9 candidates); a denser grid can only raise the recorded peak, which inflates vanilla-skip's peak-BLiMP margin over BiRWKV-7 while making its peak-to-final stability contrast conservative, since a higher recorded peak can only widen the measured drop. The three arms' trajectories are also read from separate evaluation sweeps that can differ by more than a point on an identical checkpoint, so cross-arm comparisons of absolute peak BLiMP (for instance vanilla-skip's roughly 1 pp margin over vanilla's peak) may carry a between-sweep offset; every peak-to-final degradation, by contrast, is computed within a single sweep and does not. At the degraded step-18k checkpoints, independent evaluation sweeps of the same checkpoint can differ by up to $\sim$1.5 pp (Section 3), so the per-seed vanilla drop magnitudes carry that uncertainty, though the direction of the decline is corroborated by rising validation loss and is robust to it. Separately, validation loss is estimated from $\sim$50 batches ($\sim$409K tokens) while BLiMP evaluates a 200-pair-per-paradigm subsample (about 13K of the roughly 65K pairs); this noisy-scalar versus larger-sample asymmetry does not, however, explain the consistent degradation pattern.

3. **Chunked WKV approximation.** The diagonal-only intra-chunk kernel affects both training gradient flow and inference. Exact-WKV effects are unmeasured for the from-scratch models in this paper.

4. **Mechanism of halving.** Under step-skipping, odd steps advance the LR schedule with no gradient update; under MLM alternation, odd steps update shared weights via MLM gradients. Both give $\sim$9k CLM up-

dates over 18k steps, but whether this mechanistic difference matters for long-run dynamics is unexplored. Reduced CLM update count is also not fully separated from the update-space learning-rate trajectory, since halving both shortens the cosine in CLM-update space; BiRWKV-7's total optimizer-update schedule matches vanilla's density, and only in CLM-update space does it share vanilla-skip's faster decay. The two probes in Section 4 bound but do not decompose the two effects.

5. **One architecture.** All arms are RWKV-7 at 27 to 28M parameters (27,369,984 for the vanilla and vanilla-skip backbone; 28,157,569 for BiRWKV-7 with its auxiliary components), so the degradation and its mitigation are established within a single recurrent family at one small scale. Whether they transfer to other linear-time architectures (Mamba, S4, xLSTM) or to transformers is untested, so the title's "recurrent language models" should be read as scoped to this family pending cross-architecture replication.

None of these constraints change the core observation: halving CLM exposure mitigates the degradation in every seed we ran, under either alternation or step-skipping. In practice, halving CLM update frequency reduced the late degradation we observed under cosine scheduling on this data-limited corpus. In this setting, validation loss alone may miss peak grammatical performance.

Acknowledgments

Generative AI assistants (large language models) were used throughout this work: for writing and debugging training and evaluation code, scaffolding experiment and evaluation harnesses, exploratory data analysis, and drafting and revising the manuscript text. The author designed the study, ran and supervised all experiments, and checked every reported number against its primary run artifact. The author is solely responsible for the scientific claims and for the final content of this paper.

References

BabyLM Community. 2026. BabyLM-community/BabyLM-2026-Strict-Small: Official training corpus for the 2026 BabyLM challenge (strict-small track). https://huggingface.co/datasets/BabyLM-community/BabyLM-2026-Strict-Small. Accessed: June 2026.

BabyLM Organizers. 2026. babylm-org/babylm-eval: Evaluation pipeline for the BabyLM 2026 challenge. https://github.com/babylm-org/babylm-eval. Accessed: June 2026.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2024. xLSTM: Extended long short-term memory. *arXiv preprint arXiv:2405.04517*.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every layer counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Michael K. Chen, Xikun Zhang, Fan Bai, Zhengding Hu, and Zhen Wang. 2026. Demystifying training-time augmentation for data-constrained language model pretraining. ArXiv:2606.16246.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Leshem Choshen, Ryan Cotterell, Michael Y. Hu, Tal Linzen, Aaron Mueller, Candace Ross, Alex Warstadt, Ethan Wilcox, Adina Williams, and Chengxu Zhuang. 2024. [call for papers] the 2nd BabyLM challenge: Sample-efficient pretraining on a developmentally plausible corpus. *arXiv preprint arXiv:2404.06214*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*, pages 175–191, Suzhou, China. Association for Computational Linguistics.

Awni Hannun, Jagrit Digani, Angelos Katharopoulos, and Ronan Collobert. 2023. MLX: Efficient and flexible machine learning on Apple silicon. https://github.com/ml-explore/mlx.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, volume 36, pages 50358–50376. Curran Associates, Inc. ArXiv:2305.16264.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, and 13 others. 2023. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, Singapore. Association for Computational Linguistics.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, Jiaju Lin, Jiaxing Liu, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, Nathan Wilce, Johan S. Wind, Tianyi Wu, Daniel Wuttke, and Christian Zhou-Zheng. 2025. RWKV-7 "Goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*.

Matthew Theodore Roque and Dan John Velasco. 2025. Beyond repetition: Text simplification and curriculum learning for data-constrained pretraining. In *Proceedings of the First BabyLM Workshop*, pages 246–255.

David Samuel, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2023. Trained on 100 million words and still in shape: BERT meets British National Corpus. In *Findings of the Association for Computational Linguistics: EACL 2023*.

Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. 2023. Simplified state space layers for sequence modeling. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. 2023. Retentive network: A successor to Transformer for large language models. *arXiv preprint arXiv:2307.08621*.

Joonas Tapaninaho. 2025. MoEP: Modular expert paths for sample-efficient language modeling. In *Proceedings of the First BabyLM Workshop*.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Rui Yamamoto and Keiji Miura. 2025. FORGETTER with forgetful hyperparameters and recurring sleeps can continue to learn beyond normal overtfitting limits. In *Proceedings of the First BabyLM Workshop*, pages 91–99.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. 2025. TurboQuant: Online vector quantization with near-optimal distortion rate. *Preprint*, arXiv:2504.19874.

A Training Configuration

A.1 Full Reproducibility Details

Data order. Training draws windows i.i.d. rather than shuffling epochs. At each step the loader samples `batch_size` start positions uniformly at random, with replacement, over the token memmap and takes the contiguous `ctx_len+1` window at each (`mlx_train.py`, `sample_batch`, default `schedule="random"`), so a single NumPy generator seeded by the run seed fixes the entire data order and it reproduces from the seed alone. Reported epoch counts are therefore total tokens seen divided by corpus size, not structured passes over a shuffled dataset; at matched steps all arms share the same expected per-token exposure, and MLM masking (BiRWKV-7) is resampled per step from the same generator.

Hyperparameter / Detail	Value
Parameters	27,369,984 backbone (vanilla, vanilla-skip); 28,157,569 for BiRWKV-7 with auxiliary components (`model.num_parameters()`, tied input/output embeddings)
Architecture	8 layers, hidden dim 384, head size 64 (6 WKV heads), attention-free
Optimizer	AdamW (beta1=0.9, beta2=0.999)
Weight decay	0.01
Gradient clipping	1.0 (max norm)
Learning rate schedule	Cosine; peak 4e-4, final 3e-5, 400-step linear warmup
Total steps	18,000 scheduled (vanilla-skip: 9,000 data steps $\approx$ 5 epochs over 14.87M tokens, since odd steps skip with no forward pass; full-CLM vanilla and BiRWKV-7: all 18,000 are data steps $\approx$ 10 epochs)
Batch size	16 sequences
Context length	512 tokens (sliding window on token memmap)
MLM mask probability	15% (BERT-style: independent replacement)
N-gram alpha init	0.10 (learned; stabilizes at $\sim$0.13)
Random seed (primary)	42 (seeds 123, 777 for replication)
Tokenizer	BPE, 32K vocab (trained on BabyLM Strict-Small)
Dataset format	uint16 token memmap (Strict-Small train split for training; separate held-out `val.bin` for validation)
Software versions	Python 3.11 (patch version not recorded); MLX 0.31.2; `mlx-metal` 0.31.2; `mlx-lm` 0.31.3
Training-code revision	`65685aaf6ba719a990cb5235e0486ab57a8a660a` (last committed revision of `scripts/mlx_train.py`)
Hardware and OS	Apple M5 Max, 128 GB unified memory; macOS 26.5.2 (build 25F84)

Table A.1: Full reproducibility details for the from-scratch models.

Validation construction. The training-time held-out estimate uses a separate uint16 token memmap `data/babylm/val/val.bin` (35,085,463 tokens), a distinct file from the Strict-Small training memmap used for optimization. It is built from two held-out shards of the BabyLM Strict text (about 21.9M whitespace words), tokenized with the same 32K BPE tokenizer as the training data; it serves only as a fixed held-out corpus for tracking generalization loss and never contributes a gradient update. At each validation event, the loader samples 50 batches of 16 contiguous 512-token windows with replacement; each window supplies the input sequence and its one-token-shifted next-token targets. Start positions are drawn by the same NumPy generator used by the run, so validation batches are resampled at each event rather than kept as a fixed panel. No optimizer update or MLM masking is applied to these batches.

Official evaluation reproducibility. Zero-shot scores use `babylm-org/babylm-eval` at commit `3bf5142`; the GLUE fine-tuning average uses `babylm/evaluation-pipeline-2025` at commit `bf55c11`. The seven per-task GLUE scores use the official leaderboard primary metric (F1 for MRPC and QQP, accuracy for the other five), and the reported average is their unweighted mean, recomputed from the per-task pipeline artifacts, so it can differ slightly from the server-side aggregate. The GPT-2 baseline BLiMP score (65.23%) is the published leaderboard value, read from the `babylm-eval` results table for the `Baseline-GPT2-Strict-Small` row; we did not re-run that baseline at commit `3bf5142`. COMPS probes conceptual properties and EWoK probes world knowledge; the EWoK column reports the value returned by the official 2026 scorer without modification. GlobalPIQA was not part of the Strict and Strict-Small zero-shot suite at the frozen submission-time commit (`3bf5142`); it was added to the pipeline in a later revision, so it does not appear in this table. The Text Average is recomputed from the eight sourced per-task cells, so it can differ slightly from the server-side aggregate; WUG morphology tasks from earlier rounds are not scored in the 2026 pipeline.

Kernel approximation. Chunked WKV-7 (chunk_size=64, diagonal-only intra-chunk) processes each length-512 sequence in chunks of 64 rather than step by step, which sharply reduces Metal kernel-dispatch overhead at T=512. Full recurrence correction is omitted within each chunk. This efficiency makes training recurrent models on consumer hardware feasible without a GPU cluster; exact-WKV effects are unmeasured for from-scratch models.

Code and checkpoints. Seed checkpoints (2k to 18k in 2k-step increments) are `.safetensors` weights with `metadata.json`. Reproducing any main finding requires only the seed number, one

evaluation command, and the checkpoint file.

A.2 Warm-Start Run (Non-Competitive)

A model pre-trained on ∼49M CC tokens, fine-tuned under SGDR (T0=3k), 14,250 total steps, reaching 70.8% BLiMP. Four factors differ from competitive runs (initialization, LR schedule, data volume, total steps), so no single factor can be credited for any BLiMP gap.

B Supplementary Material

All BLiMP scores in the main text use the forward CLM head only. This appendix collects the tables and diagnostics supporting the main results: the BiRWKV-7 seed-42 trajectory (Appendix B.1), the checkpoint-selection divergence (Appendix B.2), the n-gram inference ablation (Appendix B.3), GLUE fine-tuning detail (Appendix B.4), TurboQuant recurrent-state quantization (Appendix B.5), the vanilla-skip trajectories for seeds 123 and 777 (Appendix B.6), the anneal-timing probe (Appendix B.7), the entity-tracking reconciliation (Appendix B.8), and the overfitting diagnostics (Appendix B.9).

B.1 BiRWKV-7 Seed-42 Trajectory

Table B.1 reports the BiRWKV-7 seed-42 trajectory at key checkpoints, showing both the best-loss/peak-BLiMP divergence (detailed in Appendix B.2) and the absence of late degradation.

B.2 Checkpoint-Selection Divergence

Across all three BiRWKV-7 seeds, the best-validation-loss checkpoint is never the same as the peak-BLiMP checkpoint (Table B.2). The gaps are 0.80 pp (seed 42), 0.69 pp (seed 123), and 0.17 pp (seed 777). The first two exceed the measured ∼0.3 pp rerun tolerance; the seed-777 gap is within it.

Seeds 42 and 123 show gaps of 0.80 and 0.69 pp, both above the ∼0.3 pp tolerance for a difference of two BLiMP points (Section 3). Seed 777's 0.17 pp gap sits within that tolerance, so the checkpoint-selection divergence rests on seeds 42 and 123. With only three seeds, parametric confidence intervals are not meaningful, so the pattern is reported directionally.

B.3 N-gram Contribution at Inference

To assess whether n-gram features contribute beyond their training effect, we disable them at inference on seed 42 ckpt16000 (Table B.3).

Checkpoint	Step	Val Loss	BLiMP (CLM)
ckpt0002000	2k	6.06	63.47%
ckpt0010000	10k	**5.52**	68.65%
ckpt0016000	16k	5.67	**69.45%**
ckpt0018000	18k	5.75	68.92%

Table B.1: BiRWKV-7 seed 42 trajectory (key checkpoints). The best-loss checkpoint (step 10k, val loss 5.52) and the peak-BLiMP checkpoint (step 16k, 69.45%) do not coincide; the 0.80 pp gap means loss-based early stopping would miss the peak-BLiMP checkpoint. Unlike vanilla RWKV-7 (Figure 1), the model holds stable rather than degrading.

	Best-loss		Peak-BLiMP		BLiMP gap
Seed	Step	Val	Step	Val	(pp)
42	10k	5.52	16k	5.67	**0.80**
123	6k	5.49	18k	5.66	0.69
777	8k	5.36	14k	5.52	0.17

Table B.2: Checkpoint-selection divergence across all three BiRWKV-7 seeds. The best-validation-loss and peak-BLiMP checkpoints differ in every seed. The BLiMP gaps in seeds 42 and 123 exceed the measured ∼0.3 pp rerun tolerance, whereas seed 777's 0.17 pp gap is within it. The peak-BLiMP checkpoint has a higher validation loss in each row. Values are the training-time held-out loss estimates from each seed's trajectory summary.

B.4 GLUE Fine-Tuning

GLUE fine-tuning results for the submitted model (vanilla-skip seed 42, step 18k, last-token hidden state with left-padding) are in Table B.4.

B.5 TurboQuant Recurrent-State Quantization

Recurrent KV states from seed 42 (both ckpt16000 and ckpt18000) are reported in Table B.5.

B.6 Vanilla-Skip BLiMP Trajectories (Seeds 123 and 777)

Table B.6 reports the vanilla-skip trajectories for seeds 123 and 777; both peak above 69.9% and hold within 0.83 pp to step 18k, matching the seed-42 pattern of Figure 1a.

B.7 Anneal-Timing Probe (Early Learning-Rate Floor)

The constant-rate control (Section 4) shows the late decline does not require the cosine schedule's terminal low rate. The complementary question is whether reaching the floor *earlier* mitigates the decline. We train vanilla RWKV-7 at full exposure

Config	BLiMP (CLM)
With n-grams (alpha learned)	∼69.44%
Without n-grams (alpha = 0)	∼67.67%
Contribution	**+1.77 pp**

Table B.3: N-gram inference-time ablation on seed 42 ckpt16000. The +1.77 pp contribution is larger than the gap between the best-loss and peak-BLiMP checkpoints (0.80 pp). A training ablation would be needed to establish how much comes from training vs. inference-time features.

Task	Metric	Score
BoolQ	Acc.	66.06%
MultiRC	Acc.	56.72%
RTE	Acc.	48.92%
WSC	Acc.	59.62%
MRPC	F1	81.60%
QQP	F1	57.46%
MNLI	Acc.	42.87%

Table B.4: GLUE fine-tuning results for vanilla-skip seed 42 (step 18k), Fine-tuned with LR=3e-5, batch size 32, 10 epochs (WSC: 30 epochs). Last-token hidden state with left-padding, appropriate for causal recurrent models. Each task reports its official BabyLM 2026 leaderboard primary metric: F1 for MRPC and QQP, accuracy for the other five (each results file stores accuracy, F1, and MCC, and the leaderboard selects one metric per task). The GLUE average (59.03%, Table 2) is the unweighted mean of these seven per-task metrics. These scores are below typical transformer baselines at this parameter count; GLUE classification is a known challenge for causal recurrent models.

(the same 18k CLM updates, no step-skipping) but anneal the cosine learning rate to its terminal value over the first N updates and then hold it flat, varying N over 4500, 9000, and 13500 (seed 42). This holds update count fixed and moves only the update-space learning-rate trajectory. Table B.7 reports the peak-to-final BLiMP drops for seed 42. Repeating the default early floor (N=9000) across all three seeds gives drops of 1.00, 1.79, and 0.71 pp (seeds 42, 123, 777; mean 1.17, sample standard deviation 0.56), each below vanilla's 2.57 to 3.85 pp but with a spread that leaves the tighter question, whether an early floor fully eliminates the decline, unresolved at three seeds.

B.8 Entity Tracking Reconciliation

The submitted model's official ET (19.98%) is deterministic, reproducing across three independent runs and across first-index versus random tiebreaks applied to the same captured log-probs.

Mode	s42 Peak	s42 Final
none (FP32)	69.44%	68.92%
fp16	69.44%	68.92%
bf16	69.44%	68.92%
uniform_int8	69.44%	68.92%
uniform_int4	69.44%	68.92%
turbo_int4	69.44%	68.92%

Table B.5: TurboQuant (Zandieh et al., 2025) recurrent-state quantization (seed 42 ckpt16000 and ckpt18000). BLiMP is identical to the reported precision across all six modes, from FP32 through fp16/bf16 to int8 and per-row int4 (the FP32 peak reads 69.44% here versus 69.45% in Table B.1, a 0.01 pp difference between separate scoring runs). This is expected rather than evidence of lossless compression: BLiMP scoring inputs fit in a single recurrence chunk, so they never read a state that was compressed and carried across a chunk boundary. The table therefore reports that quantization does not disturb the single-chunk forward pass, not that the recurrent-state compression is lossless. We report BLiMP quality only and make no compression-ratio claim; the realized ratio depends on bit packing and per-row scale overhead that our feasibility path does not measure consistently.

Step	Seed 123	Seed 777
2,500	66.98%	65.86%
8,500	**70.55%** (peak)	69.45%
10,500	69.90%	**69.96%** (peak)
18,000	69.72%	69.21%

Table B.6: Vanilla-skip seeds 123 and 777 trajectories (key checkpoints). Seed 123 peaks at 70.55% (step 8,500) and ends at 69.72% at step 18k (−0.83 pp). Seed 777 peaks at 69.96% (step 10,500) and ends at 69.21% at step 18k (−0.75 pp). Both show small drops (0.83 and 0.75 pp), well below vanilla's 2.57 to 3.85 pp under halved CLM exposure.

The internal MLX script (loading weights directly) and the official HuggingFace-converted pipeline therefore differ by only ∼1.4 pp for this model, with the official score above the internal one; this is the reverse of the BLiMP offset for the submitted checkpoint in Section 3, where the internal sweep read ∼1.25 pp higher than the official pipeline.

The exception is the fused BiRWKV-7 representation: 40.20% internally (step 14k) against 22.45% officially. Both official scores are low in absolute terms, in the near-chance range for this task. The source of the fused gap, likely numerical precision, batching, or HuggingFace conversion, is unresolved, and we treat the official scores as authoritative; the qualitative pattern of fused ET

Anneal N	Peak (@step)	Final (18k)	Drop
4,500 (early)	68.97% (12k)	68.43%	0.54
9,000 (skip-matched)	68.10% (6k)	67.10%	1.00
13,500 (late)	69.68% (6k)	66.59%	3.09
18,000 (vanilla)	69.35% (6k)	65.50%	3.85

Table B.7: Anneal-timing probe, seed 42, full 18k-update exposure throughout; only the learning-rate anneal horizon N varies. The 18,000 row is the vanilla cosine schedule (Table 1); the skip-matched row (N=9000) reaches the floor in update space at the same rate as vanilla-skip. The robust contrast is between the extremes: the earliest floor (N=4500) ends at 68.43% against the vanilla schedule's 65.50%, a 2.9 pp gap, roughly twice the up-to-$\sim$1.5 pp spread seen at degraded step-18k checkpoints (Section 3). The two intermediate horizons (N=9000, 13500) end between these extremes, but their finals differ by 0.51 pp from each other and the N=13500 final (66.59%) sits only 1.09 pp above the vanilla endpoint, inside that degraded-checkpoint spread, so we do not read the intermediate ordering as resolved. The N=9000 peak (68.10%) is also the lowest in the family, so the individual peak-to-final drops are not directly comparable across rows. This is a single-seed probe over widely separated conditions, not a powered dose-response; the three-seed replication at N=9000 (Section text above) has a sample standard deviation of 0.56 pp.

rising while CLM ET degrades may be real but cannot be confirmed officially.

The high early-checkpoint ET is an artifact of a barely-trained model on a near-chance task. Internally, vanilla-skip CLM peaks at 41.72% at step 500 and degrades to 18.56% by step 18k, and BiRWKV-7 CLM peaks at 36.77% then degrades to 20.43%; the step-500 peak re-confirms through the official pipeline (41.33%), so it is not a capability the converged model retains. Finally, BiRWKV-7 seed 42 shows a three-way checkpoint split, best-loss (10k), peak-BLiMP (16k), and peak-internal-fused-ET (14k), extending Table B.2 to a third axis. The fusion gate grows over training (mean sigmoid $\sim$0.19 at step 2k to $\sim$0.42 at step 18k), so the backward pass contributes more to the joint representation as training progresses.

B.9 Overfitting Diagnostics

The late-training degradation carries a held-out overfitting signature (Figure 2). On seed 42, over the 2k-step checkpoint grid shared with BiRWKV-7, vanilla RWKV-7's validation loss reaches its minimum early, near step 4k (5.38 nats), then

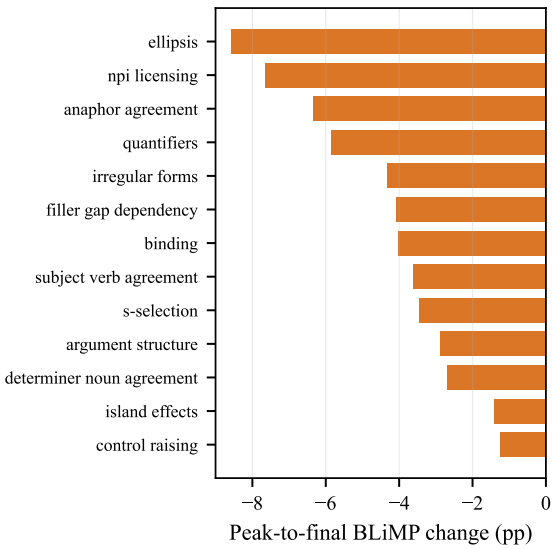


Figure B.1: Peak-to-final BLiMP change (step 6k to 18k) for all 13 high-level linguistic categories, vanilla RWKV-7 seed 42. Every category declines, from about 1 pp to about 9 pp, so the degradation is broad rather than concentrated in one fragile paradigm.

rises about 1.3 to step 18k (6.67) while its training loss keeps falling, tracking the BLiMP decline of Figure 1. BiRWKV-7, trained on the same $\sim$10 epochs of data but with half the CLM-objective updates, instead reaches its validation-loss minimum near step 10k and rises only about 0.2 to step 18k, with BLiMP stable. Both use the training-time held-out estimate, so the contrast is between comparable quantities.

The degradation is broad rather than concentrated (Figure B.1). Grouping the BLiMP paradigms into their high-level linguistic categories, all 13 categories decline from peak (step 6k) to step 18k for vanilla seed 42, ranging from about 1 pp to about 9 pp. (These are within-run diagnostic re-evaluations and carry the same $\sim$0.23 pp single-point rerun variation as the main tables; we read only the peak-to-final direction, which is negative in every category.) A single fragile paradigm would show one large drop against a flat remainder; instead the decline is spread across phenomena, consistent with overfitting that erodes grammatical acceptability broadly.

Sentence-Level Foresight in Small Language Models: A Diagnostic Study of Learning Signals and Calibration

Ahmet Rıfat Öztürk **Yağız Ekrem Dalar** **Nedim Mutlu Sezer**

Feyzi Arda Salihoğlu **Ömer Faruk Aksoy**

Abstract

Small language models provide a useful setting for separating *signal acquisition* from *behavioral usefulness*. We ask whether a DistilGPT-2-scale autoregressive language model can learn a sentence-level representation that favors a plausible next sentence over self-generated loop-like alternatives, and whether that representation independently improves generation. ForesightLM-v2 combines ordinary causal language modeling with a contrastive next-sentence embedding objective and a margin loss over self-loop hard negatives. We specify the full preprocessing, future-head architecture, hard-negative construction, scoring functions, thresholds, and decoding protocol. Across three WikiText-103 seeds and three WritingPrompts seeds, the held-out positive–negative cosine gap is consistently positive. However, generation reveals a calibration gap. On a fixed five-candidate pool, removing the learned future term changes only 2 of 120 selections, while generic, random, and shuffled score controls remain close. A newly matched training-free Base-c5 control—using the same temperature, top-p, maximum generation length, candidate count, and seed schedule—slightly outperforms future-calibrated c5 on the main automatic diagnostics. Local coefficient sensitivity also shows that conclusions near the nominal generic reranker are stable to the repetition and length weights. We therefore make a deliberately narrow claim: small autoregressive models can acquire a measurable sentence-level future signal, but the present evidence does not show that this learned signal is the main cause of generation improvements.

1 Introduction

Small language models are useful not only as compressed systems, but also as scientific instruments. Their failures are frequent, inexpensive to reproduce, and easier to audit across training, generation, and control conditions. This makes them well suited to a question that can be obscured in larger systems: when an auxiliary representation is measurable during training, does it actually cause better behavior at generation time?

We study this question through sentence-level future prediction. Standard autoregressive language modeling optimizes next-token likelihood, yet many weak-model failures are visible at a coarser scale: a continuation may remain locally fluent while repeating a semantic state, drifting from the prompt, or failing to progress. We define a *future signal* operationally as a context-conditioned vector whose cosine similarity is higher for an appropriate next-sentence representation than for a loop-like model-generated alternative. This definition is diagnostic rather than cognitive: we do not claim that semantic similarity itself is undesirable, nor that all repetition is incoherent. Repetition can be natural when a writer clarifies, emphasizes, or elaborates. Our hard negatives instead target high-repetition model outputs as a tractable proxy for weak-model degeneration.

ForesightLM-v2 augments a DistilGPT-2 checkpoint with a sentence-level projection head. At a context boundary, the last-token hidden state from the final transformer layer is projected into the 384-dimensional space of a frozen `all-MiniLM-L6-v2` sentence encoder. A batch-contrastive loss encourages the projected state to identify the actual following sentence, and an additional margin loss separates positive futures from self-loop hard negatives mined from model generations.

The central contribution is not a new decoding algorithm or a state-of-the-art coherence result. Instead, we separate five questions that are often conflated: (1) can the future signal be learned, (2) does it emerge during short training, (3) does raw auxiliary training change generation, (4) does a learned future score improve candidate selection beyond generic reranking, and (5) do the conclusions sur-

Proceedings of the Second BabyLM Workshop, pages 15–23
October 29, 2026 ©2026 Association for Computational Linguistics

vive random, shuffled, no-future, and training-free controls? This framing is consistent with the BabyLM workshop's broader interest in learning dynamics, data-efficient modeling, and analysis of small systems (Choshen et al., 2026). This is a non-competition workshop study rather than a BabyLM Challenge leaderboard submission.

Our main findings are mixed but informative. Training-side hard-negative gaps are positive across three WikiText-103 seeds and three WritingPrompts seeds. Generation-side results are much less specific: generic five-candidate reranking accounts for most of the semantic/progression improvement, the learned future term changes only 2/120 selections, and random or shuffled future-score controls remain close. A matched Base-c5 control confirms that this conclusion is not an artifact of mismatched sampling settings. We refer to this mismatch between representational learning and generation-time influence as a *calibration gap*.

Contributions.

- We give a fully specified sentence-level future-prediction objective for a small causal LM, including context boundary handling, projection architecture, contrastive loss, hard-negative margin loss, and all major training hyperparameters.
- We define and document the self-loop mining pipeline and explicitly separate gold-positive from pseudo-positive hard-negative pairs.
- We show positive held-out future/hard-negative gaps across three WikiText-103 seeds and three WritingPrompts seeds, plus a short training-step sensitivity analysis.
- We test causal attribution at generation time with no-future, future-only, random, shuffled, lexical, coefficient-sensitivity, and matched training-free Base-c5 controls.
- We report the negative result directly: the learned future score is measurable, but it is weakly coupled to five-candidate selection under the present interface.

2 Related Work

Small models as controlled learning systems. TinyStories constructs a synthetic corpus restricted to child-accessible vocabulary and shows that models with fewer than 10M parameters, including very shallow transformers, can generate multi-paragraph stories with strong fluency and consistency (Eldan and Li, 2023). Its main strategy is to simplify the data distribution so that coherent generation becomes learnable at small scale. Our goal is different: we keep a standard natural-text setting and ask whether an auxiliary sentence-level signal can be acquired by a weak causal LM, then test whether that learned representation actually affects generation. BabyLM similarly motivates small-scale language modeling as a setting for studying sample efficiency and learning behavior rather than only deployment efficiency (Choshen et al., 2026).

Text degeneration and negative supervision. Likelihood-trained neural generators can produce repetitive or otherwise degenerate continuations, and decoding strategy alone can substantially change those behaviors (Holtzman et al., 2020). Unlikelihood training addresses degeneration by explicitly reducing probability assigned to undesirable tokens or sequences and has been shown to reduce repetition while preserving language-modeling quality (Welleck et al., 2020). Our hard-negative objective is related in motivation but differs in level: it does not directly penalize token probabilities for a repeated phrase. Instead, it asks a sentence-level predicted future vector to separate an appropriate continuation from a loop-heavy model-generated sentence.

Future prediction in representation space. Contrastive Predictive Coding learns representations by predicting future latent samples using a contrastive objective with negatives (van den Oord et al., 2018). We adopt the same broad principle at sentence scale. Rather than predicting a future acoustic or visual latent, the model predicts the frozen embedding of the next sentence. Sentence-BERT provides an efficient embedding space in which semantically meaningful sentence vectors can be compared with cosine similarity (Reimers and Gurevych, 2019). This choice is computationally convenient, but it also limits the method: sentence-level semantic similarity can miss factual consistency, entity identity, discourse relations, and the distinction between legitimate restatement and degeneration.

Candidate selection as a confound. Decoding and reranking can improve generation without

changing model parameters. Contrastive generation methods likewise demonstrate that candidate scoring can strongly alter output properties (Su et al., 2022). This creates a causal attribution problem for our setting: if a five-candidate selector improves generation, the gain may come from candidate diversity and a generic semantic score rather than the learned future representation. Our evaluation therefore treats generic, random, shuffled, no-future, future-only, and training-free matched reranking as central controls rather than peripheral ablations.

3 Method

3.1 Sentence-pair construction

For WikiText-103 (Merity et al., 2017), we use the raw `wikitext-103-raw-v1` text. We normalize common WikiText tokenization artifacts (e.g., `@-@`, `@,@`, and `@.@`) and split non-heading text using punctuation followed by whitespace and an uppercase letter, digit, quote, or apostrophe. We keep sentences containing 5–80 lexical tokens and at least 20 characters. Each valid adjacent pair forms

$$c = s_i, \qquad s^+ = s_{i+1}. \tag{1}$$

We retain 30,000 training pairs and 2,000 validation pairs per domain. WritingPrompts (Fan et al., 2018) is processed analogously over story text. This simple rule is intentionally reproducible but imperfect: abbreviation-like punctuation can occasionally create spurious boundaries, which we treat as a preprocessing limitation rather than manually correcting examples.

3.2 Future prediction head

Let the causal LM process the concatenation of the context and its following sentence. We separately tokenize the context to identify its boundary index. Let $H^{(L)}$ be the final transformer-layer hidden states and i_c the last context-token index after truncation. The context representation is

$$h_c = H^{(L)}_{i_c}. \tag{2}$$

A two-layer trainable projection maps this vector to the 384-dimensional frozen sentence-encoder space:

$$\hat{z} = \mathrm{norm}(W_2 \tanh(W_1 h_c + b_1) + b_2), \tag{3}$$

where norm denotes ℓ_2 normalization. The target vector is

$$z^+ = \mathrm{norm}(E(s^+)), \tag{4}$$

with the frozen `all-MiniLM-L6-v2` Sentence-Transformers encoder, kept fixed throughout training.

3.3 Contrastive future loss

For a batch of B context–future pairs, we compute

$$a_{ij} = \frac{\hat{z}_i^\top z_j^+}{\tau}, \tag{5}$$

with $\tau = 0.07$. Because all vectors are normalized, the dot product is cosine similarity. The future loss is batch-wise cross entropy:

$$L_{\text{future}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(a_{ii})}{\sum_{j=1}^{B} \exp(a_{ij})}. \tag{6}$$

The core objective is

$$L_{\text{core}} = L_{\text{LM}} + \lambda_f L_{\text{future}}, \tag{7}$$

with $\lambda_f = 0.08$.

3.4 Self-loop hard negatives

We additionally mine loop-heavy model generations. For a generated continuation y, the mining score is

$$R_{\text{loop}}(y) = R_4(y) + R_5(y) + R_{\text{adj}}(y) + R_{\text{pair}}(y), \tag{8}$$

where R_4 and R_5 are repeated 4-gram and 5-gram ratios, and R_{adj} and R_{pair} are semantic repetition diagnostics already stored with the generation artifact. We retain generations with $R_{\text{loop}} \geq 0.08$. Within a retained generation, we first choose a sentence containing the most frequent repeated 4-gram. If no repeated 4-gram exists, we choose the sentence with the highest $R_4 + R_5$. This sentence becomes s^-. Thus, the negative criterion is *loop-heavy generation*, not semantic similarity alone.

The resulting pool contains 280 pairs. Of these, 146 inherit a gold next sentence from WikiText-103 or WritingPrompts generation exports. The remaining 134 originate from existing reranking exports that do not carry a gold next sentence; for those examples, the first generated sentence not selected as the hard negative is used as a pseudo-positive. We report this composition because the pool is a diagnostic supervision set rather than a clean corpus of 280 gold next-sentence triples. No synthetic fallback pairs were needed in the final pool.

For a positive s^+ and hard negative s^-, the margin loss is

$$L_{\text{hn}} = \max\big(0,\ m - \cos(\hat{z}, E(s^+)) + \cos(\hat{z}, E(s^-))\big), \tag{9}$$

with $m = 0.10$. The full v2 objective is

$$L = L_{\text{LM}} + 0.08 L_{\text{future}} + 0.10 L_{\text{hn}}. \tag{10}$$

We do not claim these coefficients are optimal. They are fixed nominal settings from the experimental pipeline; Section 5.4 reports available sensitivity analyses rather than post-hoc tuning them for the camera-ready results.

3.5 Optimization

All main v2 runs initialize from DistilGPT-2 and jointly optimize the causal LM and future projection. We use AdamW with learning rate 5×10^{-5}, weight decay 0.01, gradient clipping at 1.0, a linear learning-rate schedule with 50 warmup steps, 1000 optimization steps, LM/future batch size 16, hard-negative batch size 32, maximum sequence length 192, and seed 7, 42, or 123. The hard-negative pool is shuffled with the run seed and split 80/20, giving 224 training and 56 held-out hard-negative examples per run. Because the split itself is seed-dependent, our multi-seed result should be read as stability across independently seeded training/held-out splits, not repeated evaluation on one identical set of 56 triples.

4 Generation-Time Evaluation

4.1 Five-candidate reranking

The future head is **not** a token-level decoder. For each of 120 WikiText validation prompts, we sample five continuations and select one after generation. Main v2 sampling uses temperature 0.9, top-p 0.95, maximum input length 96, maximum 120 new tokens, repetition penalty 1.0, and no no-repeat-ngram constraint. Candidate seeds follow a deterministic prompt/candidate schedule.

We embed the prompt, complete continuation, and generated sentences with the same frozen MiniLM encoder. Let the generated continuation contain sentences $y_1, \ldots, y_n$. The generic score is

$$s_{\text{generic}}(y) = p(y) - 0.60 r(y) + 0.60 q(y) - 0.05 \ell(y), \tag{11}$$

where

$$p(y) = \cos(E(\text{prompt}), E(y)), \tag{12}$$

$$r(y) = \frac{1}{n-1} \sum_{i=1}^{n-1} \mathbf{1}[c_i > 0.90], \tag{13}$$

$$q(y) = \frac{1}{n-1} \sum_{i=1}^{n-1} \mathbf{1}[0.35 \le c_i \le 0.88], \tag{14}$$

$$c_i = \cos(E(y_i), E(y_{i+1})), \tag{15}$$

$$\ell(y) = \frac{|N_{\text{words}}(y) - 105|}{105}. \tag{16}$$

For a one-sentence continuation we set the adjacent-sentence rates to zero. The progression band is therefore a heuristic for *related but non-duplicate* adjacent sentences: very low similarity suggests an abrupt semantic jump, while very high similarity suggests semantic stagnation.

The learned future feature compares the context-predicted future vector with the first generated sentence:

$$f(y) = \cos(\hat{z}(c), E(y_1)). \tag{17}$$

The future-calibrated score used in the actual main decoding job is

$$s_{\text{future}}(y) = s_{\text{generic}}(y) + 0.10 f(y). \tag{18}$$

We use 0.10 as a small fixed additive coefficient and do not present it as an optimized value. Crucially, the no-future ablation removes this term entirely, so the main causal conclusion does not depend on arguing that 0.10 is optimal.

4.2 Diagnostics and controls

Our automatic diagnostics are defined directly from the same measurements: PSim $= p(y)$; topic drift $= 1 - p(y)$; Prog $= q(y)$; LexLoop is repeated-4-gram ratio plus repeated-5-gram ratio; semantic-loop failure is triggered when adjacent semantic repetition exceeds 0.10 or pairwise semantic repetition exceeds 0.05; and WeakProg is triggered when a continuation has at least two sentences and $q(y) < 0.55$. These are controlled diagnostics, not substitutes for human evaluation. Topic drift is algebraically redundant with PSim, so the main tables report PSim and omit the duplicate column.

We compare raw baseline, raw core, raw v2, generic v2 c5, a scale-matched random future-score control, future scores shuffled within a prompt, future scores shuffled across prompts, future-calibrated c5, a lexical-penalty c5 variant,

Domain	Seed	LM ↓	Future ↓	Gap ↑	Sat. ↑
WikiText	7	3.9395	2.7228	.0584	.3393
WikiText	42	3.9384	2.7180	.0614	.4107
WikiText	123	3.9386	2.7347	.0599	.3571
WritingPr.	7	3.9437	2.8010	.0687	.3929
WritingPr.	42	3.9436	2.8237	.0736	.3929
WritingPr.	123	3.9428	2.8173	.0606	.3929

Table 1: Training-side diagnostics after 1000 steps. Gap is mean positive cosine minus hard-negative cosine. Sat. is the fraction of held-out triples whose positive–negative gap reaches the training margin $m = 0.10$. Each seed also induces its own 80/20 hard-negative split.

future-only selection, and a training-free matched Base-c5 control. Random scores are sampled to match the empirical mean and standard deviation of real v2 future scores; shuffling preserves the observed score distribution while breaking candidate alignment.

5 Results

5.1 The training-side signal is reproducible

Table 1 shows a consistent positive held-out margin gap across both domains. On WikiText, the three gaps lie between .0584 and .0614; on WritingPrompts they lie between .0606 and .0736. This supports a narrow representation-learning claim: under the stated objective, the context-conditioned projection learns to score positives above the mined negatives on held-out triples. It does *not* establish that the hard-negative term uniquely causes the effect, because we do not include a shuffled-hard-negative training control.

We also train WikiText seed 42 for fewer steps from the same setup. At 250, 500, and 1000 steps, the held-out gap is .0262, .0491, and .0614, respectively; the corresponding margin-satisfaction rates are .0714, .3571, and .4107. The signal is therefore detectable before the final checkpoint and strengthens during training.

5.2 Generation gains are mostly selection gains

Table 2 separates training effects from candidate-selection effects. Raw v2 changes the failure profile but does not consistently improve all diagnostics. In contrast, selecting among five candidates yields large improvements in prompt similarity and progression. Yet the learned future term is not the specific explanation: generic c5, random future scores, shuffled future scores, and the aligned future score remain close.

The matched Base-c5 comparison strengthens this conclusion. An earlier Base-c5 artifact used different generation length and temperature settings, which made the comparison unnecessarily confounded. We therefore regenerated the baseline with the same main v2 sampling settings and the same candidate-seed schedule. The matched training-free baseline reaches .5080 PSim, .8765 Prog, .0083 semantic-loop rate, and .0583 weak-progression rate, slightly outperforming v2 future c5 on all five reported diagnostics. We interpret this as evidence that candidate generation plus generic reranking is a strong independent mechanism. It does not invalidate the training-side signal; it prevents attributing five-candidate gains to that signal.

5.3 The future term is weakly coupled to selection

On the fixed v2 five-candidate pool, removing the future term changes only 2 of 120 selections. The no-future and with-future rows have PSim.4949 vs. .4954 and Prog.8735 vs. .8727. A prompt-level paired bootstrap gives a with-future minus no-future PSim difference of 0.00047 with a 95% interval $[-0.00009, 0.00150]$; for progression the difference is -0.00080 with interval $[-0.00238, 0]$. Future-only selection raises mean future similarity from approximately .277 to .337, but lowers PSim to .4370, Prog to .6446, and raises semantic-loop and weak-progression rates to .0833 and .3583. The learned feature therefore contains information that can be maximized, but maximizing it is not equivalent to selecting the best continuation under the present diagnostics.

At candidate level, future similarity has descriptive Spearman correlations of .3569 with prompt similarity, .1548 with progression, $-.0692$ with semantic-loop failure, .0302 with lexical-loop penalty, and .3029 with an oracle-style semantic score. Because candidates are grouped five-per-prompt, we do not treat naive 600-candidate p-values as independent-sample significance tests. A complementary prompt-level diagnostic reaches only 18.3% top-1 agreement between future-score selection and oracle semantic selection, with mean within-prompt Spearman correlation .111. These results support the same calibration-gap interpretation without relying on anti-conservative candidate-level inference.

Method	PSim ↑	Prog ↑	LexLoop ↓	SemLoop ↓	WeakProg ↓
Baseline raw	.4197	.6675	.1130	.0667	.3750
Core raw	.4177	.6674	.1146	.0917	.3583
v2 raw	.4274	.6361	.0770	.0667	.4000
v2 generic c5	.4954	.8727	.1024	.0250	.0667
v2 random future	.4958	.8721	.1018	.0267	.0683
v2 shuffled within prompt	.4927	.8768	.0997	.0250	.0583
v2 shuffled across prompts	.4949	.8735	.1022	.0250	.0667
v2 future c5	.4954	.8727	.1027	.0250	.0667
v2 future c5 + lexical control	.4913	.8672	.0546	.0083	.0667
Matched Base c5, no future	**.5080**	**.8765**	**.0817**	**.0083**	**.0583**

Table 2: Generation diagnostics on 120 WikiText prompts. The matched Base-c5 control uses the same 5-candidate count, temperature .9, top-p .95, maximum input length 96, maximum 120 new tokens, and candidate seed schedule as the main v2 decoding setup. Generic, random, shuffled, and learned-future v2 selectors remain close.

5.4 Coefficient sensitivity

The generic reranker contains heuristic coefficients, so we evaluate whether the main conclusion depends on one narrow setting. Holding the progression weight at its nominal .60, varying the repetition weight from .30 to .90 and the length weight from 0 to .10 changes only 0–2 of 120 selected candidates relative to the nominal $w_r = .60, w_q = .60, w_\ell = .05$ selector. Changing the progression weight has a larger effect: $w_q = .30$ or .90 changes roughly 10–20 selections, as expected because progression is a major term in the selector. The important attribution result is stable in the local nominal neighborhood: repetition and length coefficients are not acting as fragile hidden switches.

We also have a training-side sensitivity over $\lambda_h \in \{.05, .10, .20\}$ for WikiText seed 42. LM loss remains effectively unchanged (3.9385, 3.9384, 3.9383), while the held-out gap increases (.0512, .0614, .0701). We therefore use .10 as the nominal run rather than claiming it is optimal; the existence of a positive separation signal is not unique to that exact hard-negative weight. We do not have an analogous multi-value sweep for $\lambda_f = .08$, which remains a limitation.

5.5 Concrete examples

Table 3 illustrates the hard-negative construction. The negative is not simply a semantically related next sentence; it is extracted from a visibly repetitive model continuation. The example also highlights why the signal should be interpreted as a degeneration diagnostic rather than a general theory of discourse: a human-written restatement could be legitimate even when semantically similar.

Generation examples tell the same story. For a prompt describing lobster "crusher" and "cutter" claws, generic c5 and future-calibrated c5 select the same continuation containing repeated "crusher" phrases, whereas the lexical-control selector chooses a different, less lexically repetitive candidate. Conversely, several prompts show generic and future c5 choosing exactly the same candidate, and failure cases remain in which future-calibrated c5 produces entity confusion or topical drift. We therefore use qualitative examples only to make the experimental conditions concrete, not as evidence overriding the aggregate controls.

6 Discussion

The experiments distinguish three levels of evidence. First, the auxiliary representation is learnable: the predicted future vector scores held-out positives above mined negatives across seeds and domains. Second, the signal is semantically structured: it correlates with prompt anchoring and an oracle-style semantic score. Third, behavioral usefulness is much weaker: under five-candidate reranking, the future term changes almost no selections and does not outperform generic or matched training-free controls.

This is a useful negative result for small-model analysis. If we had reported only raw-v2 versus future-c5 generation, the improvement could easily be described as evidence that sentence-level foresight improves coherence. The random, shuffled, no-future, future-only, coefficient-sensitivity, and matched Base-c5 controls show that such wording would be too strong. The present model learns a direction in sentence-embedding space, but it does not learn a complete decoding policy. A good continuation must balance prompt relevance, semantic progression, non-repetition, length, factual consistency, and entity continuity; a single

Context	The closest relative of *H. gammarus* is the American lobster, *Homarus americanus*.
Gold next sentence	The two species are very similar, and can be crossed artificially, although hybrids are unlikely to occur in the wild since their ranges do not overlap.
Mined hard negative	*H. gammarus* is the American lobster, known as the American lobster, known as the American lobster, known as the American lobster, . . .

Table 3: One WikiText hard-negative example from the final mined pool. The negative is chosen from a high-repetition model generation.

future-similarity scalar captures only part of that structure.

The results also clarify the role of self-loop negatives. We use highly repetitive generations because they are frequent weak-model failures and provide cheaply mined contrast. We do *not* assume that all repetition is bad or that every semantically similar next sentence should be pushed away. The hard-negative builder can produce false positives with respect to discourse quality, and 134 of 280 pool items use pseudo-positives rather than gold next sentences. Consequently, our strongest supported statement is about learnable loop separation under this diagnostic construction, not about a universal notion of future coherence.

7 Limitations

Our evaluation is automatic and covers 120 WikiText prompts; we do not report human judgments of coherence, factuality, or discourse quality. The frozen MiniLM target space can miss entity consistency, factual contradictions, and discourse relations. The sentence splitter is rule-based and occasionally treats abbreviation-like punctuation as a boundary. The hard-negative pool is small and mixed-source, and 134/280 triples use a generated pseudo-positive because their source reranking exports lack a gold next sentence. Each training seed also uses a different 80/20 hard-negative split, so seed stability combines optimization and split variation. We do not run a shuffled-hard-negative training control, a gold-only hard-negative training control, or a λ_f sweep; these are the most direct ways to test whether the specific auxiliary construction, rather than easier repetition artifacts, causes the training-side gap. Finally, the matched Base-c5 result shows that the learned future term is not necessary for the strongest automatic generation scores in this setup. Accordingly, we make no claim that ForesightLM-v2 solves coherence or that its future term is the principal cause of generation improvements.

8 Conclusion

ForesightLM-v2 demonstrates a distinction between learning a future-oriented sentence representation and using that representation behaviorally. A DistilGPT-2-scale model trained with next-sentence contrastive prediction and self-loop hard negatives learns a consistent held-out positive–negative margin across WikiText-103 and WritingPrompts seeds. Yet under a controlled five-candidate generation interface, the learned future score has little independent effect: removing it changes only 2/120 selections, random and shuffled controls remain close, and a matched training-free Base-c5 reranker slightly outperforms it on the reported automatic diagnostics. We therefore interpret sentence-level foresight as a measurable but incompletely calibrated representation. For small-model research, this gap is itself informative: acquiring a diagnostic signal is easier than converting it into reliable generation behavior.

References

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture mod-

els. In *International Conference on Learning Representations*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992. Association for Computational Linguistics.

Yixuan Su, Tian Lan, Yan Liu, Fangyu Liu, Dani Yogatama, Yan Wang, Lingpeng Kong, and Nigel Collier. 2022. A contrastive framework for neural text generation. In *Advances in Neural Information Processing Systems*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

A Reproducibility Details

Training setting	Value
Base checkpoint	DistilGPT-2
Sentence encoder	all-MiniLM-L6-v2
Sentence embedding dim.	384
Train pairs/domain	30,000
Validation pairs/domain	2,000
Training steps	1000
LM/future batch size	16
Hard-negative batch size	32
Max sequence length	192
Learning rate	5×10^{-5}
AdamW weight decay	0.01
Warmup steps	50
Gradient clipping	1.0
λ_f	0.08
λ_h	0.10
Contrastive temperature τ	0.07
Hard-negative margin m	0.10
Hard-negative pool	280
Hard-negative train/val	224 / 56

Table 4: Main training configuration.

B Additional Sensitivity Results

Generic-score neighborhood. On the fixed v2 five-candidate pool, with $w_q = .60$, all combinations $w_r \in \{.30, .60, .90\}$ and $w_\ell \in \{0, .05, .10\}$ differ from the nominal selector on at most two of 120 prompts. For example, $(w_r, w_q, w_\ell) = (.30, .60, .05)$ changes one selection and gives PSim.4981, Prog.8707, SemLoop .0333, and WeakProg.0667; $(.90, .60, .05)$ changes zero selections relative to the nominal candidate IDs and therefore has the same selected outputs. Increasing w_q to .90 changes 10 selections and increases Prog to approximately .896 while lowering PSim to approximately .478; decreasing it to .30 changes 16–20 selections and has the opposite trade-off. This confirms that progression weighting matters, while the local repetition/length settings are not fragile.

Generation setting	Value
Prompts	120
Candidates/prompt	5
Temperature	0.9
Top-p	0.95
Max input tokens	96
Max new tokens	120
Repetition penalty	1.0
No-repeat n-gram size	0
Future rerank coefficient	0.10
Generic w_r	0.60
Generic w_q	0.60
Generic w_ℓ	0.05
Length target	105 words

Table 5: Main v2 five-candidate generation and reranking configuration. The matched Base-c5 control uses the same sampling values.

λ_h	LM ↓	Future ↓	Gap ↑	Sat. ↑
.05	3.9385	2.7214	.0512	.3571
.10	3.9384	2.7180	.0614	.4107
.20	3.9383	2.7148	.0701	.4643

Table 6: Hard-negative loss-weight sensitivity for WikiText seed 42.

C Future-Term Ablation Details

Selector	PSim ↑	Prog ↑	Sem. ↓	Weak ↓	Fut. ↑
v2 c5, no future	.4949	.8735	.0250	.0667	.2751
v2 c5, + future	.4954	.8727	.0250	.0667	.2769
Future only	.4370	.6446	.0833	.3583	.3368

Table 7: Fixed-pool future-term ablation. The future term changes only 2/120 selections.

D Qualitative Examples

Generic and future selectors often agree. For the prompt *"It is closely related to the American lobster, H. americanus."*, generic c5 and future-calibrated c5 select the same candidate. The continuation remains topically related but contains factual and taxonomic noise. This example illus-

trates both candidate-selection gains over raw decoding and the lack of a unique future-term effect.

Lexical control addresses a different failure. For the prompt describing "crusher" and "cutter" lobster claws, generic c5 and future c5 select a continuation that repeats variants of *"The crusher is a variation of the crusher"*. Adding the lexical penalty chooses a different candidate and lowers the measured surface-loop score. This matches the aggregate result that future similarity and lexical repetition are only weakly associated.

Failure case. Some future-calibrated selections remain semantically or factually confused. One taxonomy prompt yields repeated names and historically inconsistent entities despite a reasonable semantic score. We include such cases because the automatic metrics should not be mistaken for factual or human-level coherence judgments.

Hard-Wiring the Induction Head: The Surprising Effectiveness of Copy Mechanisms in Data-Efficient Language Modelling

Josh Taylor
British Standards Institution
https://www.linkedin.com/in/josh-taylor-24806975/

Abstract

Large language models need several orders of magnitude more text than a child hears to reach a comparable level of linguistic skill. One reason is that a particularly useful capability, the *induction head* (a circuit that completes a pattern of the form [A] [B] . . . [A] by copying [B]), is not built in but must emerge during training, and does so only after the model has read a great deal of text. We ask whether supplying this capability directly, as a fixed part of the architecture, improves learning when data is scarce. We present a three-layer, attention-free language model of about 12M parameters in which all long-range mixing is performed by a non-learned, sub-quadratic induction operator (its cost grows more slowly than the square of the sequence length): for each position it finds recent earlier occurrences of the same token and copies the continuation of the best match. Trained on the 10M-word BabyLM Strict-Small corpus, the model learns markedly faster than a parameter-matched self-attention baseline and stays competitive at convergence; on the public BabyLM evaluation it is competitive with the challenge's GPT-2 baselines on grammatical minimal pairs with a tenth of the parameters. Targeted interventions show the operator is used as intended: predictions travel along its links, the links are necessary, and stacking layers composes them into two-step retrieval. Because the operator forms no attention matrix, it is also cheaper at length: on a single GPU it overtakes causal Flash attention at about 1,000 tokens of context and is $3\times$ faster and $9\times$ smaller in memory at 8,192, where attention approaches the memory ceiling. These results motivate hybrid designs that interleave exact look-back with attention.

1 Introduction

Children learn to understand and produce language after hearing on the order of tens of millions of words. Modern language models reach comparable fluency only after training on three to four orders of magnitude more text (Warstadt et al., 2023). The BabyLM Challenge (Warstadt et al., 2023; Charpentier et al., 2025) turns this contrast into a benchmark by fixing a child-sized training budget; we work in its STRICT-SMALL setting, roughly 10 million words.

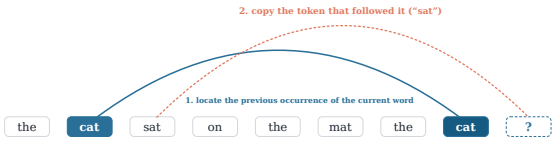


Figure 1: The fixed look-back-and-copy rule. To predict the token after the second "cat," the operator finds the previous "cat" and copies the token that followed it.

Part of the gap comes from models having to rediscover useful machinery for themselves. A clear example is the *induction head*, a circuit first identified by Elhage et al. (2021) and analysed by Olsson et al. (2022). Given a repeated pattern [A] [B] . . . [A], an induction head predicts [B] by finding the earlier occurrence of [A] and copying what followed it. In a standard transformer (Vaswani et al., 2017) this circuit is absent at the start; it appears abruptly partway through training, only once the model has seen substantial data (Olsson et al., 2022). Because the same circuit underpins much of a model's ability to reuse information from earlier in its input, its late arrival is costly when data is scarce.

If a mechanism is this useful and this well understood, there is no obvious reason to make every model learn it from scratch. We supply it directly: we replace the learned long-range mechanism with a fixed operator that performs the look-up and copy explicitly (Figure 1), and ask whether this helps at BabyLM scale.

All code, training scripts and evaluation har-

Proceedings of the Second BabyLM Workshop, pages 24–32
October 29, 2026 ©2026 Association for Computational Linguistics

nesses are publicly available.[1]

Contributions. (1) An attention-free, subquadratic architecture combining a fixed induction operator for long-range copying with a dynamic convolution for local mixing. (2) Evidence that, on a 10M-word budget, this model is substantially more data-efficient than a parameter-matched self-attention model, remains competitive at convergence, and holds its own against the challenge's GPT-2 baselines on the public benchmark at a tenth of the size. (3) Targeted interventions showing the operator is genuinely used and composes across layers into two-step retrieval. (4) A measured efficiency comparison against causal Flash attention across context lengths, locating the crossover at about 1,000 tokens. (5) An account of the alternative designs that failed on the way. (6) A direction for hybrids that interleave exact look-back with attention.

2 Background

Attention and its cost. Self-attention (Vaswani et al., 2017) computes, for every position, a weighted average over all other positions. Because every position is compared with every other, computation and memory grow with the square of the sequence length.

Induction heads. Induction heads (Elhage et al., 2021; Olsson et al., 2022) are responsible for much of a transformer's in-context copying. A key property is that the circuit needs at least two layers: one to attach to each token a record of what preceded it, and a second to use that record to copy forward (Olsson et al., 2022).

Efficient sequence mixing. Dynamic convolutions (Wu et al., 2019) predict a position-specific filter from the current token and apply it to a local window, scaling linearly with length. Long implicit convolutions extend the filter across the sequence (Poli et al., 2023). Linear attention (Katharopoulos et al., 2020) and state-space models such as Mamba (Gu and Dao, 2023) summarise the past in a compact running state. A consistent recent finding is that this family struggles precisely on *recall*: fixed-size-state architectures have difficulty retrieving specific earlier tokens, on synthetic tests and real language (Arora et al., 2024) and on explicit copying (Jelassi et al., 2024). Our design takes that seriously: the local mixer follows the dynamic-convolution approach, while the long-range mixer stores no compressed state at all and instead looks up exact token matches.

For the remaining components we use standard choices: a gated feed-forward block of the SwiGLU family (Shazeer, 2020) and rotary position embeddings (Su et al., 2021).

3 Architecture

The model has three identical layers, each applying in order: a dynamic depthwise convolution for local mixing; the induction operator for exact long-range copying; and a gated feed-forward block. It has about 12M parameters, width 384, and a vocabulary of 8,000 subword units learned from the training text. Figure 2 shows the design.

The induction operator. For each position, the operator finds the most recent earlier positions holding the same token (exact match on token identity; the five most recent matches are kept as candidates). A small learned scoring function compares the context around the current position with the context around each candidate and ranks them; the token that followed the best-ranked candidate is copied forward, without averaging over candidates. Two properties matter. The operator is *not learned*: the look-up is computed directly from the token sequence, so it works from the first training step; only the ranking is trained. And it is *cheap*: matches are found by sorting token identities, so cost grows as the sequence length times its logarithm rather than its square, and no attention matrix is formed.

The division of labour is deliberate: the convolution handles immediate neighbours, the operator handles exact reuse at any distance, and the feed-forward block transforms the result. As Section 5.3 shows, local context written by the convolution is also carried along the induction links, so the two mixers cooperate rather than duplicate each other.

4 Experimental setup

Data. We train on the BabyLM STRICT-SMALL corpus (Warstadt et al., 2023), about 10M words of child-directed and child-accessible English, tokenised into sequences of 256 tokens. Total training exposure is about 98 million tokens, roughly six passes over the corpus, within the challenge's ten-pass limit.

Models. We compare against a baseline identical in every respect except that the induction operator is replaced by multi-head self-attention (Vaswani

[1] https://github.com/joshua-taylor/conv-induction-babylm

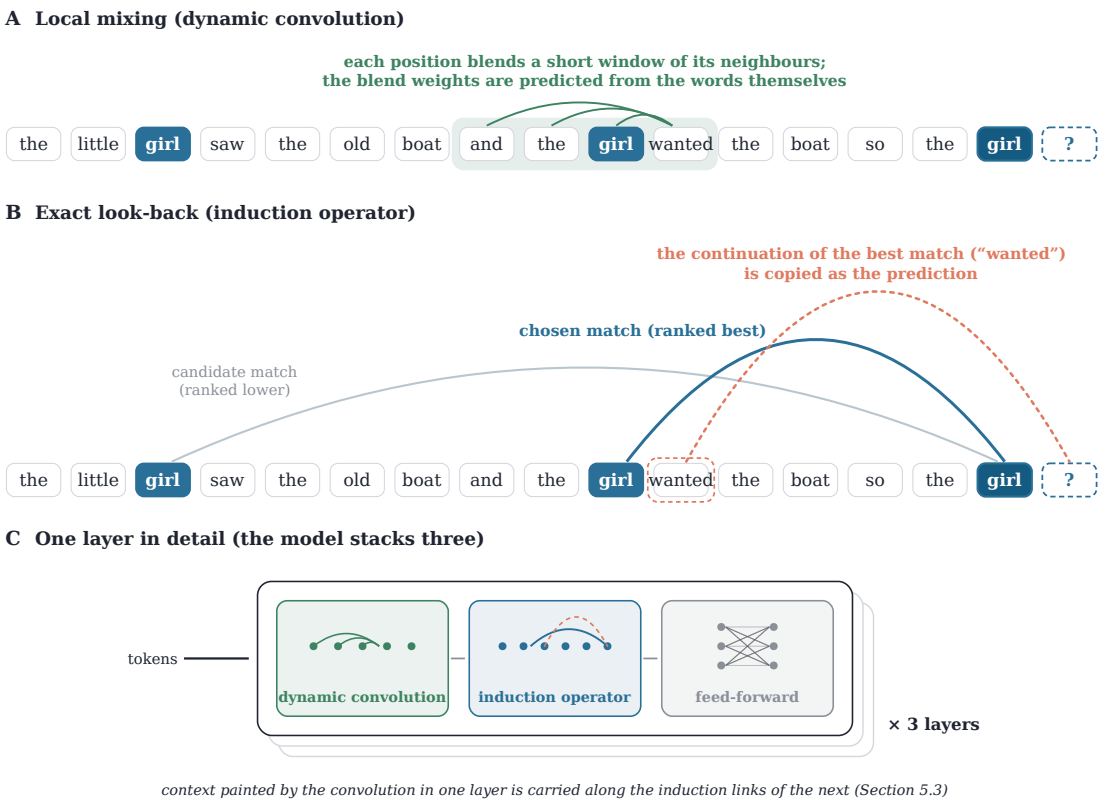


Figure 2: The full model. (A) A dynamic convolution blends each position with a short window of neighbours, with blend weights predicted from the words themselves. (B) The induction operator gathers recent earlier occurrences of the current token as candidates, ranks them, and copies the continuation of the best match; here the third "girl" looks back to two earlier "girl" positions and copies "wanted." (C) One layer in detail; the model stacks three. Context written by the convolution in one layer is carried along the induction links of the next.

et al., 2017). The two models are parameter-matched (12.3M learnable weights each), so differences reflect the mechanism, not capacity. Each is trained with three random seeds.

Reporting. Our main measure is validation perplexity (how surprised the model is by held-out text; lower is better). To measure copying directly we build sequences whose second half repeats their first half and score how often the model predicts the repeated continuation. Benchmark scores in Section 5.5 come from the official BabyLM evaluation pipeline.

5 Results

5.1 Data efficiency and competitiveness

Figure 3 (left) tracks validation perplexity as each model sees more text. Early in training our model is far ahead: after about 10M words its perplexity is 57, against 78 for the attention baseline, which closes the gap only after roughly ten times as much text. The right panel explains why. On the copying probe our model succeeds from the start (accuracy around 0.5), whereas the attention model barely learns to copy, rising from almost zero to about 0.05 even after 100M words. The fixed operator hands the model a working copy mechanism immediately; the attention model has to grow one and, at this scale, largely fails to.

At convergence the attention model reaches slightly lower perplexity (31.9 vs. 34.3, three seeds, spread about 0.5; Table 1). This is the expected price of a fixed mechanism: given enough data, a fully learned component is more flexible. The point we draw is that a non-learned, sub-quadratic long-range operator keeps the model within about seven percent of a strong attention baseline while making it considerably faster to learn from scarce data.

5.2 What each component contributes

We removed the components one at a time, retrained, and measured the rise in perplexity (Figure 4). This is a deletion test rather than a parameter-matched comparison, but the pattern is

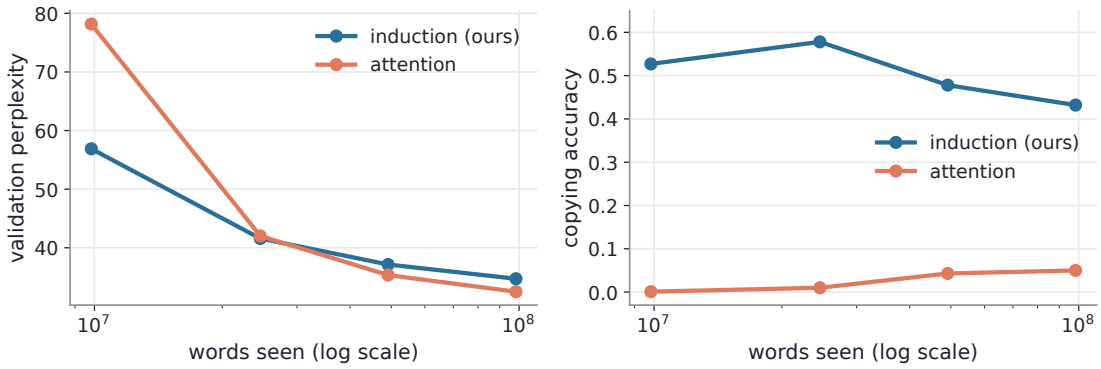


Figure 3: Left: our model is far more data-efficient early and is overtaken only after about ten times more text. Right: our model copies from the outset; the attention model barely learns to within the budget.

Model	Params	Perplexity ↓
Induction operator (ours)	12.3M	34.3
Self-attention baseline	12.3M	31.9

Table 1: Parameter-matched comparison at the full budget, averaged over three seeds (spread ∼0.5).

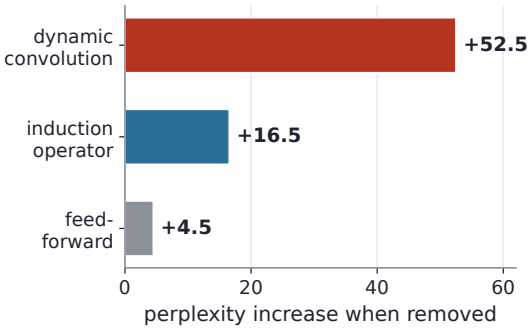


Figure 4: Removing the convolution is most damaging; removing the induction operator clearly hurts; removing the feed-forward block hurts least, despite holding the most parameters.

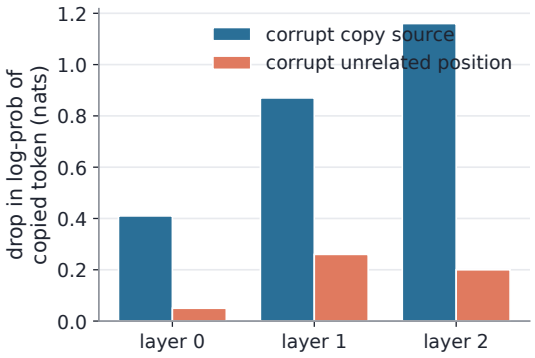


Figure 5: Corrupting the position a prediction copies from is far more damaging than corrupting an unrelated position the same distance back, and the effect grows with depth.

informative. Removing the convolution is most damaging (34 to 87); removing the induction operator clearly hurts (to 51); removing the feed-forward block costs least (to 39) despite it holding the largest share of parameters. The purpose-built mixers, not raw capacity, contribute most to model performance.

5.3 Is the operator used as intended?

A built-in tool is only worthwhile if the model relies on it. We ran four interventions. Throughout, a *link* means one use of the operator: the connection from a position to the earlier occurrence it reads from.

Predictions travel along the links. For each prediction we identified the exact earlier position it was copying from and corrupted the model's internal representation there, replacing it with an uninformative average. The model's confidence in the copied token fell sharply; corrupting an unrelated position the same distance back changed almost nothing (Figure 5). The effect grows from early to late layers, indicating the copy is assembled towards the end of the network. This held across more than two thousand cases.

The links are necessary. The first intervention corrupted the source; the second removes the link itself. For each prediction we identified the candidate the operator relied on most, then deleted it from the operator's candidate list, so the model could no longer read from that position through the operator while everything else was left intact. Confidence in the correct token fell by about three nats across two hundred cases (a nat is the natural unit of log-probability; three nats means the assigned probability falls by a factor of about twenty). As a control, on the predictions where one existed we

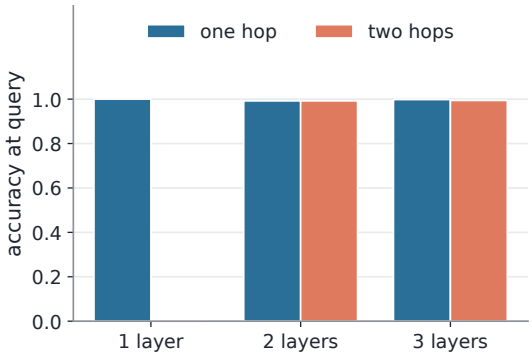


Figure 6: One layer solves a single look-up but not a two-step chain; two layers solve both.

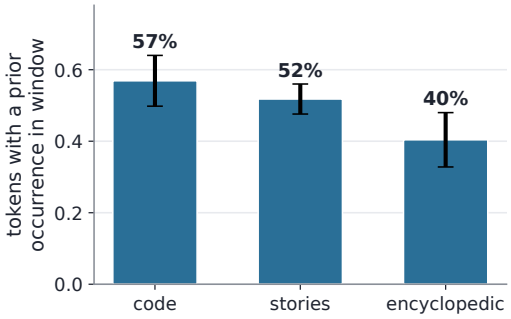


Figure 7: Token repetition is common in every genre examined; this is the structure the operator exploits.

instead deleted a rarely-used candidate whose following token differed from the answer: this cost essentially nothing, while deleting the real link on the same predictions cost about four nats. Removing the link that matters collapses the prediction; removing another changes nothing. Most strong long-range links sat on genuinely repeated spans of text, indicating the operator responds to real repetition rather than coincidence.

Stacking layers enables two-step retrieval. We built a synthetic test in which the answer can only be reached in two steps: the query pattern points to an intermediate token, and the answer is the continuation of that intermediate token's own earlier occurrence. A one-layer model handles a single look-up perfectly but fails the chain, returning the intermediate token instead; a two-layer model solves both (Figure 6), matching the known two-layer requirement for induction circuits (Olsson et al., 2022). In practice, chained look-ups are a primitive form of combining separated information. A reader does this when a passage says the captain owns the boat and, later, that the boat is called the Maria: connecting "captain" to "Maria" requires passing through "boat." A single look-up can only re-surface what sat next to a cue; chaining lets facts stated in different places be combined, and our fixed operator gains this ability exactly when layers are stacked.

What the links carry. The copied representation identifies the source token almost perfectly (about 0.98, by a linear read-out). It also carries the source's immediate left-hand context, but only when the convolution is active: the word to the left of the source is recoverable 76% of the time with the convolution on, against 16% with it off. The convolution writes local context onto a token; the operator transports the package across the gap.

How much is free. A rule with no learning, copying the continuation of the most recent match, already gets 23% of these predictions right; the trained model reaches 31%. Most of the copying ability is supplied free by the fixed operator, with learning adding a useful but secondary refinement in the ranking.

5.4 The pattern in natural text

The approach pays off only if real text repeats itself. It does: across three kinds of writing, 40–57% of tokens have already appeared earlier in the same 256-token window (Figure 7). Code repeats most (57%), children's stories next (52%), encyclopedic prose least (40%), though its repeats reach back furthest. Appendix A visualises the operator's strongest links on a sample sentence: they land on repeated content words, tying each occurrence to its earlier mention.

5.5 Benchmark evaluation

We evaluated with the official BabyLM pipeline and submitted to the public leaderboard. Table 2 compares against the challenge's official baselines: two GPT-2 entries, standard 124M-parameter transformers (Radford et al., 2019) roughly ten times our size, and the GPT-BERT baselines, which combine masked and causal training and were the strongest systems of the previous challenge. The metrics: BLiMP and its supplement test grammar by asking whether the model prefers a grammatical sentence over a minimally different ungrammatical one (Warstadt et al., 2020); EWoK tests basic world knowledge the same way; entity tracking tests whether the model follows the state of objects through a passage; COMPS tests knowledge of concept properties; (Super)GLUE is a suite of

	BLiMP	BLiMP Sup.	EWoK	Entity Tr.	COMPS	GLUE
Ours (attention-free, 12M)	65.7	55.5	**51.6**	27.6	50.7	61.8
GPT-2 (strict-small)	63.1	57.3	–	21.1	**51.8**	63.8
GPT-2 (10M baseline)	66.4	57.1	49.9	13.9	–	55.9
GPT-BERT (causal)	**71.7**	63.2	49.5	34.6	–	65.1
GPT-BERT (mntp)	71.3	**63.3**	49.9	**40.0**	–	**65.7**

Table 2: Public BabyLM leaderboard scores against the official baselines (higher is better; bold marks the best score in each column; a dash means that evaluation is not reported on the leaderboard for that entry). The GPT-2 entries are standard 124M-parameter transformers (Radford et al., 2019); the GPT-BERT baselines combine masked and causal training, and we show the strongest of their six configurations under each scoring mode (mntp = masked next-token prediction scoring). On the minimal-pair tasks chance is 50, and all models at this scale sit near it on the knowledge-heavy tasks (EWoK, COMPS). We omit the leaderboard aggregate and age-of-acquisition metric, which depend on a fuller checkpoint series than we uploaded; the columns shown are computed identically for all models.

language-understanding tasks on which the model is fine-tuned.

Three things stand out. First, an attention-free model whose only long-range mechanism is fixed is competitive with the GPT-2 baselines on the core grammar test: our BLiMP score of 65.7 sits between the two GPT-2 entries (63.1 and 66.4) at a tenth of their parameters, and we are well ahead of both on entity tracking (27.6 vs. 21.1 and 13.9), the task closest in spirit to what the operator provides: following a specific thing through a passage. Second, the GPT-BERT baselines remain clearly stronger on grammar and entity tracking; they combine masked and causal training, and interleaving our operator into such models (Section 8) is a natural way to test whether the two advantages add. Third, on the knowledge-heavy tasks every model sits near chance; a 10M-word diet contains little encyclopedic knowledge, whatever the architecture.

6 Efficiency

The long-range operator forms no attention matrix: matches are found by sorting token identities, so computation and memory grow as the sequence length times its logarithm rather than as its square. To measure what this is worth we timed two three-layer stacks that are identical except for the sequence mixer, one using the induction operator and one using causal self-attention through PyTorch's fused (Flash) kernel, on a single NVIDIA T4 at batch size 4 in half precision, building the index once per forward pass as the trained model does.

Figure 8 shows the result. At the 256-token training length attention is about $1.8\times$ faster: the quadratic term is still small and the fused kernel is well optimised, while our operator pays a fixed cost for sorting and gathering. The two cross at roughly 1,000 tokens, after which the gap widens, reaching $1.3\times$ at 2,048 tokens, $1.8\times$ at 4,096 and $3.1\times$ at 8,192. Memory is the starker and more practical difference: peak usage grows close to linearly for the operator, from 0.10 GB at 256 tokens to 1.14 GB at 8,192, against 10.43 GB for attention at that length, a $9\times$ gap. At 16,384 tokens attention exhausts the GPU while the induction stack completes the forward pass in 2.20 GB, so on this hardware the model can process contexts the parameter-matched attention model cannot run at all. These are forward-pass measurements at one model width, and the advantage belongs to long context: it is invisible at the 256-token sequences the BabyLM budget calls for.

7 What did not work

The final design is simple, but the route to it was not; we arrived at the exact-match operator largely by elimination. Throughout, we followed a fixed discipline: change one variable at a time, write down the predicted outcome and an abandonment criterion before running, and stop when the criterion was met.

Compressed-state mixers could not do exact recall. Our first long-range mixers summarised the past in a fixed-size state, in the spirit of linear attention. They trained stably but plateaued well above the attention baseline, failing exactly where retrieval had to be precise, mirroring the recall gap documented for this family (Arora et al., 2024; Jelassi et al., 2024). Squeezing history into a fixed-size summary blurs precisely the information an induction step needs.

A learned associative memory collapsed into linear attention. A memory storing key–value pairs in superposition in a single vector was abandoned under the criterion we had written down in advance: its behaviour became indistinguishable

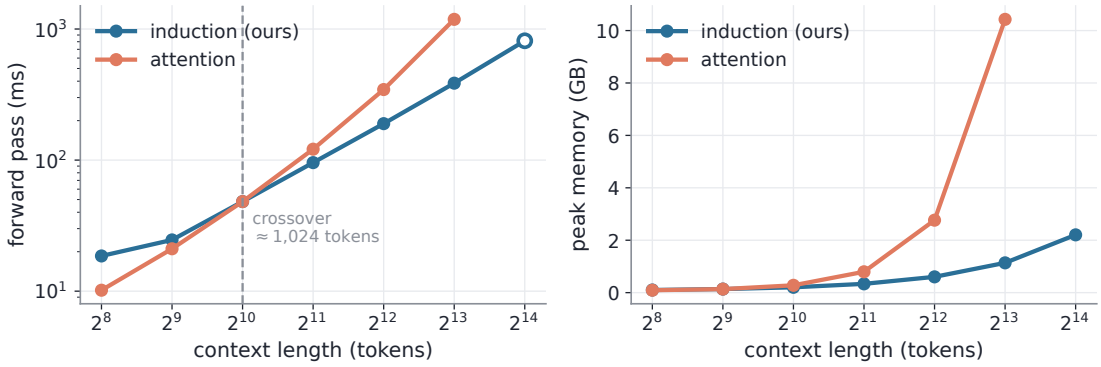


Figure 8: Measured cost against context length for two three-layer stacks differing only in the sequence mixer (NVIDIA T4, batch 4, half precision). Left: forward-pass latency, log–log; the crossover is at about 1,000 tokens, and the open marker at 16,384 is the induction stack completing a length at which attention runs out of memory. Right: peak memory, the more practical limit.

from the linear attention it was meant to improve on. New mathematics over a compressed state does not change the state's fundamental limit.

Counting statistics could not replace the learned ranking. Variants choosing among candidates by frequency or recency alone handled raw copying but fell to near chance on context-dependent grammar such as subject–verb agreement. Choosing which occurrence to trust requires comparing contexts, and that comparison must be learned; the final design keeps learning exactly there and nowhere else in the long-range path.

Fixed convolution filters underperformed content-dependent ones. Convolutions with filters fixed after training, in the spirit of long implicit-filter designs (Poli et al., 2023), were consistently worse than the dynamic form (Wu et al., 2019). At this scale, local mixing must adapt to content.

The search was broad before it was narrow. Beyond the failures above, we implemented and abandoned reservoir-style mixers with large fixed random recurrent states; purely count-based predictors; attention over compressed keys and values; temporal convolutional and trellis-network backbones; and Fourier-transform-based mixing. Each received the same treatment and fell to one of two failure modes: it compressed away the exact retrieval that language rewards, or it added flexibility a 10M-word budget could not train. The breadth of the search is part of the evidence: at this scale, one precise, almost-free mechanism beat every cleverer alternative we tried.

8 Future directions

Interleaving with attention. Exact look-back and attention are complementary. Attention spreads a normalised weight distribution across all positions, so the weight on any one distant token is diluted over long contexts: flexible but smoothed routing. The operator provides a precise pointer to one exact position. Interleaving induction and attention layers would save parameters and give attention an absolute look-back it does not naturally have, a second, sharper route for information. Hybrids that add a precise retrieval path to an efficient backbone recover most of the recall gap to attention (Arora et al., 2024), supporting the direction.

Longer context. Section 6 shows the operator is already cheaper than fused attention beyond about a thousand tokens, and can run contexts that exhaust attention on the same hardware. Its natural home is therefore long-context modelling. Training at those lengths, rather than only measuring the forward pass, is the obvious next step; the BabyLM budget gave us no reason to train beyond 256 tokens.

9 Limitations

At the full budget the attention baseline reaches slightly lower perplexity, so the contribution is data efficiency rather than an outright win. In the component study (Section 5.2), removing a component also removes its parameters, so those comparisons measure the value of a component, not of a mechanism at fixed size. In Section 5.3, the comparison between cutting the most-used link and cutting an unused one was only possible on the small number of predictions that had a suitable unused link, so those particular numbers carry more uncertainty

than the headline result. The two-step result uses a synthetic task: it shows the mechanism can chain, not that natural text requires chaining. Benchmark differences on the knowledge-heavy tasks sit near chance for all models at this scale and should not be over-read. Finally, the efficiency measurements in Section 6 are forward-pass timings of one model width on one GPU, and the advantage only appears beyond about a thousand tokens: at the 256-token sequences this model is trained on, attention is the faster of the two. We have not trained at long context, so the efficiency result speaks to the cost of the mechanism, not to the quality of a long-context model built from it.

10 Conclusion

Building the induction head in, rather than waiting for it to emerge, makes a small language model markedly more data-efficient while keeping it competitive with a parameter-matched attention model, and it holds its own against transformer baselines ten times its size on the public BabyLM grammar tests. The operator is genuinely used, necessary, and composes across layers into two-step retrieval; it rests on repetition, a pattern that pervades real text; and being attention-free it is also cheaper than fused attention beyond about a thousand tokens of context, running lengths that exhaust attention on the same GPU. Some capabilities are worth supplying directly when data is scarce, and an exact look-back may complement attention rather than merely replace it.

References

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2024. Zoology: Measuring and improving recall in efficient language models. In *International Conference on Learning Representations*. ArXiv:2312.04927.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candance Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, Suzhou, China. Association for Computational Linguistics.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. Repeat after me: Transformers are better than state space models at copying. In *International Conference on Machine Learning*. ArXiv:2402.01032.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, et al. 2022. In-context learning and induction heads. *Transformer Circuits Thread*. ArXiv:2209.11895.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. 2023. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*. ArXiv:2302.10646.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8).

Noam Shazeer. 2020. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*. ArXiv:1901.10430.

A Qualitative look-back links

Figure 9 shows the operator's strongest links on a sample sentence, one row per layer, with darker and thicker arcs marking links the model relies on more. The strong links land on repeated content words such as "old," "boat," "girl" and "river."

B Model and training details

Each of the three identical layers applies, in order, with pre-norm residual connections: (1) a gated dynamic depthwise convolution, in which a small network predicts, from each token, the weights of three short depthwise filters (kernel size 3, dilations 1, 2 and 4, 16 weight-sharing groups) whose outputs are summed and gated; (2) the induction operator, which indexes, for every position, the five most recent earlier positions holding the same token (computed exactly, without learning, by sorting token identities), scores each candidate with a learned multi-head comparison between the query context and the candidate context, and copies forward the representation at the position immediately after the best-scoring candidate, with a learned sink allowing the operator to abstain; and (3) a SwiGLU feed-forward block. Token order is encoded with learned position embeddings up to 512 positions. Table 3 lists the configuration.

Hyperparameter	**Value**
Layers	3
Model width	384
Attention-equivalent heads	4
Feed-forward width	1,536
Vocabulary (byte-level BPE)	8,000
Sequence length	256
Max position embeddings	512
Conv kernel / dilations / groups	3 / 1,2,4 / 16
Induction candidates per position	5
Parameters	12.3M
Batch size	32
Training steps	12,000 ($\sim$6 epochs)
Optimiser: 2D matrices	Muon, lr 0.03
momentum / NS steps	0.95 / 5
Optimiser: other tensors	AdamW, lr 6×10^{-4}
Weight decay / grad clip	0.1 / 1.0
Schedule	300-step warmup, cosine to 5%
Checkpoint selection	BLiMP proxy

Table 3: Architecture and training configuration.

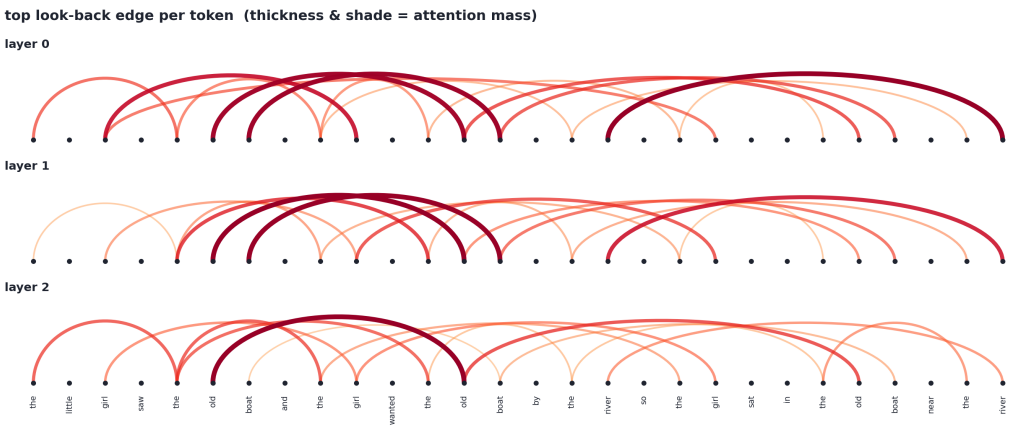


Figure 9: The operator's strongest links on a sample sentence (one row per layer); thicker, darker arcs mark links the model relies on more.

HydraLLM: Adaptive Width Slicing for Elastic Text and Vision-Language Models

Oscar Kemme[1], Janek Haberer[2], Ali Hojjat[1,2], Olaf Landsiedel[1,2,3]
[1]Hamburg University of Technology (TUHH), Germany
[2]Kiel University, Germany
[3]UNU-INWEH, Germany
{oscar.kemme,ali.hojjat,olaf.landsiedel}@tuhh.de
janek.haberer@cs.uni-kiel.de

Abstract

LLMs and LVLMs are increasingly deployed across heterogeneous hardware, ranging from data-center GPUs to resource-constrained edge devices. Their fixed-size architectures limit the ability to reuse a single model across deployment settings with different compute, latency, and model-size targets. We introduce HydraLLM[1], an elastic MQA-based architecture. Its number of active attention heads determines the effective widths of the embeddings, attention projections, feed-forward layers, and the output head, forming a nested family of subnetworks within a single parameter set. We introduce a static-to-elastic conversion pipeline that, starting from a pre-trained model checkpoint, includes importance-driven weight reordering and stochastic subnetwork training with knowledge distillation. We extend this principle to multimodal inference through an elastic projection module whose output width follows the active text width. HydraLLM covers a wider range of active parameter configurations than FFN-only elastic architectures. With 75% of heads active, HydraLLM achieves 2.2 percentage points higher TriviaQA accuracy than a state-of-the-art nested baseline configuration which has 17.25% more parameters. With 50% of heads active, our multimodal model achieves a CLIPScore comparable to that of the 7× larger Gemma 3 4B model. With only 25% of heads active, HydraLLM increases token-generation throughput by 75.3% over the full-width configuration.

1 Introduction

Large Language Models (LLMs) and Large Vision-Language Models (LVLMs) underpin applications such as document understanding (Deitke et al., 2024), conversational assistants (OpenAI, 2024), and agent systems (Google, 2024). Transformer-based architectures (Vaswani et al., 2017) exhibit robust scaling, driving the creation of larger foundation models. At the same time, their deployment spans heterogeneous hardware from data-center GPUs to resource-constrained edge devices with different latency, compute, and model-size requirements. Training separate models for each configuration is computationally prohibitive, as each variant must be trained independently from scratch. Additionally, conventional transformer-based models are inherently static at inference time, which means their parameters are fixed after training, limiting their ability to target heterogeneous systems. Recent elastic and nested methods address this by training multiple models as one, but most focus on a single component such as the feed-forward network (FFN) for size reduction or require a

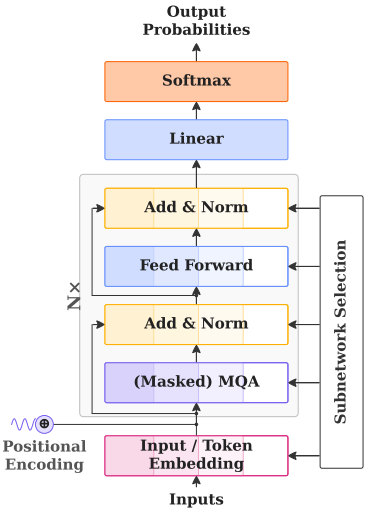


Figure 1: HydraLLM width slicing. The number of active heads determines the embedding dimension, the attention projection, and the feed-forward network width, producing nested subnetworks from a single parameter set.

[1]The source code is available at https://github.com/tuhh-ncps/hydrallm.

Proceedings of the Second BabyLM Workshop, pages 33–49
October 29, 2026 ©2026 Association for Computational Linguistics

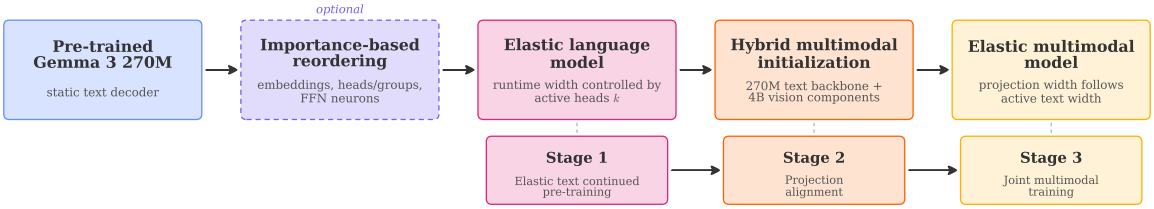


Figure 2: Pipeline overview. Starting from a static text checkpoint, such as Gemma 3 270M, the sliced axes can first be reordered to improve prefix quality. The resulting checkpoint is then trained as an elastic text model whose runtime width is controlled by the active number of query heads k. Finally, we combine the elastic text decoder with, for example, Gemma 3 4B vision components to form a multimodal model whose projection output follows the same active width.

multi-step extraction process of fixed model checkpoints (Haberer et al., 2024; Devvrit et al., 2024; Vaswani et al., 2017; Cai et al., 2020). In this paper, we introduce HydraLLM[1], a unified approach to adaptive slicing in LLMs and LVLMs by merging elastic architectural principles with efficient compression techniques while leveraging an already pre-trained static LLM.

We show that HydraViT-style (Haberer et al., 2024) end-to-end adaptive slicing transfers successfully to decoder-only LLMs, producing a single checkpoint that remains effective across several runtime width budgets. With 75% of heads active, HydraLLM improves Trivia-QA (Joshi et al., 2017) accuracy by 2.2 percentage points over a baseline configuration that has 17.25% more parameters. With 50% of heads active, the multimodal model reaches a CLIPScore of 66.9, surpassing the 65.3 score of the 7× larger Gemma 3 4B (Gemma Team et al., 2025) model. With only 25% of heads active, importance-based reordering improves post-training perplexity by 16.2%. This 25% configuration increases token-generation throughput by 75.29% compared to the full model. In summary, we contribute:

- **End-to-end slicing for decoder-only LLMs.** HydraLLM exposes a single head-count control that selects consistent prefixes of the embedding, attention, FFN, and output dimensions, producing nested decoder subnetworks from one checkpoint.
- **Unified training pipeline for elastic subnetworks.** We introduce a single training recipe to convert a static pre-trained LLM to an adaptively sliceable LLM while remaining competitive on text-only benchmarks.
- **Elastic multimodal projection for LVLMs.** We extend adaptive slicing to LVLMs by employing an elastic projection module that maps visual features into text embedding spaces of varying dimensionality.

2 Related Work

We discuss related work on nested representations and elastic architectures, transformer compression and submodel extraction, and projection-based multimodal models. We focus on how these approaches differ from HydraLLM's end-to-end runtime slicing of a pre-trained LLM and its extension to an elastic multimodal projection.

Kusupati et al. (2024) introduce multi-granularity, coarse-to-fine representations for embedding models. The same matryoshka nesting principle reappears in later work introducing elasticity within other transformer components. Cai et al. (2020) propose the Once-For-All (OFA) methodology, decoupling architecture search from training by progressively shrinking the convolutional neural network (CNN) kernel size, channel width, and model depth. OFA targets the search and extraction of specialized CNN architectures for different deployment constraints. In contrast, HydraLLM does not perform architecture search or progressive shrinking: it converts a pre-trained LLM into one elastic checkpoint whose embedding, attention, FFN, and output widths can be scaled jointly at runtime through the active head count. Devvrit et al. (2024) propose MatFormer, a natively elastic transformer that introduces a Matryoshka-like nested structure specifically *only* inside the FFN block. Haberer et al. (2024) target embeddings, attention heads, and FFN layers for their nested ViTs. Training employs stochastic subnetwork sampling by

drawing one head count k per step and updating only the corresponding subnetwork, which biases optimization toward earlier heads, implicitly ordering them by importance (Haberer et al., 2024). We apply the same training and architectural approach to LLMs and extend it to the vision domain. Cai et al. (2024) and Taghibakhshi et al. (2025) extend the nested approach in the attention and FFN components with routers in their FLEXTRON and Nemotron Elastic models. Both approaches are text-only. FLEXTRON slices only attention heads and FFN, and Nemotron Elastic targets an architecture distinct from our multi-query attention (MQA) (Shazeer, 2019) transformer. Muralidharan et al. (2024) design a static submodel extraction approach by applying iterative structured pruning on a pre-trained model, creating MINITRON. They obtain separate models that are not suitable for elastic inference, like ours.

ViTs (Dosovitskiy et al., 2021) adapt the transformer encoder to images by splitting an image into non-overlapping patches, flattening each patch, and projecting it into the model's embedding space to obtain patch embeddings. For multimodal models, Liu et al. (2023) connect a pre-trained CLIP (Radford et al., 2021) vision encoder to an open-source LLM via a projection layer that maps visual features into the LLM's text embedding space, producing so-called soft tokens. They merge the soft tokens with text tokens and process them jointly through self-attention. Deitke et al. (2024) build on these concepts with the Molmo family, while Marafioti et al. (2025) propose SmolVLM as a series of compact multimodal models. Gemma Team et al. (2025) release Gemma 3 with a tailored SigLIP (Zhai et al., 2023) vision encoder. We follow the training and architectural principles of these multimodal models and utilize insights from their training approaches. However, we employ the elastic LLM and multimodal projection.

3 HydraLLM: Architecture and Pipeline

This section introduces HydraLLM's design and the pipeline for converting a static pre-trained LLM into an elastic text and multimodal model. Specifically, we describe the text architecture and width-control in Section 3.1, the static-to-elastic conversion procedure including reordering and the optimization objective details in Section 3.2, and the multimodal assembly in Section 3.3. Our pipeline starts from a static pre-trained text checkpoint and converts it into an elastic decoder whose active width is controlled at runtime; see Figure 2. We then extend the same elasticity principle to the multimodal setting with an elastic projection module.

3.1 Runtime Width-Control

HydraLLM adapts HydraViT's (Haberer et al., 2024) head-stacking principle to decoder-only language models by using the number of active query heads k as a single runtime width-control parameter; see Figure 1. A subnetwork keeps the first k query heads together with the corresponding prefixes of the embedding channels, output projection, and FFN neurons. Each sliced dimension scales linearly with the number of heads k in relation to the total number of query heads H_{q}: the active embedding width is $d_{E,\mathrm{active}} = k/H_{\mathrm{q}} \times d_E$, with the FFN dimensions following the same proportion. This produces a strict nesting of subnetworks that share parameters across configurations. At inference, selecting k computes only the corresponding prefix, so one checkpoint serves all width budgets. We target MQA (Shazeer, 2019) in our design, in which all query heads share a single key-value head.

3.2 Static-to-Elastic Conversion

A static pre-trained checkpoint is not immediately suitable for prefix slicing. In an unmodified checkpoint, units of a component are usually not sorted by their importance. Since prefix slicing always retains the first units and discards later units, naïve slicing can remove important parameters and severely degrade smaller subnetworks. HydraLLM employs a two-stage procedure to convert static models into elastic ones. First, the pipeline reorders the checkpoint offline, moving important units toward the beginning of each sliced dimension so that prefix slicing retains the most informative parameters. Second, it optimizes the checkpoint using stochastic subnetwork training: during each training step, the pipeline samples a single k and updates only the corresponding active subnetwork. Since earlier units are active for more sampled values of k, they receive proportionally more gradient updates, reinforcing the importance ordering and adapting the shared parameters to all configured widths (Taghibakhshi et al., 2025; Muralidharan et al., 2024; Cai et al., 2024; Haberer et al., 2024; Devvrit et al., 2024).

3.2.1 Importance Scoring and Reordering

The reordering step occurs before any training and requires the definition of importance. We compare three importance measures: Activation magnitude (Muralidharan et al., 2024), variance-weighted consumer strength (VxC) adapted from Weinstein and Belinkov (2022), and CETT (Zhang et al., 2024), which scores units by their normalized contribution to the component output and therefore aligns well with prefix slicing. We estimate unit importance from activation statistics collected during a forward pass over a calibration text corpus. After computing the importance scores, we permute the corresponding tensors so that important units appear earlier along the sliceable dimensions: For embedding-width reordering, we apply one global permutation to the token embedding channels and propagate it consistently through the RMSNorm weights, the input columns of the Q/K/V projections, the output rows of the attention output projection, the input columns of the FFN gate and up projections, the output rows of the FFN down projection, the final normalization layer, and, if untied, the language-model head. For attention reordering, we permute the query heads and the corresponding attention-output input blocks. For FFN reordering, we permute FFN neurons by moving the rows of the gate and up projections and the corresponding columns of the down projection. This reordering step preserves the function of the full model.

3.2.2 Optimization Objective

After reordering, we train the model. The training objective we employ is a combination of autoregressive language modeling, logit regularization, and teacher supervision. The primary task loss, $\mathcal{L}_{\text{CE}}$, is standard next-token cross-entropy. We additionally add a z-loss term $\mathcal{L}_z$, weighted by λ_z (Chowdhery et al., 2022), and logit-based knowledge distillation (Hinton et al., 2015). Let z_t and z_s denote the teacher and student logits, respectively, and let $p_t^\tau = \text{softmax}(z_t/\tau)$ and $p_s^\tau = \text{softmax}(z_s/\tau)$ denote their temperature-scaled output distributions. The training objective is:

$$\begin{aligned} \mathcal{L}_{\text{KD}} &= \tau^2 \text{KL}(p_t^\tau \,\|\, p_s^\tau)\,, \\ \mathcal{L} &= \mathcal{L}_{\text{CE}} + \lambda_z \mathcal{L}_z + \lambda_{\text{KD}} \mathcal{L}_{\text{KD}} \end{aligned} \tag{1}$$

We keep the teacher model frozen throughout training.

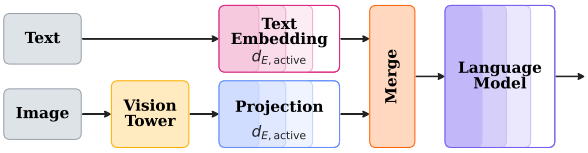


Figure 3: Elastic LVLM architecture. The projection output width follows the active text width.

3.3 Elastic Multimodal Assembly

We construct the multimodal variant of HydraLLM by combining the language decoder with a static pre-trained SigLIP-based vision encoder (Gemma Team et al., 2025; Zhai et al., 2023). The key difference from a conventional projection-based LVLM is that HydraLLM also makes the projection elastic. Let d_P denote the projection output width. We enforce $d_{P,\text{active}} = d_{E,\text{active}}$, so the same runtime parameter k controls both the active text decoder width and the projection output width; see Figure 3. This creates a single elasticity axis across modalities: selecting a smaller text subnetwork also selects a correspondingly smaller visual projection output.

4 Experiment Setup

We describe our training data and evaluation protocol in the following subsections 4.1 and 4.2, respectively.

4.1 Training Recipe

We base our experiments on Google Gemma 3 270M (Gemma Team et al., 2025), with $k \in \{1, 2, 3, 4\}$; see Appendix A.2. For text-only experiments, we use the frozen full-width Gemma 3 270M as the teacher model.

Most text experiments train on a subset of FineWeb-Edu 350BT (Lozhkov et al., 2024). We concatenate text samples with an `<eos>` separator and split the text into fixed-length token blocks. We employ non-uniform runtime width sampling of k, placing additional ($\times 3$) emphasis on the full-width model compared to the submodels; see Appendix A.5. In contrast to the packed text pretraining setup, the optional text fine-tuning stage uses text samples as they appear, without fully packing the sequence length; see Appendix C.1.

For multimodal training, we use the elastic HydraLLM Gemma 3 270m model combined with the vision encoder and projection module initialized from the Gemma 3 4B multimodal check-

Recipe	Perplexity ↓			
	H1	H2	H3	H4
no reordering	168.6M	1.1M	41.3k	49.0
mag-cett-cett	37.2k	2.43k	1.22k	49.0
mag-mag-mag	8.45M	378.9k	1.54k	49.0
vxc-vxc-vxc	38.0M	4.87M	60.4M	49.0

Table 1: Perplexity of **reordered** checkpoints measured on samples of plays by William Shakespeare. Magnitude for embeddings combined with CETT for heads and FFN yields the strongest subnetwork perplexities, while VxC consistently hurts.

Recipe	Perplexity ↓			
	H1	H2	H3	H4
no reordering	123.9	77.4	56.7	45.2
mag-cett-cett	106.6	71.6	54.4	44.3
mag-mag-mag	110.1	71.9	54.4	44.8
vxc-vxc-vxc	118.1	71.7	55.0	44.1

Table 2: Perplexity of **reordered and trained** (20k steps) checkpoints measured on samples of plays by William Shakespeare. Magnitude for embeddings combined with CETT for heads and FFN yields the strongest subnetwork perplexities.

point. The teacher for this training is the static and frozen Gemma 3 4B model. We employ uniform head sampling and separate learning rates for the text backbone, the projection module, and, optionally, the vision encoder. We configure separate peak learning rates and number of warmup steps for the text backbone, projection and vision encoder; see Appendix A.5. Multimodal training samples from PixMoCap, the COCO Recaptioned dataset, LLaVA Pretrain 558K (Liu, 2023), PixMo Ask Model Anything, PixMo Cap-QA, CoSyn-400K, and text-only data from Wikipedia (Wikimedia Foundation, 2023) and FineWeb-Edu. See Appendix A.3 for detailed composition. Projection alignment utilizes 128,000 captioning samples from the aforementioned captioning datasets.

4.2 Metrics and Evaluation Datasets

We employ a mix of metrics and benchmarks to evaluate models and derive conclusions. For text evaluation, we report ARC-Easy (Clark et al., 2018) and TriviaQA (Joshi et al., 2017). We evaluate ARC-Easy with both raw accuracy and answer-length-normalized accuracy. We evaluate TriviaQA with five-shot prompting and exact-match scoring. We measure perplexity (Jurafsky and Martin, 2026) on samples of plays by William Shakespeare (Balbay, 2024) as a general-purpose indicator of language modeling quality.

For multimodal evaluation, we report CLIPScore (Hessel et al., 2021; Radford et al., 2021) on COCO validation images and multiply the score by 100 for readability. We additionally use qualitative captioning examples to inspect generation quality across different head configurations. The standard error on text- and CLIPScore-benchmark results denotes standard errors over evaluation examples.

We measure throughput with `llama-bench` from `llama.cpp` (Gerganov and ggml-org contributors, 2025), with prompt-processing of 2048 tokens and autoregressive generation of 512 tokens on an NVIDIA RTX PRO 4000. This setup allows measuring prompt-processing and token generation speed improvements.

5 Evaluation

We evaluate the static-to-elastic conversion pipeline in terms of reordering in Section 5.1, compare HydraLLM with MatFormer-style (Devvrit et al., 2024) FFN slicing in terms of accuracy and throughput in Section 5.2, and assess the quality of multimodal extensions in Section 5.3.

5.1 Effect of Importance Reordering

To determine the effect of reordering, we measure perplexity for each head configuration and compare it to the unmodified pre-trained model in Table 1. We denote the k-head configurations as H1-H4. Reordering clearly improves the perplexity of submodels when utilizing the magnitude metric for embeddings combined with CETT for heads and FFN, or the magnitude metric for all components. Employing only VxC provides an improvement solely for the H1 configuration.

After reordering, we train the reordered models for 20,000 steps ($\sim$1B tokens) and measure their perplexities again. After training, perplexity improves for all heads, regardless of the starting point; see Table 2. However, the CETT importance metric for heads and FFN with magnitude importance for the embeddings provides the best results for the submodels. VxC slightly edges out CETT on the four-head configuration. For H4 of the not reordered model, continued training reduces perplexity from 49.0 to 45.2. However, this improvement does not consistently transfer to downstream benchmarks. Figure 4 and Figure 5 from the next Section 5.2 also show the ARC-Easy and TriviaQA

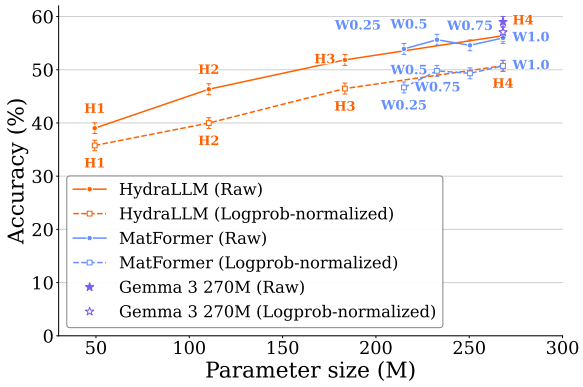


Figure 4: ARC-Easy accuracy as a function of active parameter size for HydraLLM and MatFormer configurations. HydraLLM covers a wider active parameter range than MatFormer while offering competitive accuracies.

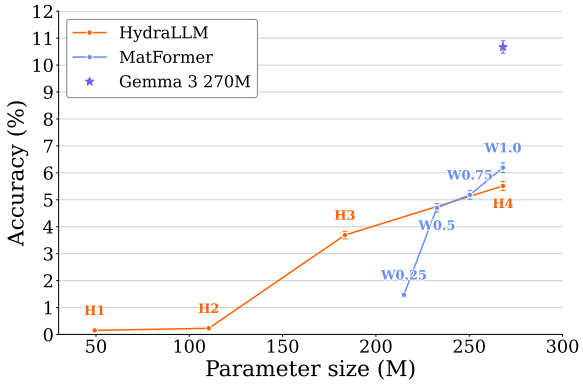


Figure 5: TriviaQA accuracy as a function of active parameter size for HydraLLM and MatFormer configurations. HydraLLM is stronger at H3 clearly outperforming the similarly sized MatFormer W0.25, while MatFormer surpasses HydraLLM at the full configuration.

accuracy of the pre-trained and the no-reordering model trained for 20,000 steps. The benchmarks actually show a lower accuracy compared to the static baseline. We provide more benchmark results in Appendix B.2.4.

5.2 HydraLLM vs. MatFormer

We compare HydraLLM to MatFormer-style FFN slicing, which we reimplement using the same Gemma 3 270M checkpoint, in terms of accuracy and runtime throughput. The hyperparameters for both models are the same, apart from the width selection mechanism. In Figures 4 and 5, we label the MatFormer FFN configurations as W0.25-W1.0, corresponding to the fraction of FFN neurons active. Because MatFormer only alters the FFN component, the smallest MatFormer configuration, W0.25, still exceeds the second-largest HydraLLM configuration, H3, by 17.25% in parameter count. The raw ARC-Easy accuracy of W0.25 is slightly higher than that of H3, though not statistically significant. The log-probability-normalized accuracies are virtually the same. The HydraLLM approach has a clear advantage with similar accuracies, but a wider parameter size coverage. The ten-shot prompting HellaSwag (Zellers et al., 2019) results lead to the same conclusion, therefore we provide them in Appendix B.1 only.

Config	HydraLLM		MatFormer	
	Prefill	Decode	Prefill	Decode
H1 / W0.25	55,891.3	1,057.0	51,549.1	669.6
H2 / W0.50	52,851.7	785.9	51,353.8	648.5
H3 / W0.75	51,165.4	594.2	49,945.2	623.4
H4 / W1.00	50,487.9	603.0	50,509.8	602.3

Table 3: Throughput in tokens per second for HydraLLM and MatFormer variants. Prompt-processing of 2048 tokens. Decode 512 tokens. Prompt-processing throughput gains are modest for both methods, while token-generation scaling is much larger for HydraLLM than for MatFormer.

The results on TriviaQA differ. While the W0.5 and W0.75 MatFormer configurations interpolate smoothly between the H4 and H3 HydraLLM configurations in Figure 5, H3 achieves 2.21 percentage points ($\times$2.51) higher accuracy than W0.25, although they are similar in size. Conversely, W1.0 outperforms H4 by 0.68 percentage points. The HydraLLM approach has a broader parameter spread, starting from 49.34M parameters, compared to MatFormer, starting from 215.01M parameters, while achieving competitive results with similar submodel sizes. However, submodels H1 and H2 only achieve close to 0% accuracy on this benchmark.

The prompt-processing improvements from smaller model configurations displayed in Table 3 are very modest. The improvement from the H4 to H1 configuration ranges from 9.78% to 11.61%, and the improvement from W1.0 to W0.25 ranges from 1.03% to 3.1%. The Gemma 3 270M model comprises $\sim$170 million embedding parameters and $\sim$100 million transformer block parameters. Since the embeddings are mainly a lookup operation, they do not contribute much to throughput, but they account for a large portion of the parameter savings in the HydraLLM approach (Gemma Team et al., 2025; Vaswani et al., 2017).

For token generation, MatFormer shows consistent speed-ups with shrinking parameter count. HydraLLM produces fewer tokens per second with the H3 configuration compared to H4, which we hypothesize stems from kernel-shape optimizations in llama.cpp. The improvement in token generation speed from the largest to the smallest configuration is 75.29% for HydraLLM and 11.17% for MatFormer. The main takeaway is that prompt-processing gains are minor, while token-generation scaling is much greater, but does not scale one-to-one with parameter count. The reported throughput gains therefore stem mostly from smaller embeddings and query, output, and FFN projections, because MQA (Shazeer, 2019) prohibits key-value cache savings from width reduction.

5.3 Multimodal Pipeline Results

The first stage in the multimodal training pipeline is aligning the projection with the text embedding space. The projection-only block of Table 4 shows that the untrained projection already produces a non-trivial CLIPScore (Hessel et al., 2021; Radford et al., 2021) across all head configurations, indicating that it inherits useful structure from the Gemma 3 4B vision tower. Training the projection in isolation yields only modest improvements over the untrained baseline. During training, we observe a plateau in the loss after 300 to 350 steps, depending on the head configuration; see more details in Appendix C.2. The rapid plateau indicates a lack of capacity in the projection layer, caused by the much smaller projection due to the lower text embedding dimension of the Gemma 3 270M model compared to Gemma 3 4B. The projection has 25% of the parameters of the Gemma 3 4B model and therefore possesses much less representational power (Gemma Team et al., 2025).

Joint training of the projection with the text model substantially improves the CLIPScore; see the projection-plus-text-model block of Table 4. The models trained from an already projection-pre-aligned checkpoint perform best across all head configurations, especially at H2, implying that aligning the projection before joint training of the projection and text model is beneficial. When training all model parameters together, the all-parameters block shows that results are similar across different starting points, indicating that joint optimization can compensate for an inferior starting projection.

Scaling the joint projection and text model training, as well as the full-parameter training, from 2,500 to 16,000 steps further closes the gap to the Gemma 3 4B reference; see the bottom block of Table 4. The full-parameter trainable run matches or exceeds Gemma 3 4B at H2, H3, and H4 and reaches 62.6 at H1. The results suggest that, on CLIPScore, the elastic model matches the vision-language capabilities of Gemma 3 4B across most subnetworks. The cause for this lies in the response style of the Gemma 3 4B model: it often responds to captioning prompts with question-answer-style continuations rather than direct descriptive sentences, which reduces its CLIP similarity score. CLIPScore is therefore sensitive to response style and does not, by itself, guarantee stable caption quality (Hessel et al., 2021).

Given the strong CLIPScore results of HydraLLM compared to Gemma 3 4B, we evaluate our model with MMMU (Yue et al., 2024). Our models consistently achieve random accuracy of around 25%, whereas the Gemma 3 4B model reaches about 39%. We illustrate the qualitative

Table 4: CLIPScore ($\times 100$) across head configurations for each multimodal training recipe. Bold marks the best score per head configuration. *non-elastic*: starting from pre-trained Gemma 3 270m and only head configuration $k{=}4$ is trained; *elastic*: starting from elastic text model and all head configurations are trained stochastically; *fine-tune*: an additional text-only fine-tuning step is applied before multimodal training; *pre-aligned*: the projection is pre-aligned on captioning data before joint training; *Gemma 3 4B*: the original frozen Gemma 3 4B multimodal model provided as a baseline.

Training Recipe	**H1**	**H2**	**H3**	**H4**
No training				
elastic	32.4	31.5	32.0	32.7
Gemma 3 4B	–	–	–	65.3
Projection only (500 steps)				
non-elastic	–	–	–	28.9
elastic	**38.2**	31.4	**31.1**	**33.5**
elastic, fine-tune	34.8	**34.2**	31.0	32.1
Projection + text model (2,500 steps)				
elastic	46.3	53.9	58.5	58.8
pre-aligned, non-elastic	–	–	–	62.5
pre-aligned	**47.4**	56.9	**61.3**	61.0
pre-aligned, fine-tune	47.1	**57.4**	59.1	**61.1**
All parameters (2,500 steps)				
elastic	50.2	59.4	**63.1**	**63.9**
pre-aligned	51.1	**60.7**	61.3	62.5
pre-aligned, fine-tune	**51.2**	60.4	61.5	63.6
From text fine-tune + pre-aligned (16,000 steps)				
Projection + text model	59.1	65.4	65.1	68.0
All parameters	**62.6**	**66.9**	**68.6**	**69.6**



<start_of_image>
"Briefly describe the image!"

k = 4 The image depicts a blue Tesla model driving down an urban street. The car is positioned on its side, facing left and slightly to one direction as it moves along the road with trees in front of it. It has black tires that are visible at both ends of the vehicle's body. A white roof line runs...

k = 3 The image depicts a blue Tesla car parked on an asphalt street in front of two houses. The house has white siding and is surrounded by trees, creating a picturesque scene with its distinctive brown roofline visible through it. In addition to this residential setting, there's also another structure that appears similar but not...

k = 2 The photograph shows a blue electric car parked on an urban street. The background is predominantly white, with some green foliage and trees visible in the foreground. There's also text that reads "electric house" at the top of this photo which appears to be part of another picture or possibly from other sources like Google Maps...

k = 1 a black car parked on a street with trees and an old house in front of it, looking at its surroundings. The scene is set against a backdrop that features white buildings along the road; there's also some green grass visible near the driveway behind them as well. A small tree stands beside this building to the...

Figure 6: Qualitative captioning example. The elastic LVLM is prompted to describe an image of a car across the four head configurations ($k = 1$ to $k = 4$). Hallucinated or incorrect statements are highlighted in orange, illustrating the qualitative gap that persists alongside the encouraging CLIPScore trends.

behavior with the captioning example in Figure 6. We prompt the model to describe an image of a car. Across all head configurations, the captions include hallucinated or incorrect statements about the scene. Subject resolution, spatial relations, and object identity are unreliable: the four-head configuration hallucinates motion and orientation, the three-head and two-head configurations misidentify surrounding objects, and the single-head configuration misidentifies the car's color and introduces incorrect scene elements. Hallucinations, weak grounding, and unstable object descriptions therefore persist across all head configurations of HydraLLM. The elastic projection establishes feasibility, but the vision-to-language interface remains a suspected bottleneck and does not yet yield robust caption quality.

6 Conclusion

We introduce HydraLLM, an MQA-based (Shazeer, 2019) elastic architecture that employs stochastic head stacking and end-to-end width slicing. A single runtime parameter k controls the active width of the decoder end-to-end, from token embeddings through attention and FFN projections to the output head, producing nested subnetworks from a single parameter set. We extended the elasticity axis to LVLMs through an elastic projection module whose output width follows the active text width.

On text, magnitude-based reordering of embeddings combined with CETT-based (Zhang et al., 2024) reordering of heads and FFN units yields the strongest subnetworks, reducing perplexity by up to a factor of approximately 4,500 compared to no reordering. However, the effect is reduced to at most 14% after training. Knowledge distillation (Hinton et al., 2015) is crucial during training and improves TriviaQA exact-match accuracy at 75% of active heads by 97.3%. HydraLLM covers a substantially wider parameter range at competitive accuracy compared to MatFormer-style (Devvrit et al., 2024) FFN slicing. It achieves larger token-generation speed-ups of up to 75.3%, but only modest prompt-processing improvements.

On the multimodal side, scaled training matches or exceeds the Gemma 3 4B CLIPScore (Hessel et al., 2021; Radford et al., 2021) of 65.3 by 2.5%, 5.1% and 6.6% with two, three and four active heads respectively, confirming that the elastic text backbone transfers to the vision-language setting. However, the model remains near random accuracy on MMMU, and qualitative analysis reveals hallucinations and weak object grounding. These limitations point to clear avenues for future research: (i) replacing the constrained small single-layer projector with a larger or a multi-layer FFN projection, which would better align vision features with the elastic text backbone; (ii) longer and more diverse multimodal training schedules to improve grounding beyond captioning tasks; and (iii) combining HydraLLM's end-to-end slicing with MatFormer-style per-layer granularity for finer accuracy-compute control. We leave these directions for future work.

Acknowledgments

We thank our anonymous reviewers for their insights and suggestions for improvement. This work has received funding from the Federal Ministry of Research, Technology and Space under the REAKT ISA project (grant no. 03DPC1415B) and from the Federal Ministry for Economic Affairs and Energy under the CAPTN X-FERRY project (grant no. 03SX612A).

References

Allen Institute for AI. 2024a. PixMo-AskModelAnything. https://huggingface.co/datasets/allenai/pixmo-ask-model-anything. Hugging Face dataset, 161,737 rows, ODC-BY-1.0. Accessed: 2026-03-11.

Allen Institute for AI. 2024b. PixMo-Cap. https://huggingface.co/datasets/allenai/pixmo-cap. Hugging Face dataset. Image-to-text dataset, 717,042 rows. Accessed: 2026-03-11.

Allen Institute for AI. 2024c. PixMo-CapQA. https://huggingface.co/datasets/allenai/pixmo-cap-qa. Hugging Face dataset. Visual Question Answering dataset, 271,714 rows. Accessed: 2026-03-11.

Allen Institute for AI. 2025. CoSyn-400K. https://huggingface.co/datasets/allenai/CoSyn-400K. Hugging Face dataset. Visual Question Answering dataset, 408,227 rows. Accessed: 2026-03-11.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report. *Preprint*, arXiv:2511.21631.

Alper Balbay. 2024. Calibration split of the tiny shakespeare dataset. GitHub repository file. Commit 5a58873, accessed 21 July 2026.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 2020. Once-for-all: Train one network and specialize it for efficient deployment. *Preprint*, arXiv:1908.09791.

Ruisi Cai, Saurav Muralidharan, Greg Heinrich, Hongxu Yin, Zhangyang Wang, Jan Kautz, and Pavlo Molchanov. 2024. Flextron: Many-in-one flexible large language model. *Preprint*, arXiv:2406.10260.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2022. Palm: Scaling language modeling with pathways. *Preprint*, arXiv:2204.02311.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *Preprint*, arXiv:2307.08691.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, and 31 others. 2024. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. *Preprint*, arXiv:2409.17146.

Devvrit, Sneha Kudugunta, Aditya Kusupati, Tim Dettmers, Kaifeng Chen, Inderjit Dhillon, Yulia Tsvetkov, Hannaneh Hajishirzi, Sham Kakade, Ali Farhadi, and Prateek Jain. 2024. Matformer: Nested transformer for elastic inference. *Preprint*, arXiv:2310.07707.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *Preprint*, arXiv:2010.11929.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Georgi Gerganov and ggml-org contributors. 2025. llama.cpp: LLM inference in C/C++. Build: 4df6e859e (7349).

Google. 2024. Gemini: Your personal ai assistant. https://gemini.google/assistant/. Accessed: 2026-02-03.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate.

Janek Haberer, Ali Hojjat, and Olaf Landsiedel. 2024. Hydravit: Stacking heads for a scalable vit. *Preprint*, arXiv:2409.17978.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. 2023. Lighteval: A lightweight framework for llm evaluation.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. Clipscore: A reference-free evaluation metric for image captioning. *CoRR*, abs/2104.08718.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *Preprint*, arXiv:1705.03551.

Daniel Jurafsky and James H. Martin. 2026. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*, 3rd edition. Online manuscript released January 6, 2026.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. 2024. Matryoshka representation learning. *Preprint*, arXiv:2205.13147.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. Microsoft coco: Common objects in context. *Preprint*, arXiv:1405.0312.

Haotian Liu. 2023. LLaVA-Pretrain. https://huggingface.co/datasets/liuhaotian/LLaVA-Pretrain. Hugging Face dataset; LLaVA Visual Instruct Pretrain dataset created in May 2023. Accessed: 2026-03-11.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Preprint*, arXiv:2304.08485.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and 1 others. 2024. FineWeb-Edu: The Finest Collection of Educational Web Data on Hugging Face.

Andrés Marafioti, Orr Zohar, Miquel Farré, Merve Noyan, Elie Bakouch, Pedro Cuenca, Cyril Zakka, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Vaibhav Srivastav, Joshua Lochner, Hugo Larcher, Mathieu Morlon, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. 2025. Smolvlm: Redefining small and efficient multimodal models. *Preprint*, arXiv:2504.05299.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation. *Preprint*, arXiv:2407.14679.

OpenAI. 2024. Chatgpt overview. https://chatgpt.com/overview?openaicom_referred=true. Accessed: 2026-02-28.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. *Preprint*, arXiv:1912.01703.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *Preprint*, arXiv:2103.00020.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *Preprint*, arXiv:1911.02150.

Ali Taghibakhshi, Sharath Turuvekere Sreenivas, Saurav Muralidharan, Ruisi Cai, Marcin Chochowski, Ameya Sunil Mahabaleshwarkar, Yoshi Suhara, Oluwatobi Olabiyi, Daniel Korzekwa, Mostofa Patwary, Mohammad Shoeybi, Jan Kautz, Bryan Catanzaro, Ashwath Aithal, Nima Tajbakhsh, and Pavlo Molchanov. 2025. Nemotron elastic: Towards efficient many-in-one reasoning llms. *Preprint*, arXiv:2511.16664.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Berry Weinstein and Yonatan Belinkov. 2022. Variance pruning: Pruning language models via temporal neuron variance.

Wikimedia Foundation. 2023. Wikimedia Downloads. Referenced via the Hugging Face wikimedia/wikipedia dataset card. Accessed: 2026-03-11.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, and 3 others. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *Preprint*, arXiv:2311.16502.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. *Preprint*, arXiv:2303.15343.

Zhengyan Zhang, Yixin Song, Guanghui Yu, Xu Han, Yankai Lin, Chaojun Xiao, Chenyang Song, Zhiyuan Liu, Zeyu Mi, and Maosong Sun. 2024. Relu^2 wins: Discovering efficient activation functions for sparse llms. *Preprint*, arXiv:2402.03804.

Table 5: Key Python package versions used in the implementation.

Package	Version
torch	2.8.0
torchvision	0.23.0
transformers	4.56.1
accelerate	1.10.1
datasets	4.2.0
lighteval	0.11.0

A Reproducibility Details

This section provides implementation and hardware details, runtime configurations, multimodal dataset composition, CLIPScore evaluation details, and training hyperparameters.

A.1 Implementation and Hardware

We build our implementation around PyTorch (Paszke et al., 2019) and Hugging Face Transformers (Wolf et al., 2020). Accelerate (Gugger et al., 2022) and DeepSpeed (Rasley et al., 2020) support distributed training, and LightEval (Habib et al., 2023) integrates the text benchmarks. CLIPScore follows Hessel et al. (2021). When available, we use optional acceleration libraries such as FlashAttention 2 (Dao, 2023), while ensuring a transparent fallback to the standard PyTorch and Transformers backends. Table 5 lists exact package versions.

Text-only experiments run on two NVIDIA RTX PRO 4000 GPUs with at least 24 GB of VRAM each. If we disable the teacher model, text-only training fits on a single GPU. Multimodal training is more resource-demanding: the unified multimodal pipeline requires at least 48 GB of VRAM, and joint training of all parameters requires roughly 60 GB. The multimodal experiments therefore run on two NVIDIA Tesla L40 GPUs or two NVIDIA A100 80 GB GPUs. Throughput measurements use `llama.cpp` build `4df6e859e` (7349) (Gerganov and ggml-org contributors, 2025).

A.2 Runtime Configurations

Table 6 lists the active component dimensions for each head configuration $k \in \{1, 2, 3, 4\}$.

A.3 Multimodal Dataset Composition

Tables 7 and 8 report the dataset mixtures for the main 2,500-step multimodal runs and the scaled 16,000-step runs, respectively. The scaled setting preserves the same broad mixture of captioning, question answering, code generation, and text-only data, but increases the number of samples. The mixtures comprise PixMo-Cap (Allen Institute for AI, 2024b), our recaptioned version of COCO (Lin et al., 2015), LLaVA Pretrain (Liu, 2023), PixMo Ask Model Anything (Allen Institute for AI, 2024a), PixMo Cap-QA (Allen Institute for AI, 2024c), the VQA and code subsets of CoSyn-400K (Allen Institute for AI, 2025), and text-only samples from Wikipedia (Wikimedia Foundation, 2023) and FineWeb-Edu (Lozhkov et al., 2024).

Projection alignment trains on a separate mixture of 64,000 samples from COCO Recaptioned and 64,000 samples from PixMo-Cap, for a total of 128,000 samples. We generate COCO Recaptioned locally by recaptioning COCO training images with Qwen 3 VL 4B Instruct (Bai et al., 2025) utilizing the fixed descriptive instruction "Describe the image in 3-5 sentences!". The resulting captions are denser and more uniform than the original short COCO annotations.

	H1	H2	H3	H4
H_{kv}	1	1	1	1
query heads k	1	2	3	4
$d_{E,\text{active}}$	160	320	480	640
$d_{\text{ff},\text{active}}$	512	1024	1536	2048
$d_{Q,\text{active}}$	256	512	768	1024

Table 6: HydraLLM's Gemma 3 270M runtime configurations and active component dimensions. $d_{E,\text{active}}$ is the active embedding width, $d_{\text{ff},\text{active}}$ the active FFN neuron count, and $d_{Q,\text{active}}$ the total query projection dimension. MatFormer's W0.25-W1.0 have the same $d_{\text{ff},\text{active}}$ as H1-H4, but otherwise keep the dimensions of H4.

Table 7: Composition of the multimodal dataset of the main multimodal experiments.

Dataset	Num Samples
Captioning & Pretraining	
PixMo-Cap	50,000
COCO Recaptioned	12,000
LLaVA Pretrain	50,000
Question Answering	
PixMo Ask Model Anything	30,000
PixMo Cap-QA	30,000
CoSyn 400K VQA	100,000
Code Generation	
CoSyn 400K Code	3,200
Text Only	
Wikipedia EN	14,800
FineWeb-Edu 10BT	30,000
Total	320,000

Table 8: Composition of the multimodal dataset of the scaled 16,000-step multimodal training runs.

Dataset	Num Samples
Captioning & Pretraining	
PixMo-Cap	500,000
COCO Recaptioned	100,000
LLaVA Pretrain	500,000
Question Answering	
PixMo Ask Model Anything	125,000
PixMo Cap-QA	125,000
CoSyn 400K VQA	1,000,000
Code Generation	
CoSyn 400K Code	50,000
Text Only	
Wikipedia EN	200,000
FineWeb-Edu 10BT	200,000
Total	2,800,000

A.4 CLIPScore Evaluation

The CLIPScore describes how to measure the caption quality of a model, but it does not define the underlying dataset or the CLIP-based (Radford et al., 2021) model (Hessel et al., 2021). We use the entire COCO validation set to calculate the similarity score with clip-vit-large-patch14 (Radford et al., 2021).

A.5 Training Hyperparameters

Table 9 summarizes the hyperparameters for the text-only and default multimodal training runs. The scaled multimodal runs retain the settings of the corresponding 2,500-step runs, but train for 16,000 steps on the larger dataset mixture in Table 8, optionally train the vision encoder at a learning rate of 6×10^{-6} and reduce the text model warmup from 125 to 75 steps, while, if enabled, adding 75 warmup steps for the vision encoder. Compared with text-only training, we employ a lower text-backbone learning rate, a separate projection learning rate, a larger effective batch size, a larger knowledge-distillation weight when distilling from Gemma 3 4B, and uniform head sampling instead of the non-uniform text-only sampling weights for both multimodal runs. The sampling weight array assigns a weight to each head for random sampling.

B Additional Text Experiments

This section presents the text training ablations, followed by the importance metrics and additional reordering results.

B.1 Training Recipe Ablation

Table 10 reports the full text-training ablation across ARC-Easy, HellaSwag, and TriviaQA. We evaluate ARC-Easy and HellaSwag with both raw and answer-length-normalized accuracy with zero- and ten-shot prompting respectively, and TriviaQA with five-shot exact-match accuracy. Relative to the default recipe, the ablations disable the teacher, use uniform head sampling, restrict training to head configurations $\{2, 3, 4\}$, increase the knowledge-distillation weight to 0.3, use the FineWeb-Edu 10BT subset, apply an EMA decay of 0.999, increase the effective batch size to 128, or reduce the learning rate to 2×10^{-5}. The long-run variant instead extends training to one million steps and uses a larger FineWeb-Edu 350BT subset.

B.2 Importance Reordering

This section describes the three importance metrics we implement to reorder embedding channels, attention heads, and FFN neurons before stochastic subnetwork training. We compute importance scores from activation statistics collected during a forward pass over a calibration text corpus (Balbay, 2024) using forward hooks (Taghibakhshi et al., 2025). Each metric ranks units independently for each component. We then apply the resulting orders to the embedding dimension, the query-head axis, and the FFN intermediate axis, as described in Section 3.2.1.

B.2.1 Magnitude-Based Importance

The simplest criterion is activation magnitude (Taghibakhshi et al., 2025; Muralidharan et al., 2024). It assumes that units with consistently large activations contribute more strongly than units with small activations.

For FFN neurons, let $g^{(\ell)}_{b,t,j}$ denote the j-th channel of the gate-projection output in layer ℓ, for batch element b and sequence position t. The magnitude score is:

$$s^{\text{mag}}_{j,\ell} = \sum_{b,t} |g^{(\ell)}_{b,t,j}| \tag{2}$$

For attention heads, we score the query-side projection slice. If $q^{(\ell)}_{b,t,h} \in \mathbb{R}^{d_h}$ denotes the query activation of head h, then:

$$s^{\text{mag}}_{h,\ell} = \sum_{b,t} \|q^{(\ell)}_{b,t,h}\|_2 \tag{3}$$

For embedding dimensions, we compute magnitude from the normalized hidden states at the output of the pre-layer normalization. Let $x^{(\ell)}_{b,t,c}$ denote

Table 9: Training hyperparameters for the text-only and default multimodal runs. The full multimodal run where all components are trained differs only in vision encoder learning rate (6×10^{-6}) and warmup steps (text: 75, vision: 75). Where not listed, the multimodal configuration inherits the text default.

Hyperparameter	**Text**	**Multimodal**
Sequence length	3072	3072
Learning rate (text backbone)	3×10^{-4}	2×10^{-5}
Learning rate (projection)	–	2×10^{-4}
LR schedule	cosine	cosine
Weight decay	0.1	0.1
Optimizer	AdamW	AdamW
Adam betas	[0.9, 0.999]	[0.9, 0.999]
Adam epsilon	1×10^{-8}	1×10^{-8}
Warmup ratio (text)	0.01	–
Warmup steps (text model)	–	125
Warmup steps (projection)	–	25
Training steps	20,000	2,500
Effective batch size	16	128
Gradient clipping (norm)	1.0	1.0
Head selection	[1, 2, 3, 4]	[1, 2, 3, 4]
Sampling weights	[1, 1, 1, 3] (non-uniform)	[1,1,1,1] (uniform)
Loss	CE + z-loss + KD	CE + z-loss + KD
z-loss alpha	1×10^{-5}	1×10^{-5}
KD weight	0.1	0.2
KD temperature	2	2
Teacher model	Gemma 3 270M	Gemma 3 4B
Precision	bf16	bf16
Random seed	42	42

embedding channel c after normalization in layer ℓ. The layer-local score is:

$$s^{\text{mag}}_{c,\ell} = \sum_{b,t} |x^{(\ell)}_{b,t,c}| \tag{4}$$

Because we consistently slice embedding dimensions across the entire decoder, we sum layer-local scores into a single global ranking:

$$s^{\text{mag}}_{c} = \sum_{\ell} s^{\text{mag}}_{c,\ell} \tag{5}$$

Magnitude scoring is simple and inexpensive, but it does not account for how strongly downstream weights consume a unit's output.

B.2.2 Variance × Consumer Norms

The second metric, VxC, combines activation variance (Weinstein and Belinkov, 2022) with downstream consumer strength. The intuition is that a unit should rank highly if it varies across inputs and if strong outgoing weights amplify its output. Given activations z_i, the empirical variance is $\text{Var}(z_i) = \mathbb{E}[z_i^2] - \mathbb{E}[z_i]^2$, where the expectation is taken over the calibration data.

For FFN neurons, VxC multiplies the gate-projection activation variance by the squared norm of the corresponding down-projection column:

$$s^{\text{vxc}}_{j,\ell} = \text{Var}_{b,t}\left(g^{(\ell)}_{b,t,j}\right) \cdot \|W_{\text{down},\ell}[:, j]\|_2^2 \tag{6}$$

For attention heads, we first sum the channel-wise variance within each query head:

$$v_{h,\ell} = \sum_{d=1}^{d_h} \text{Var}_{b,t}\left(q^{(\ell)}_{b,t,h,d}\right) \tag{7}$$

Then we weight this term by the squared Frobenius norm of the corresponding output-projection slice $W^O_{\ell,h}$:

$$s^{\text{vxc}}_{h,\ell} = v_{h,\ell} \cdot \|W^O_{\ell,h}\|_F^2 \tag{8}$$

Several projections inside a transformer block consume the embedding channels. Let $\mathcal{C}_\ell = \{W^Q_\ell, W^K_\ell, W^V_\ell, W^{\text{gate}}_\ell, W^{\text{up}}_\ell\}$ denote these consumers. The per-layer embedding score is:

$$s^{\text{vxc}}_{c,\ell} = \text{Var}_{b,t}\left(x^{(\ell)}_{b,t,c}\right) \cdot \sum_{W \in \mathcal{C}_\ell} \|W[:, c]\|_2^2 \tag{9}$$

To obtain one global ranking, we sum the embedding scores across layers, just like we do with magnitude scoring. In comparison to magnitude, VxC directly captures a unit's role in the computation graph. Units that fluctuate significantly but have weak consumption receive lower rankings than those whose variation is amplified by large outgoing weights.

Table 10: Training recipe ablation across text benchmarks after conversion of the static Gemma 3 270M model, with MatFormer as a width-pruning comparison. Values are accuracies in percent, with standard error, for the four runtime head configurations H1–H4 and MatFormer widths W0.25–W1.0. ARC-Easy Norm is answer-length-normalized accuracy. TriviaQA is five-shot exact-match accuracy.

Variant	H1/W0.25	H2/W0.5	H3/W0.75	H4/W1.0
ARC-Easy (Raw)				
MatFormer	53.9 ± 1.02	55.6 ± 1.02	54.6 ± 1.02	56.0 ± 1.02
default	39.0 ± 1.00	46.3 ± 1.02	51.9 ± 1.03	56.4 ± 1.02
1M steps	43.6 ± 1.02	51.1 ± 1.03	56.9 ± 1.02	60.3 ± 1.00
no teacher	37.2 ± 1.00	46.2 ± 1.03	51.9 ± 1.02	56.1 ± 1.02
uniform sampling	38.8 ± 0.99	48.2 ± 1.02	52.2 ± 1.02	55.4 ± 1.02
heads 2, 3, 4	26.8 ± 0.91	46.2 ± 1.02	52.3 ± 1.03	57.8 ± 1.01
KD=0.3	37.9 ± 1.00	46.3 ± 1.02	52.9 ± 1.02	58.8 ± 1.01
FW-Edu 10BT	38.3 ± 1.00	45.7 ± 1.02	52.6 ± 1.02	56.7 ± 1.02
EMA=0.999	26.8 ± 0.91	28.3 ± 0.92	29.3 ± 0.93	54.2 ± 1.02
batch size=128	31.5 ± 0.95	39.2 ± 1.00	47.3 ± 1.02	57.3 ± 1.02
lower LR=2e-5	30.4 ± 0.94	34.4 ± 0.97	42.1 ± 1.01	57.4 ± 1.01
ARC-Easy (Norm)				
MatFormer	46.7 ± 1.02	49.8 ± 1.03	49.3 ± 1.03	50.7 ± 1.03
default	35.8 ± 0.98	40.0 ± 1.01	46.5 ± 1.02	50.8 ± 1.03
1M steps	38.6 ± 1.00	44.5 ± 1.02	49.5 ± 1.03	52.7 ± 1.02
no teacher	34.4 ± 0.99	40.5 ± 1.01	46.5 ± 1.02	49.7 ± 1.03
uniform sampling	36.3 ± 0.97	41.5 ± 1.01	46.8 ± 1.02	49.0 ± 1.03
heads 2, 3, 4	28.5 ± 0.92	41.2 ± 1.01	47.0 ± 1.02	51.0 ± 1.03
KD=0.3	34.7 ± 0.98	40.2 ± 1.01	48.4 ± 1.03	51.9 ± 1.03
FW-Edu 10BT	36.1 ± 0.99	39.9 ± 1.00	48.2 ± 1.03	51.8 ± 1.03
EMA=0.999	27.6 ± 0.92	27.7 ± 0.92	28.8 ± 0.93	51.4 ± 1.03
batch size=128	30.9 ± 0.95	34.8 ± 0.98	42.2 ± 1.01	50.2 ± 1.03
lower LR=2e-5	30.8 ± 0.95	32.2 ± 0.96	37.8 ± 1.00	52.8 ± 1.02
HellaSwag (Raw)				
MatFormer	29.2 ± 0.45	30.1 ± 0.46	30.7 ± 0.46	31.3 ± 0.46
default	26.2 ± 0.44	26.9 ± 0.44	29.0 ± 0.45	31.1 ± 0.46
1M steps	26.3 ± 0.44	28.2 ± 0.45	30.7 ± 0.46	32.5 ± 0.47
no teacher	26.0 ± 0.44	27.1 ± 0.44	28.9 ± 0.45	31.0 ± 0.46
uniform sampling	26.2 ± 0.44	27.1 ± 0.44	28.9 ± 0.45	29.9 ± 0.46
heads 2, 3, 4	25.7 ± 0.44	27.1 ± 0.44	29.1 ± 0.45	31.2 ± 0.46
KD=0.3	25.7 ± 0.44	26.9 ± 0.44	28.8 ± 0.45	31.1 ± 0.46
FW-Edu 10BT	25.9 ± 0.44	26.9 ± 0.44	28.6 ± 0.45	31.2 ± 0.46
EMA=0.999	25.7 ± 0.44	25.6 ± 0.44	26.1 ± 0.44	31.6 ± 0.46
batch size=128	25.7 ± 0.44	26.3 ± 0.44	27.6 ± 0.45	32.1 ± 0.47
lower LR=2e-5	25.5 ± 0.43	26.3 ± 0.44	27.0 ± 0.44	33.7 ± 0.47
HellaSwag (Norm)				
MatFormer	31.6 ± 0.46	33.6 ± 0.47	34.6 ± 0.47	36.1 ± 0.48
default	26.4 ± 0.44	28.3 ± 0.45	31.2 ± 0.46	35.6 ± 0.48
1M steps	27.4 ± 0.44	30.3 ± 0.46	34.4 ± 0.47	38.4 ± 0.49
no teacher	26.2 ± 0.44	27.9 ± 0.45	30.7 ± 0.46	35.2 ± 0.48
uniform sampling	26.5 ± 0.44	28.2 ± 0.45	31.0 ± 0.46	33.5 ± 0.47
heads 2, 3, 4	26.0 ± 0.44	28.3 ± 0.45	32.2 ± 0.46	36.4 ± 0.48
KD=0.3	26.0 ± 0.44	27.8 ± 0.45	31.3 ± 0.46	35.8 ± 0.48
FW-Edu 10BT	26.3 ± 0.44	27.7 ± 0.45	31.3 ± 0.46	35.7 ± 0.48
EMA=0.999	26.0 ± 0.44	26.1 ± 0.44	26.3 ± 0.44	37.2 ± 0.48
batch size=128	24.8 ± 0.43	27.1 ± 0.44	29.7 ± 0.46	38.0 ± 0.48
lower LR=2e-5	25.5 ± 0.43	26.3 ± 0.44	29.1 ± 0.45	40.8 ± 0.49
TriviaQA				
MatFormer	1.47 ± 0.09	4.70 ± 0.16	5.18 ± 0.17	6.19 ± 0.18
default	0.15 ± 0.03	0.23 ± 0.04	3.69 ± 0.14	5.51 ± 0.17
1M steps	0.13 ± 0.03	0.66 ± 0.06	2.35 ± 0.11	3.55 ± 0.14
no teacher	0.01 ± 0.01	0.02 ± 0.01	1.87 ± 0.10	2.16 ± 0.11
uniform sampling	0.14 ± 0.03	0.33 ± 0.04	3.38 ± 0.13	5.48 ± 0.17
heads 2, 3, 4	0.00 ± 0.00	0.25 ± 0.04	3.81 ± 0.14	6.50 ± 0.18
KD=0.3	0.14 ± 0.03	0.18 ± 0.03	3.72 ± 0.14	6.88 ± 0.19
FW-Edu 10BT	0.11 ± 0.02	0.22 ± 0.04	2.89 ± 0.12	6.12 ± 0.18
EMA=0.999	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.01	3.90 ± 0.14
batch size=128	0.00 ± 0.00	0.17 ± 0.03	2.67 ± 0.12	7.17 ± 0.19
lower LR=2e-5	0.00 ± 0.00	0.02 ± 0.01	1.33 ± 0.09	8.00 ± 0.20

B.2.3 CETT-Based Importance

Finally, the metric based on CETT (Zhang et al., 2024) measures the extent to which tail truncation affects the component output. We adapt this idea into per-unit scores based on the norm of each unit's direct contribution to the output.

For FFN neurons, let $h^{(\ell)}_{b,t,j}$ denote the j-th component of the input to the down projection in layer ℓ, and let $W_{\text{down},\ell}[:, j]$ denote the corresponding output weight vector. The contribution norm of neuron j is:

$$\begin{aligned} c^{(\ell)}_{b,t,j} &= \|h^{(\ell)}_{b,t,j} W_{\text{down},\ell}[:, j]\|_2 \\ &= |h^{(\ell)}_{b,t,j}| \cdot \|W_{\text{down},\ell}[:, j]\|_2 \end{aligned} \tag{10}$$

For attention heads, let $u^{(\ell)}_{b,t,h} \in \mathbb{R}^{d_h}$ denote the output of head h before the output projection, and let $W^{O}_{\ell,h} \in \mathbb{R}^{d_{\text{model}} \times d_h}$ be the corresponding output-projection slice. The head contribution norm is:

$$c^{(\ell)}_{b,t,h} = \|W^{O}_{\ell,h} u^{(\ell)}_{b,t,h}\|_2 \tag{11}$$

The unnormalized CETT score is the empirical mean contribution norm. The normalized variant divides the contribution norm by the norm of the full component output at the same token position. For FFN neurons, this gives:

$$s^{\text{cett}}_{j,\ell} = \mathbb{E}_{b,t}\left[\frac{c^{(\ell)}_{b,t,j}}{\|y^{\text{FFN},(\ell)}_{b,t}\|_2}\right] \tag{12}$$

For attention heads, this gives:

$$s^{\text{cett}}_{h,\ell} = \mathbb{E}_{b,t}\left[\frac{c^{(\ell)}_{b,t,h}}{\|y^{\text{Attn},(\ell)}_{b,t}\|_2}\right] \tag{13}$$

The normalized form is the most relevant variant for prefix slicing because it estimates a unit's contribution relative to the total component output. Since adaptive slicing removes the tail of a dimension, this ranking directly targets the expected relative output error incurred when we discard low-ranked units.

B.2.4 Effect of Importance Reordering

Table 11 complements the perplexity comparison in Section 5.1 with downstream accuracy after 20,000 training steps. Each row corresponds to a different combination of importance metrics for embeddings, attention heads, and FFN units. Mag-CETT-CETT improves TriviaQA exact-match accuracy across all head configurations, while ARC-Easy and HellaSwag remain close to the no-reordering baseline.

C Additional Multimodal Results

This section provides additional results on text recovery after multimodal training and the optimization behavior of the elastic projection during alignment.

C.1 Text Recovery after Multimodal Training

Table 12 lists TriviaQA accuracies after text-only continued pre-training and after subsequent multimodal or text-only fine-tuning stages. The comparison illustrates how teacher choice and continued fine-tuning affect recovery of factual recall after the multimodal stage. The text-only baseline is trained for one million steps. We compare it with the default multimodal run and with two text fine-tuning variants that use either Gemma 3 270M or Gemma 3 4B as the teacher.

The Gemma 3 4B teacher consistently outperforms the 270M teacher at the four-head configuration, suggesting that the larger teacher transfers more factual knowledge to the student. The improved accuracy relative to the long-run text baseline is likely due to the shift from packed context-window training to training on individual padded samples, together with the lower learning rate and larger batch size used in this stage.

C.2 Projection Alignment Loss Curves

Figure 7 shows the training loss for each head configuration during projection alignment. The one-head configuration converges to a substantially higher loss than the wider configurations, indicating that the single-head subnetwork reaches its capacity limit earlier. Configurations with two or more heads quickly reduce their loss and stabilize around steps 50-100, consistent with the rapid plateau observed in the main text.

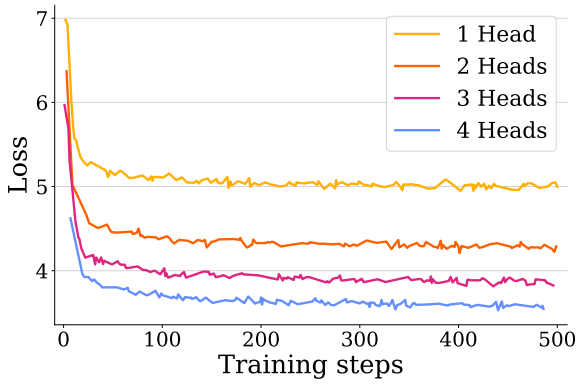


Figure 7: Training loss per-head configuration during projection alignment. The one-head model converges to a higher loss, while configurations with two or more heads reduce their loss rapidly and stabilize within the first 100 steps.

Table 11: Effect of importance reordering on the four nested runtime head configurations after 20,000 training steps. Values are accuracies in percent with standard error. ARC-Easy Norm is answer-length-normalized accuracy. TriviaQA is five-shot exact-match accuracy. The no-reordering variant is the default baseline from Table 10.

Variant	H1	H2	H3	H4
ARC-Easy (Raw)				
no reordering	39.0 ± 1.00	46.3 ± 1.02	51.9 ± 1.03	56.4 ± 1.02
Mag-CETT-CETT	38.4 ± 1.00	46.3 ± 1.02	52.2 ± 1.02	56.6 ± 1.02
Mag-Mag-Mag	37.6 ± 0.99	46.8 ± 1.02	52.9 ± 1.02	57.1 ± 1.02
VxC-VxC-VxC	36.9 ± 0.99	46.8 ± 1.02	51.9 ± 1.03	56.7 ± 1.02
Mag-no-Mag	38.3 ± 1.00	46.3 ± 1.02	53.2 ± 1.02	56.5 ± 1.02
Mag-VxC-Mag	37.7 ± 0.99	45.4 ± 1.02	52.1 ± 1.02	56.0 ± 1.02
ARC-Easy (Norm)				
no reordering	35.8 ± 0.98	40.0 ± 1.01	46.5 ± 1.02	50.8 ± 1.03
Mag-CETT-CETT	34.7 ± 0.98	41.6 ± 1.01	46.9 ± 1.02	50.5 ± 1.03
Mag-Mag-Mag	33.9 ± 0.97	42.0 ± 1.01	46.9 ± 1.02	51.1 ± 1.03
VxC-VxC-VxC	34.2 ± 0.97	41.4 ± 1.01	47.3 ± 1.02	51.1 ± 1.03
Mag-no-Mag	36.2 ± 0.99	41.1 ± 1.01	47.3 ± 1.02	49.4 ± 1.03
Mag-VxC-Mag	34.9 ± 0.98	40.7 ± 1.01	46.3 ± 1.02	51.3 ± 1.03
HellaSwag (Raw)				
no reordering	26.2 ± 0.44	26.9 ± 0.44	29.0 ± 0.45	31.1 ± 0.46
Mag-CETT-CETT	26.2 ± 0.44	26.8 ± 0.44	28.9 ± 0.45	30.9 ± 0.46
Mag-Mag-Mag	26.1 ± 0.44	27.0 ± 0.44	28.5 ± 0.45	31.2 ± 0.46
VxC-VxC-VxC	26.1 ± 0.44	26.9 ± 0.44	28.9 ± 0.45	31.1 ± 0.46
Mag-no-Mag	26.0 ± 0.44	26.8 ± 0.44	28.7 ± 0.45	31.2 ± 0.46
Mag-VxC-Mag	26.1 ± 0.44	26.8 ± 0.44	28.8 ± 0.45	31.1 ± 0.46
HellaSwag (Norm)				
no reordering	26.4 ± 0.44	28.3 ± 0.45	31.2 ± 0.46	35.6 ± 0.48
Mag-CETT-CETT	26.3 ± 0.44	28.0 ± 0.45	31.2 ± 0.46	36.0 ± 0.48
Mag-Mag-Mag	26.3 ± 0.44	28.7 ± 0.45	31.4 ± 0.46	35.9 ± 0.48
VxC-VxC-VxC	25.6 ± 0.44	28.1 ± 0.45	31.3 ± 0.46	35.7 ± 0.48
Mag-no-Mag	26.2 ± 0.44	28.0 ± 0.45	31.8 ± 0.46	35.9 ± 0.48
Mag-VxC-Mag	25.8 ± 0.44	28.4 ± 0.45	31.8 ± 0.46	35.6 ± 0.48
TriviaQA				
no reordering	0.15 ± 0.03	0.23 ± 0.04	3.69 ± 0.14	5.51 ± 0.17
Mag-CETT-CETT	0.21 ± 0.03	0.82 ± 0.07	4.84 ± 0.16	6.63 ± 0.19
Mag-Mag-Mag	0.14 ± 0.03	0.08 ± 0.02	3.08 ± 0.13	5.59 ± 0.17
VxC-VxC-VxC	0.00 ± 0.00	0.77 ± 0.07	3.73 ± 0.14	5.43 ± 0.17
Mag-no-Mag	0.11 ± 0.02	0.45 ± 0.05	4.26 ± 0.15	5.57 ± 0.17
Mag-VxC-Mag	0.03 ± 0.01	0.79 ± 0.07	2.99 ± 0.13	4.60 ± 0.16

Table 12: TriviaQA accuracy (exact match, %) across head configurations for text-only continued pre-training and subsequent multimodal or text-only fine-tuning variants.

Variant	H1	H2	H3	H4
text, 1M steps	0.13	0.66	2.35	3.55
multimodal, default	0.80	2.16	4.08	6.48
text fine-tune, 270M teacher	0.12	1.10	3.64	5.07
text fine-tune, 4B teacher	0.08	2.13	4.56	7.32

Can Child Language Acquisition be Modeled by Linear Causal or Bidirectional Masked Models Trained on Child-Directed Input?: A Comparison of Baby Language Models and Children in Acquiring *Wanna* Contraction

Haerim Hwang
The Chinese University of Hong Kong
haerimhwang@cuhk.edu.hk

Abstract

Children's remarkable ability to acquire language remains one of the most fascinating questions in cognitive science. How do they develop sophisticated syntactic knowledge? To contribute to this question, the present study investigates the role of input and learning mechanisms by modeling child language acquisition across increasing age. Specifically, this study trains 30 baby language models (LMs) on naturalistic input directed to children up to ages 3 through 7 and compares their syntactic knowledge directly with that of children aged 3 to 7, focusing on the constraint on *wanna* contraction, a phenomenon not previously tested in language modeling. Furthermore, the study examines two major LM architectures: an autoregressive causal LM, represented by GPT, and a bidirectional masked LM, represented by RoBERTa. The results showed that children were sensitive to the constraint and displayed age-related development, but neither GPT-based nor RoBERTa-based baby models showed the same pattern. In this respect, the baby LMs do not seem to fully capture human-like development, possibly suggesting that the learning mechanisms underlying child language acquisition do not entirely align with the architectures tested here and cannot be reduced to input-driven learning alone within the context of these models.

1 Introduction

The question of how children acquire grammatical knowledge has been central to research in linguistics and cognitive science (Hsu et al., 2011; Lidz and Gagliardi, 2015; Linzen and Baroni, 2021; Piantadosi, 2023; Yang, 2016; Yang et al., 2017). Is the linguistic input available to children sufficient to explain the grammatical knowledge they ultimately develop (see also Chang et al., 2025)? Some theoretical approaches hold that language acquisition is fundamentally input-driven, with domain-general cognitive abilities supporting the abstraction of linguistic forms (e.g., Lewis and Elman, 2001; Tomasello, 2003). According to other approaches, the input children receive is too sparse and underdetermined to lead to the acquisition of abstract, hierarchical structures, making innate language-specific cognitive mechanisms necessary (e.g., Chomsky, 1968; Chomsky, 1973; Crain et al., 2006; Yang, 2004). The current study aims to contribute to this area of research by investigating how children and baby language models (LMs), trained on child-directed input, evaluate sentences that involve an abstract structural dependency. While a number of studies have moved in this direction (e.g., Padovani et al., 2025), few have included language performance data from human children, thereby limiting the extent to which additional insights can be gained.

The significance of the present study lies in two main respects. First, this study directly compares the performance of children with that of baby LMs to reveal whether the processes underlying child language acquisition are similar to, or different from, the mechanisms these models rely on for language learning. Specifically, the study examines two major architectures: an auto-regressive causal model, represented by GPT (Radford et al., 2019), and a bidirectional masked model, represented by RoBERTa (Zhuang et al., 2021). These two approaches learn in fundamentally different ways: the former learns by predicting each next token based on the preceding ones in a linear, left-to-right fashion, consistent with the error-based learning account that humans process language incrementally and learn when the incoming input diverges from their expectations (Chang et al., 2012); the latter learns by reconstructing masked tokens from their surrounding context, thereby building strong bidirectional text representations that may better capture structural dependencies existing in human languages across both visible and invisible

Proceedings of the Second BabyLM Workshop, pages 50–64
October 29, 2026 ©2026 Association for Computational Linguistics

linguistic material (e.g., Jawahar et al., 2019).

Furthermore, the study systematically varies the amount of input and vocabulary size in LMs to simulate child language acquisition. As an initial step in examining the role of input in language acquisition, it trains five LMs on different datasets representing the input available up to ages 3 through 7, while comparing their performance against children's performance. Specifically, the study uses the CHILDES corpus (MacWhinney and Erlbaum, 2000), which contains naturalistic child-directed input, rather than relying on data that is commonly used in this area of research (e.g., Wikipedia).

The target linguistic phenomenon in this study is the contraction of *wanna*, which is a reduced form of *want to*. This construction yields a subtle contrast in grammaticality, as shown in (1) versus (2). Whereas *want to* cannot contract to *wanna* when the question is about the subject of the *to*-infinitival clause (e.g., (2)), such contraction is possible if the question is about any other constituents within the *to*-infinitival clause, such as a direct object of the verb (e.g., (1a)) or the preposition (e.g., (1b)). Some linguists assume that the contrast in *wanna* contraction results from the blocking effect created by *wh*-movement. In (2), the subject of the *to*-infinitival complement is extracted from the position between *want* and *to* and moved to the front. This movement leaves behind a gap (which is marked as _) in its original position, preventing *want* and *to* from contracting across it. In contrast, (1) involves the extraction of non-subject elements from positions elsewhere within the infinitival clause, not from between *want* and *to*, so no blocking effect arises. In this way, *wanna* contraction is subject to a structural constraint (cf., Getz, 2019; O'Grady et al., 2008). The present study focuses on this phenomenon because it can be expressed using simple vocabulary, making it suitable for child participants and baby LMs, while still exhibiting structural dependencies.

(1) a. Who do you [want to, wanna] kiss _?

b. Who do you [want to, wanna] dive with _?

(Zukowski and Larsen, 2011, p. 213, (1a)–(1b))

(2) a. Who do you [want _ to, *wanna] kiss you?

b. Who do you [want _ to, *wanna] dive first?

(Zukowski and Larsen, 2011, p. 213, (2))

Child language acquisition research on *wanna* contraction showed somewhat inconsistent results, mostly using a production paradigm (cf., Hwang, 2024). Thornton (1990) reported that children as young as 3;6 demonstrated target-like knowledge of the constraint on *wanna* contraction. However, Zukowski and Larsen (2011), in a later study that addressed a methodological issue involving the use of different prompts across experimental conditions in Thornton (1990)'s study, suggested that young children (age: 4;00–7;03) do not possess grammatical knowledge of *wanna* contraction. A similar pattern was shown by Getz (2019), whose findings showed that children's use of *wanna* diverged from adults across age groups (younger group's age: 3;09—4;10; older group's age: 5;00—7;03). The study also found a frequency effect limited to impossible *wanna* contraction contexts, where children produced fewer impossible contractions when the embedded verb was more frequent. Although this finding appears to support the view that input frequency plays a decisive role in child language acquisition, it is important to note that the frequency effect emerged only in the case of impossible *wanna* contraction. If children learn through the storage of specific patterns and their associated frequencies, then a similar frequency effect should also be expected for possible *wanna* contraction, at least among younger children, who are still building abstract generalizations on the basis of item-based patterns (Hwang, 2024).

It is only in recent work that children's knowledge of *wanna* contraction has been tested in comprehension. Since production is more likely to be affected by performance factors, such as cognitive demands, and since an inability to produce a form does not necessarily reflect a lack of underlying knowledge, Hwang (2024) assessed children's knowledge using a child-friendly acceptability judgment task. The results indicated that children as young as 3;11 can distinguish between possible and impossible sentences involving *wanna* contraction, consistent with Thornton (1990).

Notably, while previous language modeling studies have examined linguistic phenomena involving structural dependencies, including parasitic gaps and across-the-board movement (Lan et al., 2026), as well as island effects (Chang et al., 2025; McInnerney, 2025), no study has specifically investigated *wanna* contraction, despite the considerable attention it has received in child language acquisition (see above). Moreover, this line of research

has yielded mixed findings. McInnerney (2025) reported limited evidence of island effects in causal and masked LMs, only when the models were trained on developmentally implausible quantities of data, including Wikipedia and the Pile (Gao et al., 2020). Chang et al. (2025) addressed this issue by using "predominantly" child-oriented and child-sized input to train their LMs and found that the LMs did not consistently achieve full structural generalization across the linguistic phenomena they tested. Building on the work of Chang et al. (2025) but using "only" child-oriented data, the present study extends the investigation to consider the following research questions (RQs).

- RQ 1: Can LMs trained on child-oriented, child-sized input acquire the constraint on *wanna* contraction?
- RQ 2: Do LMs exhibit child-like age-related development of sensitivity to the constraint on *wanna* contraction, as they are trained on datasets representing older children and a larger vocabulary size?
- RQ 3: Which LM type is better at learning *wanna* contraction: linear causal models or bidirectional masked models?

These RQs give rise to several plausible predictions. For RQ 2, it is conceivable that baby LMs trained on a dataset covering older children and employing a larger vocabulary size may align more closely with older children's developmental patterns, thereby showing greater sensitivity than baby LMs trained on a dataset restricted to younger children and a smaller vocabulary size. As for RQ 3, GPT-based LMs' performance may better resemble children's because its auto-regressive architecture mirrors the incremental way in which children process language in real time and predict the upcoming information (Chang et al., 2012). Alternatively, RoBERTa-based LMs' performance may align more closely with children's because language itself relies on bidirectional dependencies. Even though language learners do not encounter sentences in both directions, findings from a preferential-looking experiment using eye-tracking indicate that they reanalyze and fix their initial parses and interpretations (e.g., Abbot-Smith et al., 2017). However, it is also possible that neither model architecture, nor any dataset designed to approximate human input, will show human-like performance (RQ 1 and RQ 3). Taken together, the findings of this study may provide insight into whether language acquisition is solely based on autoregressive or bidirectional parsing mechanisms and/or whether it can be understood as purely input-driven within such learning architectures. Ultimately, they are expected to shed light on the mechanisms involved in child language acquisition.

2 Methodology

The methodology used in the present study draws on earlier work examining the grammatical knowledge of LMs (e.g., Chang et al., 2025; Howitt et al., 2024; Ozaki et al., 2022; Wilcox et al., 2018). Prior to the beginning of the study, administrative and ethical approval had been obtained, along with informed consent from one parent of each child participant.

2.1 Child participants

Data from 46 child participants were drawn from Hwang (2024). All participants lived in the United States at the time of testing. The age distribution was as follows: 3-year-olds (n = 2), 4-year-olds (n = 3), 5-year-olds (n = 13), 6-year-olds (n = 19), and 7-year-olds (n = 9). These participants were recruited through flyers distributed to their parents at a local kindergarten. Upon completing the study, all participants were given stationery items equivalent to $5 (e.g., stickers), following the convention in the area where they resided.

2.2 Baby language models

The current study sought to approximate the linguistic input available in child language acquisition by training a range of baby LMs that differed in three main respects: (1) model architecture, (2) vocabulary size, and (3) random seed. With respect to architecture, two types of LM were used: GPT-2-based models (Radford et al., 2019), which instantiate linear causal modeling, and RoBERTa-based models (Zhuang et al., 2021), which instantiate bidirectional masked modeling. This comparison enables to assess whether, and to what extent, the learning procedures associated with these model types can account for the acquisition of abstract syntactic knowledge. To ensure comparability, both models used the same architectural specifications, with 8 transformer layers, 8 attention heads, an embedding size of 512, and an intermediate feed-forward size of 2,048.

The specific model configurations in the present study are provided in Appendix A. For each LM type, 15 separate models were trained on authentic linguistic input directed to English-speaking children. Given that the child participants in this study resided in the United States at the time of testing, all data for training LMs were drawn from the North American English portion of CHILDES (MacWhinney and Erlbaum, 2000). On the basis of this corpus, five datasets were constructed, with each representing input available up to ages 3, 4, 5, 6, and 7 (see Appendix B).

Vocabulary size was also systematically varied so that it would approximate the vocabulary growth typically observed across these age groups. In the context of language modeling, vocabulary size, which is defined as the number of distinct token types that an LM can represent, determines how finely text is segmented into units, such as words or subwords. This factor is important for investigating LM performance because it directly affects sequence length, memory and computational cost, and the extent to which the model captures linguistic patterns rather than artifacts of tokenization. Following Shipley and McAfee (2015), vocabulary sizes were set to 900 for 3-year-olds, 2,000 for 4-year-olds, 10,000 for 5-year-olds, 13,000 for 6-year-olds, and 20,000 for 7-year-olds. For each of the two architectures, baby LMs were trained on each of the five datasets three times using different random seeds, namely 13, 30, and 42.

Importantly, the present modeling setup was designed to approximate age-conditioned learner states rather than to isolate the independent contribution of any single developmental factor. To approximate children's linguistic experience as closely as possible, the age conditions were jointly varied in terms of exact input data, vocabulary size, and resulting parameter count, mirroring how these factors naturally change together during language development. In this respect, the models should be interpreted as cross-sectional snapshots of learners at different points in development, consistent with our cross-sectional design for child participants with different ages, rather than as sequential checkpoints from a single learner tracked longitudinally (Shah et al., 2024). Overall, the goal of this study is ecological approximation rather than causal decomposition. That is, the study focuses on whether models trained under age-indexed input conditions show behavior that aligns with children's performance, rather than which individual component of development is responsible for such behavior.

The codebase used in this study was adapted from Padovani et al. (2025)[1] solely for the purposes of this research. Following their setup, all models were trained from scratch for 100,000 steps using the AdamW optimizer (Loshchilov and Hutter, 2017), together with a linear scheduling scheme and a 40,000-step warm-up phase, which ensured consistent training conditions across models. A learning rate of 0.0001 was also applied uniformly across all experiments (Padovani et al., 2025). The data were split into training and validation sets in a 92/8 ratio. Each baby LM training run required roughly 1 hour and 50 minutes on an RTX 5090. Training and evaluation loss curves of the trained LMs are provided in Appendix C.

2.3 Experimental design

The experiment employed an acceptability judgment task given that this paradigm is well established and widely validated in child language acquisition research (e.g., Ambridge et al., 2008). In the task, the two factors, Clause (Who vs. If) and Gap (Gap vs. No gap), were crossed in a 2 × 2 Latin square design. This yielded four conditions, as shown in (3). In total, 80 sentences were developed, with 20 sentences for each condition (for the full list, see Appendix D). The two If conditions served as the baseline for comparison with the two Who conditions in terms of grammaticality: the *If + Gap condition was contrasted with the Who + Gap condition, and the If + No gap condition was contrasted with the *Who + No gap condition.

(3) a. Who + Gap: I wonder who you wanna work with.

b. *Who + No gap: *I wonder who you wanna work.

c. *If + Gap: *I wonder if you wanna work with.

d. If + No gap: I wonder if you wanna work.

To reduce the cognitive demands placed on both child participants and baby LMs trained on child-directed input, all experimental sentences were kept short (6–7 words) and composed of simple, high-frequency vocabulary. The embedded verbs following *wanna* were either clearly intransitive (k = 16;

[1] https://github.com/fpadovani/childes_vs_wiki

e.g., *work*) or verbs that can be used both transitively and intransitively but do not normally take a human object when transitive (k = 4; e.g., *read*, *walk*, *eat*, *drink*).

2.4 Experimental procedure

Child participants completed an acceptability judgment task where every sentence was presented both aurally and in writing (for the specific instructions that the child participants received, see Appendix E). For each trial, participants heard the sentence twice, separated by a one-second interval, while the corresponding sentence was shown on the screen. They then provided their judgments on a four-point smiley-face scale consisting of four response options, including "very bad," "bad," "good," and "very good," alongside an "I don't know" option, which was indicated by a question mark. To avoid practice effects, participants were not given different versions of the same item set (e.g., (3a), (3b), (3c), (3d))). Instead, they saw only one version from each item set, meaning that they were presented with five items per condition, for a total of 20 target items, in addition to 44 fillers.

The syntactic performance of each LM was assessed on the basis of sentence probabilities, following earlier studies (Linzen et al., 2016; Marvin and Linzen, 2018; Wilcox et al., 2018), using the "minicons" library (Misra, 2022). For linear causal models, the summed sequence log-probability with Beginning-of-Word (BOW) correction was used (Pimentel and Meister, 2024; Oh and Schuler, 2024). For bidirectional masked models, a likelihood score based on a within-word left-to-right masking strategy was used, as this reduces the tendency to overestimate token probabilities for words consisting of multiple tokens (Kauf and Ivanova, 2023). Each LM's probabilities were calculated for all 80 experimental sentences.

2.5 Analysis

Prior to conducting the statistical analyses, the children's "I don't know" responses were removed (5.22%). For analysis, "very bad," "bad," "good," and "very good," were respectively coded as 1, 2, 3, and 4. Next, z-transformations were applied to the children's acceptability scores and the LMs' sentence probabilities to place both measures on a common, standardized scale (M = 0, SD = 1) (e.g., Suijkerbuijk et al., 2025). This procedure minimizes differences in scale and variance between the raw ratings and model probabilities, enabling direct comparisons across the two measures.

To evaluate LM's performance on *wanna* contraction (RQ 1), a linear mixed-effects regression model was constructed with Clause (Who vs. If), Gap (Gap vs. No gap), Learner type (Children vs. LMs) as categorical fixed effects, Age as a continuous fixed effect, and Item as a random effect. The interaction between Construction and Gap is crucial, as grammaticality aligns with the Gap condition in Who clauses but with the No Gap condition in If clauses; therefore, collapsing these into a single variable, such as grammaticality, could obscure important patterns in the data.

To examine age-related development in sensitivity to the constraint on *wanna* contraction (RQ 2), separate linear mixed-effects regression models were fitted to the child and LM data. These models included Clause (Who vs. If), Gap (Gap vs. No gap), and Age as fixed effects, and Item as a random effect. For a more detailed investigation of age-related effects, grammaticality sensitivity scores were computed for each Clause type by subtracting the acceptability scores of the ungrammatical condition from those of the grammatical condition: i.e., Who + Gap −*Who + No gap; If + No gap −*If + Gap. These grammaticality sensitivity scores were then analyzed using a simple linear regression model with Age as a predictor.

To explore any effect of model architecture (RQ 3), another linear mixed-effects regression model was constructed on the LM data. This model included Clause (Who vs. If), Gap (Gap vs. No gap), and Model architecture (linear causal model vs. bidirectional masked model) as fixed effects, and Item as a random effect.

3 Results

3.1 Knowledge of the constraint on *wanna* contraction

A visual inspection of Figure 1 indicates that the children aged 5 and older rated the Who + Gap and If + No gap conditions higher than the *Who + No gap and *If + Gap conditions, while the children aged 3 and 4 showed no clear pattern. In contrast, the baby LMs consistently rated the If + No gap condition highest, followed by the *If + Gap condition, then the *Who + No gap condition, which became rated more highly than the *If + Gap condition with increasing age—at age 4 for the linear causal model and age 5 for the bidirectional masked model—and finally the Who + Gap condition. This

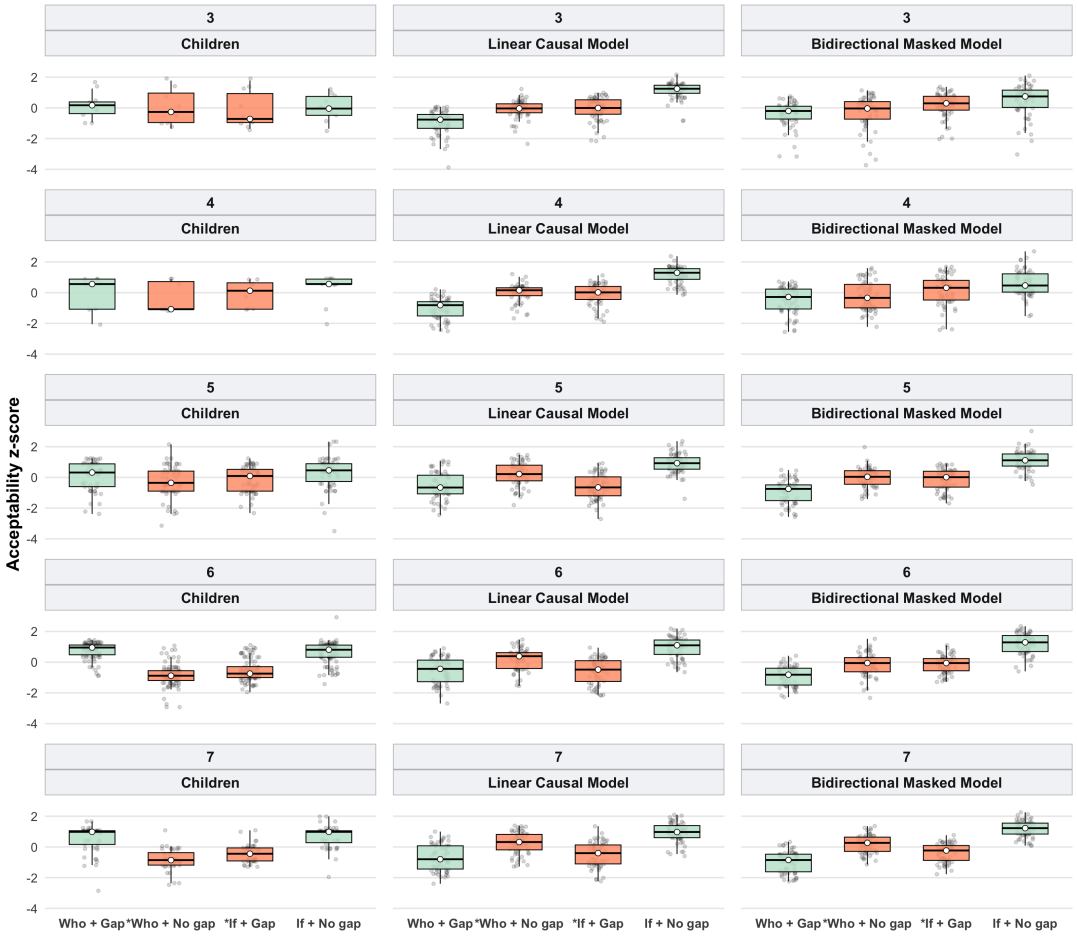


Figure 1: Mean acceptability z-scores by learner type and condition

Note. The results visualized for each language model with a distinct model architecture trained on input directed to children up to ages 3 through 7 are averaged over three random seeds.

pattern indicates that, although the children possess grammatical knowledge of the constraint on *wanna* contraction, the baby LMs developed for this study do not.

As shown in Appendix F, a linear mixed-effects regression model fitted to the acceptability z-scores revealed a significant effect of Clause (b = .427, SE = .147, p = .004), as well as significant interactions between Clause and Gap (b = −1.06, SE = 0.29, p < .001), between Clause and Learner type (b = 1.29, SE = 0.29, p < .001), between Clause, Gap, and Learner type (b = 2.10, SE = 0.59, p < .001), between Clause, Gap, and Age (b = 0.40, SE = 0.05, p < .001), betewen Clause, Learner type, and Age (b = −0.13, SE = 0.05, p = .012), and between Gap, Learner type, and Age (b = 0.24, SE = 0.05, p < .001). Crucially, the model also identified a four-way interaction among Clause, Gap, Learner type, and Age (b = −0.62, SE = −6.05, p < .001). These results suggest that the children and the LMs differed in their performance on experimental sentences involving *wanna* contraction, consistent with our visual inspection.

3.2 Age-related development of sensitivity to the constraint on *wanna* contraction

To further investigate the Learner type-related interaction observed in the previous section, separate follow-up linear mixed-effects regression models were constructed for each learner type. In the child model, a significant Clause × Gap interaction was found (b = −2.52, SE = 0.65, p < .001), as shown in Appendix G. Post hoc pairwise comparisons further indicated that children rated the Who + Gap condition higher than the *Who + No gap condition (b = 1.09, SE = 0.08, p < .001) and the If + No gap condition higher than the *If + Gap condition (b = 0.89, SE = 0.08, p < .001), suggesting robust knowledge of the constraint on *wanna* contraction. In addition, the model revealed a significant interaction among Clause, Gap, and Age (b = 0.79, SE = 0.57, p < .001). Subsequent simple regression

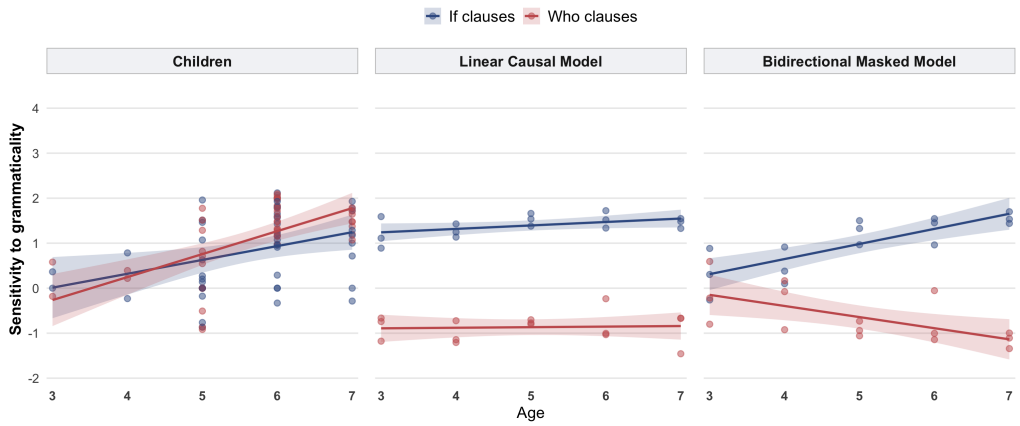


Figure 2: Relationship between age (whether measured in the case of children or approximated from the child-directed input in the case of language models) and grammaticality sensitivity scores by learner type

analyses showed that sensitivity to grammaticality increased with children's age for both Who clauses (b = 0.51, SE = 0.10, p < .001) and If clauses (b = 0.31, SE = 0.12, p = .012), as shown in Figure 2.

A different pattern emerged from the linear mixed-effects regression model fitted to the baby LM data, as shown in Appendix H. The model revealed a significant effect of Clause (b = 1.07, SE = 0.09, p < .001), as well as significant interactions between Clause and Age (b = –0.07, SE = 0.02, p < .001) and between Gap and Age (b = 0.16, SE = 0.02, p < .001). There was also a significant interaction among Clause, Gap, and Age (b = 0.09, SE = 0.03, p = .008). However, subsequent simple regression analyses indicated that the pattern of age-related development in the baby LMs was different from the one found in children. These analyses showed that sensitivity to grammaticality tended to increase with the baby LMs' age, as approximated by the input, for If clauses (b = 0.21, SE = 0.05, p < .001), but to decrease with the baby LMs' age for Who clauses (b = –0.12, SE = 0.06, p = .049), although the pattern differed across the two model architectures (see Figure 2).

3.3 Effects of model architecture

Another linear mixed-effects model applied to the baby LM data showed significant main effects of Clause (b = 0.73, SE = 0.02, p < .001) and Gap (b = 0.97, SE = 0.07, p < .001) (see Appendix I). In addition, the model revealed significant interaction effects between Clause and Gap (b = 0.43, SE = 0.05, p < .001), between Clause and Model architecture (b = 0.19, SE = 0.05, p < .001), and between Gap and Model architecture (b = –0.32, SE = 0.05, p < .001). Crucially, the analysis also identified a significant three-way interaction among Clause, Gap, and Model architecture (b = –0.19, SE = 0.09, p = .049). Further post hoc pairwise comparisons indicated that both the linear causal model and the bidirectional masked model gave higher ratings to the If + No gap condition than to the *If + Gap condition (linear causal model: b = –1.29, SE = 0.08, p < .001; bidirectional masked model: b = –0.98, SE = 0.08, p < .001), but to the *Who + No gap condition than to the Who + Gap condition (linear causal model: b = –0.87, SE = 0.08, p < .001; bidirectional masked model: b = –0.64, SE = 0.08, p < .001). The three-way interaction seems to be thus driven by the larger difference between the two If conditions in the linear causal model compared with the bidirectional masked model, as shown in Figure 2.

4 Discussion

Regarding RQ 1, the present study found that unlike children, baby LMs trained on child-oriented, child-sized input did not demonstrate knowledge of the constraint on *wanna* contraction. This finding is somewhat difficult to reconcile with the view that input is a key factor in the context of the model architectures tested in this study. In particular, the models struggled to accept the Who + Gap condition and to reject the *Who + No gap condition (see Figure 1). A possible reason is that probabilistic language learning may not be sufficient to capture the syntactic dependency between *who* and the gap. In sum, these results seem to be more compatible with the theoretical accounts positing that input-based, domain-general learning via linear causal

or bidirectional masked learning mechanisms may not alone be sufficient and that language-specific learning mechanisms are needed for the acquisition of complex syntactic knowledge (Chomsky, 1968; Chomsky, 1973; Crain et al., 2006; Yang, 2004; cf., Lewis and Elman, 2001; Piantadosi, 2023; Tomasello, 2003).

Furthermore, baby LMs did not exhibit child-like, age-related development in grammaticality sensitivity to the constraint on *wanna* contraction (RQ 2). The models were trained on a range of datasets designed to approximate the authentic input children receive, as well as the vocabulary sizes they develop at different ages. However, their performance did not reflect human-like developmental changes in grammatical sensitivity. More specifically, they exhibited largely similar patterns across age groups and vocabulary sizes, without showing the target-like grammatical contrast between the Who-Gap and *Who-No gap conditions; this contrasts with the child data, which showed clear developmental growth in the target grammatical sensitivity and suggested full attainment by age 5. Although children receive input that is sparse and often underspecified, they nonetheless come to acquire target-like knowledge of the *wanna* contraction constraint without explicit instruction, suggesting that the development of syntactic knowledge in LMs may differ fundamentally from its development in children.

Regarding RQ 3, neither linear causal models nor bidirectional masked models showed child-like performance on sentences involving *wanna* contraction. This result may be considered surprising, given psycholinguistic evidence that human language processing appears to share some features with the mechanisms used by these models. Humans, like linear causal models, use prior linguistic context to predict upcoming input, and mismatches between prediction and input may contribute to learning. In line with this, error-based learning has been argued to play a role in language development (Chang et al., 2006; Phillips and Ehrenhofer, 2015). On this account, learners continuously generate predictions, and when those predictions are disconfirmed, the gap between predicted and actual input provides implicit feedback that helps move the learner toward the target grammar. On the other hand, human sentence processing also involves reanalysis: parsers may go back to earlier parts of a sentence and revise initial interpretations to resolve syntactic dependencies, similar in some respects to bidirectional masked models (Lopopolo et al., 2019; Pickering and Traxler, 1998).

Since both types of LMs failed to show robust knowledge of the *wanna* contraction constraint, unlike children, the present findings suggest that while these models may resemble humans in certain aspects of online processing, they still differ from them in the underlying mechanisms that support the acquisition of low-frequency or weakly evidenced grammatical knowledge. This interpretation also aligns with Murphy's (2025) Representation, Operation, Structure, and Encoding (ROSE) model. In that framework, syntax is divided into two broad domains: a horizontal domain shaped by surface-based distributional regularities, and a vertical domain involving hierarchical and symbolic structure. One reasonable possibility is that LMs are effective at learning surface-level, horizontal patterns but struggle with deeper hierarchical structures, particularly when they are rarer (Hwang, 2026). *Wanna* contraction fits this description, since it depends on complex structural relations and does not provide strong distributional regularities (Hwang, 2024). If LMs are weak at acquiring this type of vertical structure, it is less surprising that they failed to show human-like syntactic knowledge involving structural dependency.

Focusing on vertical structure, Chang et al. (2025) reported a similar pattern of results. Their results showed that although some small GPT-2 models trained primarily on child-oriented input exhibited limited learning of filler-gap dependencies, they still did not achieve human-like island sensitivity. Chang et al. (2025) took this as potentially suggestive of a role for language-dedicated mechanisms in language development. The present findings may be broadly consistent with this interpretation, although it should be noted that alternative architectures and inductive biases remain to be further explored before any firm conclusions can be reached (see Section "Limitations").

5 Conclusion

In conclusion, baby LMs appeared to fall short of fully capturing human-like language development. This suggests some divergence between the learning mechanisms underlying child language acquisition and those instantiated in the tested architectures, as well as the possibility that input-driven learning alone may not be fully sufficient within these model architectures.

Limitations

The conclusions of the present study should be interpreted cautiously in light of several limitations outlined below, which future research should address. One limitation is that the inclusion of child participants and the systematic comparison across age groups made it difficult to examine a wider range of linguistic phenomena. Although the study investigated a novel phenomenon that has not previously been tested in LMs, future work should consider a broader set of phenomena in order to draw stronger generalizations about LMs' syntactic knowledge.

In addition, *wanna* contraction occurs more frequently in spoken than in written language, which may have given child participants an advantage, since the experimental sentences were presented to them in both spoken and written form, whereas the LMs were exposed only to written input. A useful next step would therefore be to investigate whether speech or multimodal LMs show different patterns of syntactic knowledge when auditory forms or both auditory and written forms of the experimental sentences are made available.

While CHILDES offers a large and authentic source of child-directed input, it cannot be assumed to fully reflect the complete linguistic environment to which children are exposed. A more ideal approach would involve collecting input data directly from the children participating in the study and training the language models on this data, so that both the LMs and the children could be tested on syntactic knowledge under more comparable conditions.

Furthermore, it is possible that the LMs developed for this study were too small in scale to adequately approximate the neural capacity that children bring to language learning. Also, the two types of learning mechanisms were represented only by GPT-2 and RoBERTa, corresponding to a linear causal model and a bidirectional masked model, respectively. Although the training data were varied in terms of input age and vocabulary size, the range of models considered remains narrow, which limits the generalizability of our conclusions. Other LMs with different inductive biases, such as LLaMA with decoder-only transformers or GPT-BERT with hybrid architectures, may show different patterns in how they learn syntactic constraints and structures (see also Chang et al., 2025).

Finally, the hyperparameter configuration used in this study was closely based on Padovani et al. (2025) to replicate and optimize model performance. However, computational limitations restricted the search space explored in this study, and thus a wider range of configurations should be explored in further research (see also Chang et al., 2025).

Supplementary material

The dataset and analysis codes used for the current study are available at `https://osf.io/p48ef` under a CC BY-NC-SA 4.0 license. The dataset shared should not be used outside of research contexts.

Acknowledgments

This study was funded by the Early Career Scheme supported to the author by Research Grants Council in Hong Kong (Reference: 24607925).

AI statement

For grammar checks and rephrasing of some sections, AI tools (e.g., Gemini-3, Perplexity) were used. All content remains the original work of the author.

References

Kirsten Abbot-Smith, Franklin Chang, Caroline Rowland, Heather Ferguson, and Julian Pine. 2017. Do two and three year old children use an incremental first-np-as-agent bias to process active transitive and passive sentences?: A permutation analysis. *PLOS ONE*, 12:Article e0186129.

Ben Ambridge, Julian M Pine, Caroline F Rowland, and Chris R Young. 2008. The effect of verb semantic class and verb frequency (entrenchment) on children's and adults' graded judgements of argument-structure overgeneralization errors. *Cognition*, 106:87–129.

Chi-Yun Chang, Xueyang Huang, Humaira Nasir, Shane Storks, Olawale Akingbade, and Huteng Dai. 2025. Mind the gap: How babylms learn filler-gap dependencies. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15049–15065, Suzhou, China. Association for Computational Linguistics.

Franklin Chang, Gary S. Dell, and Kathryn Bock. 2006. Becoming syntactic. psychological review. *Psychological Review*, 113:234–272.

Franklin Chang, Marius Janciauskas, and Hartmut Fitz. 2012. Language adaptation and learning: Getting explicit about implicit learning. *Linguistics and Language Compass*, 6:259–278.

Noam Chomsky. 1968. *Language and Mind*. Cambridge University Press, Cambridge, UK.

Noam Chomsky. 1973. Problems of knowledge and freedom. *Philosophy*, 48:194–195.

Stephen Crain, Takuya Goro, and Rosalind Thornton. 2006. Language acquisition is language change. *Journal of Psycholinguistic Research*, 35:31–49.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv*, 2101.00027:1–39.

Heidi R. Getz. 2019. Acquiring wanna: Beyond universal grammar. *Language Acquisition*, 26:119–143.

Kevin Howitt, Sunil Nair, Amelia Dods, and Robert M. Hopkins. 2024. Generalizations across filler-gap dependencies in neural language models. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, pages 269–279, Miami, FL. Association for Computational Linguistics.

Anne S. Hsu, Nick Chater, and Paul M.B. Vitányi. 2011. The probabilistic analysis of language acquisition: Theoretical, computational, and experimental analysis. *Cognition*, 120:380–390.

Haerim Hwang. 2024. Wanna contraction in first language acquisition, child second language acquisition, and adult second language acquisition. *Bilingualism: Language and Cognition*, 27:322–333.

Haerim Hwang. 2026. Can large language models be viewed as a cognitive model of human language? not yet, regardless of reasoning capability and size. *Acta Pscyhologica*, page Article 107025.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Carina Kauf and Anna Ivanova. 2023. A better way to do masked language model scoring. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 925–935, Florence, Italy. Association for Computational Linguistics.

Nur Lan, Emmanuel Chemla, and Roni Katzir. 2026. Large language models and the argument from the poverty of the stimulus. *Linguistic Inquiry*, 57:315–342.

John D. Lewis and Jeffrey L. Elman. 2001. Learnability and the statistical structure of language: Poverty of stimulus arguments revisited. In *Proceedings of the 26th annual Boston University conference on language development*, pages 359–370, Boston, MA. Cascadilla Press.

Jeffrey Lidz and Annie Gagliardi. 2015. How nature meets nurture: Universal grammar and statistical learning. *Annual Review of Linguistics*, 11:333–353.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7:195–212.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Alessandro Lopopolo, Stefan L. Frank, Antal van den Bosch, and Roel Willems. 2019. Dependency parsing with your eyes: Dependency structure predicts eye regressions during reading. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 77–85.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*, pages 195–212.

Brian MacWhinney and Lawrence Erlbaum. 2000. *The CHILDES project . Volume I: Tools for analyzing talk: Transcription format and programs*. Lawrence Erlbaum, Mahwah, NJ.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Andrew McInnerney. 2025. Transformer modeling of syntactic island conditions across training. *University of Pennsylvania Working Papers in Linguistics*, 31:119–128.

Kanishka Misra. 2022. minicons: Enabling flexible behavioral and representational analyses of transformer language models. arXiv:2203.13112.

Elliot Murphy. 2025. Rose: A universal neural grammar. *Cognitive Neuroscience*, 16:49–80.

Byung-Doh Oh and William Schuler. 2024. Leading whitespaces of language models' subword vocabulary pose a confound for calculating word probabilities. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3464–3472, Miami, Florida. Association for Computational Linguistics.

Satoru Ozaki, Daniel Yurovsky, and Lauren Levin. 2022. How well do lstm language models learn filler-gap dependencies? In *Proceedings of the Society for Computation in Linguistics 2022*, pages 76–88. Association for Computational Linguistics.

William O'Grady, Michiko Nakamura, and Yasuko Ito. 2008. Want-to contraction in second language acquisition: An emergentist approach. *Lingua*, 118:478—-498.

Francesca Padovani, Jaap Jumelet, Yevgen Matusevych, and Arianna Bisazza. 2025. Child-directed language does not consistently boost syntax learning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19735–19756, Suzhou, China. Association for Computational Linguistics.

Colin Phillips and Lara Ehrenhofer. 2015. The role of language processing in language acquisition. *Linguistic Approaches to Bilingualism*, 5:409–453.

Steven Piantadosi. 2023. Modern language models refute chomsky's approach to language. In *From fieldwork to linguistic theory: A tribute to Dan Everett*, pages 353–414. Language Science Press.

Martin J. Pickering and Matthew J. Traxler. 1998. Plausibility and recovery from garden paths: An eye-tracking study. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 24:940–961.

Tiago Pimentel and Clara Meister. 2024. How to compute the probability of a word. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18358–18375, Miami, Florida. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. openai. pages 1–24.

Raj Sanjay Shah, Khushi Bhardwaj, and Sashank Varma. 2024. Development of cognitive intelligence in pre-trained language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9632–9657, Miami, FL. Association for Computational Linguistics.

Kenneth G. Shipley and Julie G. McAfee. 2015. *Assessment in speech-language pathology: A resource manual*. Plural Publishing, San Diego, CA.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. Blimp-nl: A corpus of dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51:1267–1301.

Rosalind J. Thornton. 1990. *Adventures in long-distance moving: The acquisition of complex wh-questions (Unpublished PhD dissertation)*. University of Connecticut, Storrs, CT.

Michael Tomasello. 2003. *Constructing a language: a usage-based theory of language acquisition*. Harvard University Press, Cambridge, MA.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do rnn language models learn about filler-gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels, Belgium. Association for Computational Linguistics.

Charles D. Yang. 2004. Universal grammar, statistics or both? *Trends in Cognitive Sciences*, 8:451–456.

Charles D. Yang. 2016. *The price of linguistic productivity: How children learn to break the rules of language*. MIT Press, Cambridge, MA.

Charles D. Yang, Stephen Crain, Robert C. Berwick, Noam Chomsky, and Johan J. Bolhuis. 2017. The growth of language: Universal grammar, experience, and principles of computation. *Neuroscience Biobehavioral Reviews*, 81:103–119.

Liu Zhuang, Lin Wayne, Shi Ya, , and Zhao Jun. 2021. A robustly optimized bert pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China. Chinese Information Processing Society of China.

Andrea Zukowski and Jaiva Larsen. 2011. Wanna contraction in children: Retesting and revising the developmental facts. *Language Acquisition*, 18:211–241.

Appendices

A Model configurations in the study

The model configurations in the present study are summarized in Table 1. Note that the model setup aims to approximate the bundle of changes that co-occur across development in children's input environment. In naturalistic contexts, older children are exposed to more language input, possess larger vocabularies, and likely have access to richer representational resources for language acquisition. Consequently, the models here therefore represent cross-sectional simulations of development, rather than checkpoint-based alignments of a single learner's development.

B Dataset construction

The datasets for this study were created through two main steps: data extraction followed by preprocessing. First, child-directed input for baby LM training was extracted from CHILDES (MacWhinney and Erlbaum, 2000) in R using the "childesr" package. To ensure that only data from children living in the United States were included, we used the following command: collection = "Eng-NA".

The extracted files, which were in .cha format, were then preprocessed in Python to obtain the child's age from the @ID: lines. Each utterance was cleaned by removing CHAT annotations, such as those indicating body language or other annotations, and unwanted symbols. Only utterances produced by non-child speakers were retained, so

	Linear causal model	Bidirectional masked model
Architecture	GPT-2	RoBERTa
Layers	8	
Attention heads	8	
Intermediate size	2048	
Max sequence length	512	
Total parameters	10.88M for input data to 3-year-olds; 11.16M for input data to 4-year-olds; 13.21M for input data to 5-year-olds; 13.98M for input data to 6-year-olds; 15.77M for input data to 7-year-olds	11.18M for input data to 3-year-olds; 11.75M for input data to 4-year-olds; 15.85M for input data to 5-year-olds; 17.39M for input data to 6-year-olds; 20.98M for input data to 7-year-olds
Learning rate	0.0001	
Scheduler type	linear	
Training batch size	16	
Evaluation batch size	16	
Gradient accumulation step	2	
Vocabulary size and input data	900 for input data to 3-year-olds; 2,000 for input data to 4-year-olds; 10,000 for input data to 5-year-olds; 13,000 for input data to 6-year-olds; 20,000 for input data to 7-year-olds	
Random seeds	13, 30, 42	

Table 1: Model configurations in the study

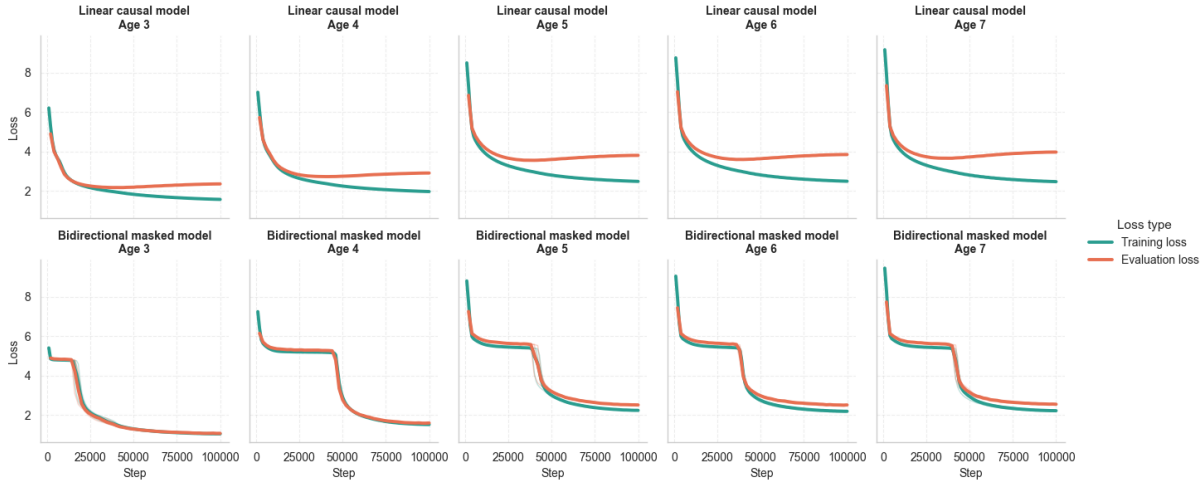


Figure 3: Training and evaluation loss curves

that the resulting dataset reflected child-directed input. For each retained utterance, metadata were recorded, including age, speaker code, corpus, subfolder, transcript filename, file path, and the cleaned utterance text. The complete dataset was saved as "CHILDES_sorted.csv", and cumulative age-based datasets in CSV files were then generated for all utterances available up to ages 3, 4, 5, 6, 7, and 8. The sizes of these datasets, in terms of word tokens and utterances, are summarized in Table 2.

Dataset	Word tokens	Utterances
Input up to age 3	790,259	4,189,835
Input up to age 4	1,007,504	5,432,165
Input up to age 5	1,334,574	7,379,696
Input up to age 6	1,423,082	7,919,487
Input up to age 7	1,468,227	8,188,443

Table 2: Size of five datasets extracted from CHILDES and used for training baby language models

C Training and evaluation loss curves of the baby language models

Training and validation losses were logged at 100-step intervals. The resulting loss curves for the models trained in this study are shown in Figure 3. Training a single model took about 1 hour and 50 minutes on an RTX 5090.

D Experimental stimuli

Below is the full set of experimental sentences employed in this study.

(1) a. I wonder who you wanna read with.

b. *I wonder who you wanna read.

c. *I wonder if you wanna read with.

d. I wonder if you wanna read.

(2) a. I wonder who you wanna dance with.

b. *I wonder who you wanna dance.

c. *I wonder if you wanna dance with.

d. I wonder if you wanna dance.

(3) a. I wonder who you wanna sing with.

b. *I wonder who you wanna sing.

c. *I wonder if you wanna sing with.

d. I wonder if you wanna sing.

(4) a. I wonder who you wanna walk with.

b. *I wonder who you wanna walk.

c. *I wonder if you wanna walk with.

d. I wonder if you wanna walk.

(5) a. I wonder who you wanna stay with.

b. *I wonder who you wanna stay.

c. *I wonder if you wanna stay with.

d. I wonder if you wanna stay.

(6) a. I wonder who you wanna swim with.

b. *I wonder who you wanna swim.

c. *I wonder if you wanna swim with.

d. I wonder if you wanna swim.

(7) a. I wonder who you wanna run with.

b. *I wonder who you wanna run.

c. *I wonder if you wanna run with.

d. I wonder if you wanna run.

(8) a. I wonder who you wanna smile with.

b. *I wonder who you wanna smile.

c. *I wonder if you wanna smile with.

d. I wonder if you wanna smile.

(9) a. I wonder who you wanna start with.

b. *I wonder who you wanna start.

c. *I wonder if you wanna start with.

d. I wonder if you wanna start.

(10) a. I wonder who you wanna talk with.

b. *I wonder who you wanna talk.

c. *I wonder if you wanna talk with.

d. I wonder if you wanna talk.

(11) a. I wonder who you wanna eat with.

b. *I wonder who you wanna eat.

c. *I wonder if you wanna eat with.

d. I wonder if you wanna eat.

(12) a. I wonder who you wanna travel with.

b. *I wonder who you wanna travel.

c. *I wonder if you wanna travel with.

d. I wonder if you wanna travel.

(13) a. I wonder who you wanna drink with.

b. *I wonder who you wanna drink.

c. *I wonder if you wanna drink with.

d. I wonder if you wanna drink.

(14) a. I wonder who you wanna live with.

b. *I wonder who you wanna live.

c. *I wonder if you wanna live with.

d. I wonder if you wanna live.

(15) a. I wonder who you wanna cry with.

b. *I wonder who you wanna cry.

c. *I wonder if you wanna cry with.

d. I wonder if you wanna cry.

(16) a. I wonder who you wanna laugh with.

b. *I wonder who you wanna laugh.

c. *I wonder if you wanna laugh with.

d. I wonder if you wanna laugh.

(17) a. I wonder who you wanna wait with.

b. *I wonder who you wanna wait.

c. *I wonder if you wanna wait with.

d. I wonder if you wanna wait.

(18) a. I wonder who you wanna shout with.

b. *I wonder who you wanna shout.

c. *I wonder if you wanna shout with.

d. I wonder if you wanna shout.

(19) a. I wonder who you wanna pray with.

b. *I wonder who you wanna pray.

c. *I wonder if you wanna pray with.

d. I wonder if you wanna pray.

(20) a. I wonder who you wanna work with.

b. *I wonder who you wanna work.

c. *I wonder if you wanna work with.

d. I wonder if you wanna work.

E Instructions for child participants

The specific instructions that the child participants were presented with were as follows: "This is a 'smiley face' choice task. In this task, you will decide whether sentences are good/possible or bad/impossible in English. First, you will see a sentence on the screen while hearing that same sentence twice on the headphones. After you hear the sentence for the second time, a smiley face scale will pop up on the screen. It is at this point that you can press a number on the keyboard to rate the sentence on a 4-point smiley face scale with an additional "I don't know" option."

F Results of the linear mixed-effects regression for the entire data

Table 3 summarizes the results of the linear mixed-effects regression constructed for the entire data, including Clause, Gap, Learner type, and Age as fixed effects, as well as Item as a random effect.

Table 3: Linear mixed-effects regression results for the entire data

Predictor	***b***	***SE***	***t***	***p***
(Intercept)	0.02	0.11	0.18	.857
Clause	0.43	0.15	2.90	.004
Gap	0.24	0.15	1.57	.117
Learner type	−0.04	0.15	−0.28	.781
Age	0.00	0.01	−0.32	.753
Clause × Gap	−1.06	0.29	−3.59	$< .001$
Clause × Learner type	1.29	0.29	4.37	$< .001$
Gap × Learner type	−0.16	0.30	−0.53	.599
Clause × Age	0.00	0.03	−0.14	.893
Gap × Age	0.04	0.03	1.68	.092
Learner type × Age	0.01	0.03	0.32	.753
Clause × Gap × Learner type	2.10	0.59	3.56	$< .001$
Clause × Gap × Age	0.40	0.05	7.77	$< .001$
Clause × Learner type × Age	−0.13	0.05	−2.51	.012
Gap × Learner type × Age	0.24	0.05	4.57	$< .001$
Clause × Gap × Learner type × Age	−0.62	0.10	−6.05	$< .001$

G Results of the linear mixed-effects regression for the child data with the Age factor added

Table 4 summarizes the results of the linear mixed-effects regression constructed for the child data, including Clause, Gap, and Age as fixed effects, as well as Item as a random effect.

Table 4: Linear mixed-effects regression results for the child data with the Age factor added

Predictor	*b*	*SE*	*t*	*p*
(Intercept)	0.05	0.17	0.29	.769
Clause	−0.25	0.32	−0.78	.438
Gap	0.38	0.33	1.17	.244
Age	−0.01	0.03	−0.35	.730
Clause × Gap	−2.52	0.65	−3.90	$< .001$
Clause × Age	0.07	0.06	1.21	.227
Gap x Age	−0.08	0.06	−1.50	.134
Clause × Gap × Age	0.79	0.11	7.05	$< .001$

H Results of the linear mixed-effects regression for the baby language model data with the Age factor added

Table 5 summarizes the results of the linear mixed-effects regression constructed for the baby LM data, including Clause, Gap, and Age as fixed effects, as well as Item as a random effect.

Table 5: Linear mixed-effects regression results for the baby language model data with the Age factor added

Predictor	*b*	*SE*	*t*	*p*
(Intercept)	0.00	0.12	0.00	1.000
Clause	1.07	0.09	12.41	$< .001$
Gap	0.16	0.11	1.52	.132
Age	0.00	0.01	0.00	1.000
Clause × Gap	−0.01	0.17	−0.06	.952
Clause × Age	−0.07	0.02	−4.11	$< .001$
Gap x Age	0.16	0.02	9.73	$< .001$
Clause × Gap × Age	0.09	0.03	2.67	.008

I Results of the linear mixed-effects regression for the baby language model data with the Model architecture factor added

Table 6 summarizes the results of the linear mixed-effects regression constructed for the baby LM data, including Clause, Gap, and Model architecture as fixed effects, as well as Item as a random effect.

Table 6: Linear mixed-effects regression results for the baby language model data with the Model architecture factor added

Predictor	*b*	*SE*	*t*	*p*
(Intercept)	0.00	0.11	0.00	1.000
Clause	0.73	0.02	30.75	$< .001$
Gap	0.97	0.07	14.08	$< .001$
Model architecture	0.00	0.02	0.00	1.000
Clause × Gap	0.43	0.05	9.13	$< .001$
Clause × Model architecture	0.19	0.05	4.08	$< .001$
Gap × Model architecture	−0.32	0.05	−6.69	$< .001$
Clause × Gap × Model architecture	−0.19	0.09	−1.97	.049

Does Typological Distance Affect Cross-lingual Transfer?

Pengfei Ren
School of Artificial Intelligence, Nanjing University
pengfeiren@smail.nju.edu.cn
github.com/Nonepf/multilang-babylm

Abstract

The Contrastive-Analysis Hypothesis (CAH) (Lado, 1957) proposes that similarities between a learner's first language (L1) and second language (L2) can facilitate L2 learning, while differences can make it more difficult. We ask a related computational question: under a limited, developmentally motivated data budget, does a coarse near/far relation between L1 and L2 predict grammatical performance when a language model is trained first on L1 and then on L2? We train GPT-2 Small models from scratch on English, Dutch, or Chinese with a fixed Qwen2.5 tokenizer, and continue training them on the corresponding L2 targets. Across five seeds, we compare zero-shot starting scores, terminal L2 scores, Transfer Increment (TI; terminal minus zero-shot), and Transfer Benefit (TB; terminal minus a same-budget target-only baseline). For English, NL→EN and ZH→EN have nearly identical TI. For Dutch, the near-L1 condition has a larger TI, but most of this difference comes from the zero-shot starting scores, while the terminal scores differ by only 0.9 percentage points. Headroom-normalized TI gives the same target-dependent picture. Across the tested pairs, the near/far contrast therefore does not produce a uniform, target-independent advantage for the near L1.

1 Introduction

This work is inspired by human second-language (L2) learning. A long-standing idea in second-language acquisition is that learning an L2 can be influenced by a learner's first language (L1). In the Contrastive-Analysis Hypothesis (CAH), Lado (1957) proposed that linguistic features shared by the L1 and L2 can facilitate learning, whereas differences can create difficulties. Later SLA research refined this strong formulation and developed more nuanced accounts of cross-linguistic influence (Odlin, 1989; Ringbom, 2006; Jarvis and Pavlenko, 2008).

In recent years, a related question has also been studied in NLP, with the learner replaced by a pretrained language model. Work on multilingual models has found that zero-shot performance varies across source–target language pairs and is often related to their linguistic similarity (Pires et al., 2019; Lauscher et al., 2020). Much of this work studies models jointly pretrained on many languages. Another line of research instead considers sequential exposure, where a model is trained first on one language and then on another (Oba et al., 2023; Yadavalli et al., 2023; Gogoulou et al., 2024). We follow this sequential L1→L2 setting under a restricted data budget.

We use the English, Dutch, and Chinese corpora from BabyBabelLM (Jumelet et al., 2026). Unlike typical Web corpora, BabyBabelLM is curated to better approximate the linguistic input available from birth and is built around a bounded data budget. We use these two properties to make the training setting more developmentally motivated. The analogy concerns the data and exposure budget rather than the full learning process: the models remain Transformers trained with next-token prediction, a fixed subword vocabulary, and standard neural optimization.

English and Dutch form our coarse "near" condition because both are West Germanic languages written in Latin script. Chinese paired with either Germanic language forms the coarse "far" condition. These pairs differ not only in genealogy, but also in script, lexical and token overlap, and a range of structural properties. Following the concerns about how typological diversity is operationalized in NLP (Ploeger et al., 2026), we use *near* and *far* as categorical condition labels rather than as numerical measurements of language distance.

Using these datasets, we train GPT-2 models (Radford et al., 2019) from scratch with a fixed Qwen2.5 tokenizer (Qwen et al., 2025). We ask: **Within this three-language design, does**

Proceedings of the Second BabyLM Workshop, pages 65–74
October 29, 2026 ©2026 Association for Computational Linguistics

the coarse near/far L1 contrast consistently predict L2 grammatical outcomes during sequential training?

The pattern differs across targets and metrics. For English, the near and far conditions show almost identical Phase 2 increments. For Dutch, the near condition has a larger increment, but much of that difference comes from its lower zero-shot starting score; the terminal scores differ by only 0.9 percentage points (pp). Across the two targets, the experiments do not show a consistent near-L1 advantage.

2 Related Work

2.1 Cross-lingual Transfer and Typological Distance

Cross-lingual transfer has been widely studied in massively multilingual pretrained models such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020). Their zero-shot performance varies substantially across source–target language pairs, and linguistic similarity is one factor associated with this variation. For example, Pires et al. (2019) found stronger mBERT transfer between typologically similar languages and identified shared word-order features as one useful predictor. Lauscher et al. (2020) further examined how linguistic proximity and other factors shape zero-shot transfer.

Typological "distance" can refer to different properties. Languages vary in genealogy, syntax, morphology, script, and lexicon, while observed model behavior can also depend on corpora and tokenization. Ploeger et al. (2024) show that typological diversity has been operationalized inconsistently in NLP, and Ploeger et al. (2026) propose a principled framework for selecting and evaluating typologically diverse language samples.

For each target, the near and far conditions use the same L2 corpus and evaluation suite, while their L1 sources differ in script, lexical and token overlap, corpus statistics, and Phase 1 exposure. The comparison therefore asks whether this coarse near/far grouping predicts the expected ordering of L2 outcomes within a fixed target, rather than assigning a numerical effect size to typological distance itself.

2.2 Computational SLA and Sequential Transfer

Computational models of SLA long predate Transformers. Gasser (1990), for example, trained a connectionist network first on L1 patterns and then on L2 patterns to model constituent-order transfer, while later work varied the amount and onset of L1/L2 exposure (Matusevych et al., 2017). These studies make the assumed learning process explicit. A related caution is raised by Cotterell (2026) in the context of surprisal theory: cognitive interpretations require independently motivated constraints on the model rather than behavioral fit alone. In this paper, SLA motivates the question and training order; GPT-2 itself is not intended as a literal model of a child learner.

Cross-lingual transfer does not require joint multilingual pretraining. Artetxe et al. (2020) showed that monolingual representations can support transfer to a new language even without shared vocabulary or joint training. More directly related to our setup, several studies train on an L1 before introducing an L2. Oba et al. (2023) found that L1 pretraining can accelerate later linguistic generalization depending on the L1 and transfer configuration. Yadavalli et al. (2023) observed more negative transfer between more distant language families in an age-ordered child-directed-speech setting. From a continual-learning perspective, Gogoulou et al. (2024) found that forward and backward transfer depend on language order and language characteristics, including syntactic similarity and possible contamination. We use the same broad sequential idea, with two fixed-target near/far comparisons, target-only references, and five random seeds.

Our focus is narrower: under BabyLM-scale data constraints, does the coarse near/far contrast predict L2 grammatical outcomes when the target corpus and evaluation are fixed? We report performance before L2 exposure, improvement during L2 training, and terminal performance separately so that the starting state, the Phase 2 change, and the final outcome remain distinguishable.

2.3 Data-Constrained Language Modeling

BabyLM studies language modeling under restricted data budgets (Choshen et al., 2026). BabyBabelLM extends this setting to 45 languages and curates corpora intended to approximate linguistic input observed from birth (Jumelet et al., 2026). Comparing data budgets across languages requires

a common reference because writing systems and encodings differ. Byte Premium is defined following Arnett et al. (2024); the Dutch and Chinese coefficients used here are the BabyLM 2026 values reported by (Choshen et al., 2026). We use their published values as scaling factors for our per-language extraction, described below.

3 Methodology

3.1 Model Architecture

We use GPT-2 Small (Radford et al., 2019), with 12 Transformer layers, a hidden size of 768, 12 attention heads, and a 512-token context window. All model weights are initialized randomly.

Our preliminary diagnostics showed that the standard GPT-2 byte-level tokenizer fragments Chinese heavily, producing 2.17 tokens per character. We therefore use the Qwen2.5-0.5B tokenizer (Qwen et al., 2025). With this tokenizer, the measured token-to-unit ratios are 1.20 for English words, 1.70 for Dutch words, and 0.70 for Chinese characters (Appendix D). Its vocabulary contains 151,665 entries. Because GPT-2 ties the input and output embeddings, the embedding matrix contributes 116.5M parameters, or about 58% of the resulting 201.9M-parameter model.

The tokenizer, vocabulary, and embedding dimensionality are fixed across all languages, conditions, and phases. A model trained only on L1 therefore has token *IDs* that can represent L2 strings, but all embedding rows begin from random initialization and are updated only through the model's own training data; no pretrained Qwen embeddings or lexical meanings are imported. Fixing the tokenizer keeps the model shape identical across conditions and allows Phase 1 checkpoints to be continued directly in Phase 2. Pairwise script and token overlap can still vary under this shared vocabulary, so the tokenizer is treated as a controlled part of the setup rather than as an explanation of the transfer pattern.

3.2 Data Preparation

We use `BabyLM-2026-Strict` for English, `babylm-nld` for Dutch, and `babylm-zho` for Chinese from BabyBabelLM (Jumelet et al., 2026). English and Dutch are counted in whitespace-delimited words. Chinese is counted in characters because the corpus does not have a universal whitespace word boundary. Table 1 gives the extraction used in our runs. For each phase, we targeted 50M "effective words" by dividing this value by the published language-specific Byte Premium (Arnett et al., 2024).

Character counting avoids choosing a particular Chinese word-segmentation scheme, but it also means that the counting unit differs from English and Dutch. In addition, Byte Premium is defined over bytes rather than over a mixture of word and character counts. We keep the extraction used in the original runs; the resulting byte and update counts are reported in Appendix D.

3.3 Training Pipeline

All runs use the same training configuration: AdamW, a cosine learning-rate schedule with a peak of 5×10^{-4}, an effective batch size of 65,536 tokens per update, and BF16 precision. These GPT-2/BabyLM-style settings were chosen before the comparison rather than through a condition-specific hyperparameter search. Appendix A lists the full configuration and update counts.

Training has two phases:

- **Phase 1 (L1 pretraining):** Each model is trained from random initialization on English, Dutch, or Chinese for one pass over the allocated corpus (Table 1).
- **Phase 2 (L1→L2 continuation):** We load the Phase 1 model weights and continue training all parameters on the L2 corpus. The optimizer state is reset: each Phase 2 run starts with a new AdamW optimizer and a fresh cosine schedule peaking at 5×10^{-4}. No replay or continual-learning regularizer is used.

The extracted corpora contain different numbers of tokens, giving 916 Phase 1 updates for EN, 1,234 for NL, and 540 for ZH. The English and Dutch target phases contain 916 and 1,234 updates, respectively. Table 1 and Appendix A make these source-side exposure differences explicit.

3.4 Evaluation

We evaluate grammatical minimal-pair accuracy using the BabyLM `lm-eval` configurations (Choshen et al., 2026). The benchmarks are BLiMP for English (Warstadt et al., 2020), BLiMP-NL for Dutch (Suijkerbuijk et al., 2025), and ZhoBLiMP for Chinese (Liu et al., 2026). A minimal pair is counted as correct when the model assigns higher log-likelihood to the grammatical sentence. Chance accuracy is 50%.

Language	Byte Premium	Counting Units per Phase	Qwen2.5 Tokens
English	1.0000	50.0M words	∼60.0M
Dutch	1.0516	47.5M words	∼80.8M
Chinese	0.9894	50.5M chars	∼35.4M

Table 1: Data allocation used in the experiments. English and Dutch are measured in whitespace-delimited words, while Chinese is measured in characters. Byte Premium values follow Arnett et al. (2024).

Language	Configuration	Benchmark
English	`blimp_babylm_filtered`	BLiMP%
Dutch	`blimp_nl`	BLiMP-NL%
Chinese	`zhoblimp`	ZhoBLiMP%

Table 2: Minimal-pair evaluation suites used in the experiments. The three suites cover different languages and grammatical phenomena, so comparisons are made within each target language.

For a fixed source X, target Y, and seed, let A_{ZS} be the Phase 1 model's accuracy on Y before L2 exposure, A_{term} the corresponding Phase 2 terminal accuracy, and A_{target} the same-seed target-only model's accuracy after one target-language budget. We report three derived quantities:

$$\begin{aligned} \text{TI} &= A_{\text{term}} - A_{\text{ZS}}, \\ \text{nTI} &= \frac{A_{\text{term}} - A_{\text{ZS}}}{1 - A_{\text{ZS}}}, \\ \text{TB} &= A_{\text{term}} - A_{\text{target}}. \end{aligned} \tag{1}$$

TI measures how much a model improves during Phase 2 from its L1-derived starting point. Because the available room for improvement depends on the zero-shot score, we also report normalized TI (nTI), the fraction of the remaining accuracy headroom recovered during Phase 2. TB compares the terminal sequential model with a target-only model trained on the same L2 budget. Thus, TI describes change during L2 training, whereas TB describes the terminal advantage over the target-only reference. The sequential model has also received Phase 1 training, so TB is a terminal reference comparison rather than a compute-matched effect estimate.

All near/far comparisons are made within the *same target suite*. English BLiMP and Dutch BLiMP-NL probe different sets of phenomena, so their absolute scores and gain magnitudes are not placed on a common scale. Appendix C gives the per-seed derivations, and Appendix B reports all 35 evaluation rows.

3.5 Experimental Design

We use five random seeds (42, 123, 456, 789, 1024), giving 35 training runs across seven conditions:

- **Phase 1 L1-only Baselines** (3 conditions × 5 seeds = 15 runs): EN-only, NL-only, ZH-only.
- **Phase 2 Transfer Conditions** (4 conditions × 5 seeds = 20 runs): NL→EN (main test, near-L1), ZH→EN (main test, far-L1), EN→NL (auxiliary, near-L1), ZH→NL (auxiliary, far-L1).

We report the mean and sample standard deviation across the five seeds, together with the full per-seed results in the appendix. Small between-condition differences are interpreted descriptively.

4 Results

4.1 Phase 1: Same-language and Zero-shot Scores

Table 3 shows the complete Phase 1 evaluation matrix. The diagonal entries are same-language references, while the off-diagonal entries are zero-shot scores before any L2 exposure. All three diagonal means are above chance. Because each column uses a different benchmark suite, the useful comparisons are within columns rather than across target languages.

For the English target, Dutch pretraining gives a zero-shot score of 54.6%, compared with 50.5% after Chinese pretraining, a difference of 4.1 pp. The near condition therefore begins English training from a higher grammatical score. This starting difference may reflect several properties that co-vary between the two source languages, including linguistic structure, script and token overlap, corpus content, and Phase 1 exposure.

In the reverse direction, English pretraining gives 49.4% on Dutch. This should not be read as a direct reversal of the Dutch→English result, because the two values come from different target suites and different training conditions. The directional comparison is therefore less informative than the fixed-target near/far comparisons used below.

Phase 1 Language	English BLiMP	Dutch BLiMP-NL	Chinese ZhoBLiMP
English	**60.1%**	49.4%	48.1%
Dutch	54.6%	**71.0%**	50.6%
Chinese	50.5%	51.9%	**63.2%**

Table 3: Complete Phase 1 evaluation matrix (means over five seeds). Bold diagonal cells are same-language references and off-diagonal cells are zero-shot evaluations. Comparisons are made within columns, where the target language and benchmark suite are fixed.

Target	Source	Label	Zero-shot (%)	Terminal (%)	TI (pp)	nTI (%)	TB (pp)
English	Dutch	near	54.6 ± 0.7	64.9 ± 0.8	10.4 ± 0.9	22.8 ± 1.8	4.8 ± 1.2
English	Chinese	far	50.5 ± 2.5	61.3 ± 1.2	10.8 ± 3.5	21.6 ± 6.0	1.1 ± 0.5
Dutch	English	near	49.4 ± 1.8	73.1 ± 0.5	23.7 ± 1.5	46.8 ± 1.5	2.0 ± 0.7
Dutch	Chinese	far	51.9 ± 1.6	72.2 ± 0.7	20.3 ± 1.8	42.2 ± 2.6	1.2 ± 1.3

Table 4: Target-wise transfer summary across five seeds (mean $\pm$ sample SD). TI and TB are percentage-point differences, while nTI is the percentage of available accuracy headroom recovered during Phase 2. Near/far comparisons are made within each target block.

4.2 Phase 2: Target-wise Comparisons

Table 4 summarizes the Phase 2 results as means and standard deviations; the per-seed values are given in Appendices B and C. For English, NL→EN and ZH→EN have almost the same TI (10.4 vs. 10.8 pp). Their normalized TI is also close (22.8% vs. 21.6% of the available headroom). The terminal English score is nevertheless 3.7 pp higher after Dutch pretraining, matching the 3.7 pp near–far difference in TB. In other words, the English terminal gap mainly reflects the different zero-shot starting points rather than a larger Phase 2 increment for the near condition.

For Dutch, the near condition has a 3.4 pp larger TI and a 4.6 pp larger nTI, but its terminal advantage is only 0.9 pp. The reason is visible in the starting scores: English pretraining begins 2.5 pp below Chinese pretraining on BLiMP-NL. The 3.4 pp TI contrast therefore combines a small terminal advantage with a larger zero-shot disadvantage in the opposite direction. This decomposition explains why the Dutch TI difference looks larger than the difference between the final models.

TB gives a complementary view of the same runs by comparing each terminal sequential model with target-only training. All four mean TB values are positive. Since the sequential models also received a source-language phase, TB is best read as a terminal reference comparison under this training setup. For the near/far comparison, the English terminal/TB contrast is larger than the Dutch contrast, whereas TI is nearly tied for English and more separated for Dutch. The ordering therefore depends on which aspect of transfer is being measured.

5 Discussion

The main result is that the near/far pattern changes with both the target language and the quantity being measured. TI shows essentially no near-L1 advantage for English and a modest advantage for Dutch. Terminal scores and TB show the opposite emphasis: a clearer near advantage for English and only a small one for Dutch. Normalizing TI by the available headroom leaves this qualitative pattern unchanged. The zero-shot score, Phase 2 increment, and terminal score therefore describe different parts of the transfer process and are most informative when read together.

This result is less uniform than the broad distance pattern reported by Yadavalli et al. (2023). At the same time, it is consistent with the broader observation in Oba et al. (2023) and Gogoulou et al. (2024) that sequential transfer depends on the source language, target language, training order, and experimental setup. These studies use different architectures, language sets, corpora, vocabulary treatments, and evaluations, so the comparison is conceptual rather than a direct replication. In our setting, coarse linguistic proximity alone is not enough to predict the full transfer pattern.

Table 6 summarizes what is held fixed and what varies with the source language. The target corpus, target-side updates, evaluation suite, architecture, and tokenizer are controlled within each comparison. Other source-side properties—including genealogy, script, token overlap, corpus composition, and Phase 1 exposure—move together with the near/far label. The fixed tokenizer therefore helps keep model structure constant, but does not by itself identify why Dutch and Chinese pretraining

Target	Δ Zero-shot	Δ Terminal	Δ TI	Δ nTI	Δ TB
English	+4.1 pp	+3.7 pp	−0.4 pp	+1.2 pp	+3.7 pp
Dutch	−2.5 pp	+0.9 pp	+3.4 pp	+4.6 pp	+0.9 pp

Table 5: Near-minus-far contrasts within each target. The identity $\Delta\mathrm{TI} = \Delta\mathrm{Terminal} - \Delta\mathrm{Zero\text{-}shot}$ separates the contribution of the starting score from the terminal difference. ΔTB equals ΔTerminal because both conditions use the same target-only reference.

Factor	Treatment	Role in comparison
Target corpus and suite	Fixed within target	Near/far conditions for one target receive the same L2 data, updates, and evaluation.
Architecture and tokenizer	Fixed globally	Model shape and token inventory do not vary, but realized token overlap can still differ by pair.
Genealogy and script	Vary with source	These properties co-vary with the near/far label.
Token/lexical overlap	Varies by pair	Shared tokens and lexicon may contribute independently of broader typological similarity.
Phase 1 exposure	Varies by source	Source conditions contain different token totals and 540–1,234 updates.
Corpus–benchmark overlap	Unmeasured	Any overlap would contribute to observed zero-shot variation.

Table 6: Experimental controls and source-side factors relevant to the near/far comparison.

lead to different starting or terminal scores.

6 Conclusion

We trained 35 models across seven conditions and five seeds to test whether a coarse near/far L1 contrast consistently predicts grammatical outcomes after sequential L1→L2 training. Across the two target languages, it does not produce a single consistent pattern. For English, near and far conditions have almost identical Phase 2 increments even though their terminal scores differ. For Dutch, the near condition has a larger increment, while the terminal difference is small. Headroom normalization preserves the same qualitative result.

These experiments therefore provide evidence against a uniform near-L1 advantage in this particular sequential training setup, rather than a general verdict on the CAH in human learners. Extending the question will require a broader language sample, more explicit typological sampling, better matched training budgets, and direct measurements or ablations of factors such as vocabulary overlap and script.

Limitations

Language sample and operationalization. The experiments include English, Dutch, and Chinese, with near/far used as a categorical grouping rather than a measured scalar distance. In this three-language design, genealogy, script, likely lexical/token overlap, corpus composition, and source update count co-vary with that grouping. A broader language sample with explicit distance measures would be needed to estimate a general relationship between typological distance and transfer.

Tokenizer and overlap. The fixed Qwen2.5 tokenizer keeps the vocabulary dimensionality identical across conditions, while also giving every model access to the same multilingual token inventory. The tied embedding matrix accounts for roughly 58% of model parameters. Appendix D reports token-to-unit ratios, but these ratios measure tokenization efficiency rather than corpus-level vocabulary overlap. Direct overlap statistics or tokenizer/script ablations would be needed to separate these mechanisms.

Chinese counting and Byte Premium. English and Dutch are counted in whitespace-delimited words, whereas Chinese is counted in characters. Our allocation then applies the published Byte Premium to these mixed units rather than matching UTF-8 bytes directly as defined by Arnett et al. (2024). Under the rough byte assumptions in Appendix D, direct byte matching would correspond to about 82.5M Chinese characters rather than the 50.5M used here. The Chinese conditions therefore differ in both exposure and update count from a strictly byte-matched design.

Optimization and compute. Phase 1 contains 916 updates for EN, 1,234 for NL, and 540 for ZH. The experiments therefore follow the intended effective-unit allocation rather than matching tokens, updates, bytes, or total compute across source languages. Phase 2 resets AdamW and the cosine schedule for every condition, with a new peak learning rate of 5×10^{-4}. This reset is consistent within each target comparison, although uninterrupted op-

timization could produce a different learning trajectory. The shared hyperparameters were not selected through a sweep.

Evaluation and metrics. BLiMP, BLiMP-NL, and ZhoBLiMP differ in coverage and difficulty. We therefore compare near/far conditions within a target language rather than comparing absolute scores or TI magnitudes across suites. nTI adjusts for available accuracy headroom but does not place the suites on a common scale. We also did not audit corpus–benchmark overlap, so cross-direction zero-shot differences are not interpreted as a direct measure of directional language-to-language transfer.

Statistical uncertainty. Each condition has five seeds, which gives only a coarse estimate of variability for small effects. ZH→EN in particular has a TI standard deviation of 3.5 pp. We therefore report mean±SD together with the full per-seed values and interpret small between-condition differences descriptively.

Acknowledgments

This work was carried out as an independent project during the first year of my undergraduate studies and was one of my first attempts to develop a research question into a complete experiment. I am grateful to the reviewers for their detailed feedback, which helped me improve the paper, and to the BabyLM community for providing the datasets, evaluation setting, and an opportunity to pursue this question.

Generative AI tools, including DeepSeek and ChatGPT, were used throughout the project to discuss and critique the research question and experimental design, assist with training and analysis code, and translate, rewrite, and polish portions of the manuscript based on written drafts by. All AI-assisted suggestions, code, analyses, and text were critically reviewed and verified by me. I made the final research and experimental decisions, conducted the experiments and evaluation, verified the reported results and citations.

References

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Ryan Cotterell. 2026. Surprisal theory is tautological (without rational grounding). *Preprint*, arXiv:2607.21574.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael Gasser. 1990. Connectionism and universals of second language acquisition. *Studies in Second Language Acquisition*, 12(2):179–199.

Evangelia Gogoulou, Timothée Lesort, Magnus Boman, and Joakim Nivre. 2024. Continual learning under language shift. In *Text, Speech, and Dialogue: 27th International Conference, TSD 2024, Brno, Czech Republic, September 9–13, 2024, Proceedings, Part I*, page 71–84, Berlin, Heidelberg. Springer-Verlag.

Scott Jarvis and Aneta Pavlenko. 2008. *Crosslinguistic Influence in Language and Cognition*, 1st edition. Routledge, New York.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329. Association for Computational Linguistics.

Robert Lado. 1957. *Linguistics across cultures: Applied linguistics for language teachers*. University of Michigan Press.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in Chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Yevgen Matusevych, Afra Alishahi, and Ad Backus. 2017. The impact of first and second language exposure on learning second language constructions. *Bilingualism: Language and Cognition*, 20(1):128–149.

Miyu Oba, Tatsuki Kuribayashi, Hiroki Ouchi, and Taro Watanabe. 2023. Second language acquisition of neural language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13557–13572, Toronto, Canada. Association for Computational Linguistics.

Terence Odlin. 1989. *Language Transfer*. Cambridge Applied Linguistics. Cambridge University Press.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Esther Ploeger, Wessel Poelman, Miryam de Lhoneux, and Johannes Bjerva. 2024. What is "typological diversity" in NLP? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5681–5700, Miami, Florida, USA. Association for Computational Linguistics.

Esther Ploeger, Wessel Poelman, Andreas Holck Høeg-Petersen, Anders Schlichtkrull, Miryam de Lhoneux, and Johannes Bjerva. 2026. A principled framework for evaluating on typologically diverse languages. *Computational Linguistics*, 52(1):237–269.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, and 24 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Håkan Ringbom. 2006. *Cross-linguistic Similarity in Foreign Language Learning*. Multilingual Matters, Bristol and Blue Ridge Summit.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Aditya Yadavalli, Alekhya Yadavalli, and Vera Tobin. 2023. SLABERT talk pretty one day: Modeling second language acquisition with BERT. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11763–11777, Toronto, Canada. Association for Computational Linguistics.

A Hyperparameters

The same configuration was used for all conditions. These GPT-2/BabyLM-style settings were chosen before the comparison and were not tuned separately for individual conditions.

Hyperparameter	Value
Architecture	GPT-2 Small (12 layers, 768 hidden, 12 heads)
Tokenizer	Qwen2.5-0.5B (fixed vocabulary: 151,665 entries)
Model Parameters	201,927,936 (all trainable)
Tied Token Embedding	116,478,720 parameters (57.7% of total)
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999, \varepsilon =$ 1e-8)
Peak Learning Rate	5×10^{-4}, cosine decay
Phase 2 Learning Rate	Fresh cosine schedule from 5×10^{-4} (optimizer state not loaded)
Weight Decay	0.01 (excluding bias and LayerNorm)
Gradient Clipping	max norm = 1.0
Effective Batch Size	65,536 tokens/step (128 sequences $\times$ 512 tokens)
Warmup Steps	100
Precision	BF16 mixed precision
Training Steps (Phase 1)	EN: 916, NL: 1,234, ZH: 540
Training Steps (Phase 2)	EN target: 916, NL target: 1,234
Random Seeds	42, 123, 456, 789, 1024

Table 7: Complete shared hyperparameter configuration. Phase 2 loads model weights but not optimizer or scheduler state.

B Complete BLiMP Evaluation Results

Table 8 reports the evaluation accuracy of all 35 models on the three language-specific suites. Each entry is the mean accuracy over the subtasks exposed by that suite. Phase 1 models (P1) are the L1-only models, and Phase 2 models (P2) are the sequential transfer models.

Phase	Condition	Seed	EN BLiMP	NL BLiMP	ZH BLiMP
P1	en	42	60.83%	48.78%	47.55%
P1	en	123	58.91%	47.26%	48.93%
P1	en	456	60.91%	51.60%	48.39%
P1	en	789	60.59%	48.63%	50.45%
P1	en	1024	59.47%	50.74%	45.27%
P1	nl	42	55.08%	70.52%	51.39%
P1	nl	123	53.81%	71.13%	51.73%
P1	nl	456	55.47%	71.32%	48.79%
P1	nl	789	54.18%	70.29%	51.67%
P1	nl	1024	54.25%	71.90%	49.60%
P1	zh	42	48.72%	50.32%	62.10%
P1	zh	123	51.70%	51.96%	63.24%
P1	zh	456	50.58%	54.37%	62.81%
P1	zh	789	47.40%	50.91%	63.35%
P1	zh	1024	53.90%	52.10%	64.45%
P2	nl2en	42	65.17%	55.48%	49.22%
P2	nl2en	123	64.19%	55.41%	54.85%
P2	nl2en	456	65.05%	52.69%	52.24%
P2	nl2en	789	64.08%	54.00%	50.94%
P2	nl2en	1024	66.15%	56.72%	50.75%
P2	zh2en	42	61.69%	49.70%	53.62%
P2	zh2en	123	59.45%	50.38%	53.68%
P2	zh2en	456	62.11%	48.94%	56.22%
P2	zh2en	789	62.44%	48.66%	54.01%
P2	zh2en	1024	60.68%	48.36%	50.84%
P2	en2nl	42	58.69%	72.30%	47.64%
P2	en2nl	123	58.45%	72.88%	50.78%
P2	en2nl	456	57.71%	73.58%	49.44%
P2	en2nl	789	57.00%	73.37%	52.60%
P2	en2nl	1024	57.46%	73.23%	48.00%
P2	zh2nl	42	54.05%	72.14%	52.64%
P2	zh2nl	123	53.45%	72.50%	54.56%
P2	zh2nl	456	54.21%	72.31%	50.90%
P2	zh2nl	789	53.28%	73.07%	52.29%
P2	zh2nl	1024	53.13%	71.07%	53.56%

Table 8: Complete BLiMP evaluation scores for all 35 models (7 conditions $\times$ 5 seeds) across all three language test suites. P1 = Phase 1 monolingual; P2 = Phase 2 transfer.

C Zero-Shot and Transfer Increment per Seed

Table 9 gives the per-seed zero-shot scores used in Table 3. Table 10 shows the corresponding TI values, and Table 11 gives the headroom-normalized TI for each seed.

Seed	NL→EN ZS	ZH→EN ZS	EN→NL ZS	ZH→NL ZS
42	55.08%	48.72%	48.78%	50.32%
123	53.81%	51.70%	47.26%	51.96%
456	55.47%	50.58%	51.60%	54.37%
789	54.18%	47.40%	48.63%	50.91%
1024	54.25%	53.90%	50.74%	52.10%
Mean	**54.56%**	**50.46%**	**49.40%**	**51.93%**

Table 9: Zero-shot cross-lingual BLiMP scores per seed. ZS = Phase 1 L1-only model evaluated on an unseen L2 BLiMP suite.

Seed	TI(NL→EN)	TI(ZH→EN)	Δ
42	+10.09	+12.97	−2.88
123	+10.38	+7.75	+2.63
456	+9.58	+11.53	−1.95
789	+9.90	+15.04	−5.14
1024	+11.90	+6.78	+5.12
Mean	**+10.37**	**+10.81**	**−0.44**
± SD	±0.90	±3.49	—

Seed	TI(EN→NL)	TI(ZH→NL)	Δ
42	+23.52	+21.82	+1.70
123	+25.62	+20.54	+5.08
456	+21.98	+17.94	+4.04
789	+24.74	+22.16	+2.58
1024	+22.49	+18.97	+3.52
Mean	**+23.67**	**+20.29**	**+3.38**
± SD	±1.52	±1.81	—

Table 10: Transfer Increment per seed for both target languages (all values in percentage points). Δ = near-L1 TI − far-L1 TI.

Seed	nTI(NL→EN)	nTI(ZH→EN)	nTI(EN→NL)	nTI(ZH→NL)
42	22.46%	25.29%	45.92%	43.92%
123	22.47%	16.05%	48.58%	42.76%
456	21.51%	23.33%	45.41%	39.32%
789	21.61%	28.59%	48.16%	45.14%
1024	26.01%	14.71%	45.66%	39.60%
Mean	**22.81%**	**21.59%**	**46.75%**	**42.15%**
± SD	±1.84	±6.00	±1.50	±2.60

Table 11: Headroom-normalized Transfer Increment, $\mathrm{nTI} = (A_{\mathrm{term}} - A_{\mathrm{ZS}})/(1 - A_{\mathrm{ZS}})$, computed per seed.

D Token-to-Word Ratio Analysis

This appendix reports the Token-to-Word Ratio (TWR) measurements from our preliminary data diagnostics. These measurements motivated the use of the Qwen2.5 tokenizer instead of the standard GPT-2 BPE tokenizer.

Language	Counting Unit	GPT-2 TWR	Qwen2.5 TWR
English	words (whitespace)	1.21	1.20
Dutch	words (whitespace)	2.16	1.70
Chinese	characters	2.17	0.70
Chinese	words (jieba)	5.05	2.87

Table 12: TWR comparison between the GPT-2 BPE tokenizer and Qwen2.5-0.5B. TWR is the number of tokens produced per language-specific counting unit; lower values indicate less fragmentation. For Chinese, both character-level and jieba-word-level values are shown.

With GPT-2 BPE, Chinese has a TWR of 2.17 per character and 5.05 per jieba-segmented word. This corresponds to roughly 2–3 byte-level tokens per Chinese character and reduces a 512-token context to about 236 characters. Qwen2.5 lowers the character-level TWR to 0.70 (2.87 per jieba word), giving an effective context of roughly 731 characters and allowing multi-character subword units to remain intact more often. This reduction in fragmentation was the main reason for choosing Qwen2.5 despite increasing the model size from about 124M to 202M parameters.

Table 13 shows the resulting token, byte, and update counts at the 50M effective-word scale.

Language	Counting Units	Qwen2.5 Tokens	Est. UTF-8 Bytes	Training Steps
English	50.0M words	∼60.0M	∼250 MB	916
Dutch	47.5M words	∼80.8M	∼238 MB	1,234
Chinese	50.5M chars	∼35.4M	∼152 MB	540

Table 13: Estimated training budgets per phase. Steps assume effective batch size 65,536 tokens/step. UTF-8 byte estimates use ∼5 bytes/word (EN, NL) and ∼3 bytes/character (ZH).

Under the rough byte approximation used here, direct byte-level matching would allocate about 82.5M Chinese characters per phase (247 MB), rather than the 50.5M used in the experiments. The Chinese conditions therefore contain fewer estimated bytes and fewer updates than this alternative allocation. Because that difference affects both the L1 starting state and the later L2 trajectory, it does not translate into a simple one-direction correction of TI or TB.

What Survives the BabyLM Budget? A Controlled Study of Sequence Mixers and Generative Objectives

Filipe Laitenberger **Patrick Haller** **Jonas Golde** **Alan Akbik**
Humboldt-Universität zu Berlin
filipe.laitenberger@hu-berlin.de

Abstract

BabyLM is a pressure test for language-modeling assumptions: which ideas still work when the data budget is closer to a child's linguistic input than to web-scale pretraining? We test this question at 10M- and 100M-word budgets, holding the corpus, tokenizer, backbone size family, training schedule, and public evaluation wrapper fixed while varying two sequence mixers and six objectives. The retained scale sweep places autoregressive Transformers first in broad transfer at every tested scale and shows little gain beyond the 50M-parameter range. Corrected matched-base experiments give a more local picture: non-autoregressive systems approach the leaders on individual grammar, supplement, and entity metrics and lead controlled-probe slices, but no one system combines these gains. Paired candidate-ranking and generation tests expose the interface gap. Masked diffusion can exceed autoregression on WikiText completion or infilling, yet no tested decoder wins across LAMBADA and the two WikiText tasks, categorical flow remains unreliable across its tested decoder grid. Holding the model fixed and varying only the scoring interface bounds how much of the ranking that interface explains: supplying right context on an identical target span helps all ten non-AR cells and no AR cell, and swapping candidate ranking for open generation multiplies the between-objective spread more than tenfold – but the corrections are smaller than the transfer gap they would have to close. Small-data conclusions therefore depend on the objective, sequence mixer, decoder, and scoring interface together.

1 Introduction

BabyLM is built around the comparison that Children acquire language from a tiny fraction of the text used to train modern language models, yet many claims about new objectives or architectures are tested only at much larger scale. The 2026 challenge keeps the 10M-word Strict-Small and 100M-word Strict English tracks, adds a multilingual track, and evaluates submitted models with shared zero-shot and fine-tuning tasks (Choshen et al., 2026). This lets us ask not only *what scores highest*, but which modeling assumptions remain useful when data, compute, and evaluation are all constrained.

Controlled comparison is already established BabyLM practice: previous entries contrast causal and masked training inside one model (Charpentier and Samuel, 2024), benchmark recurrent and linear-time mixers against attention (Haller et al., 2024, 2025), and re-audit elementary baselines under matched budgets (Păpuşoi and Nisioi, 2025). Our contribution is therefore the coverage of the resulting grid. We cross the sequence-mixer axis with the *generative objective* axis, and we include the diffusion and flow families that have reshaped generative modeling at scale but have not been audited at BabyLM budgets. A *sequence mixer* is the part of each model block that exchanges information among token positions. We cross two sequence mixers, standard Transformer attention (Vaswani et al., 2017) and Gated DeltaNet (Yang et al., 2025), with six objectives: left-to-right autoregression, BERT-style masked language modeling (Devlin et al., 2019), three absorbing-mask diffusion variants (Sahoo et al., 2024, Nie et al., 2025, and Lou et al., 2024), and a non-distilled categorical flow-map endpoint objective over Gaussian-to-one-hot interpolants (Roos et al., 2026 and Davis et al., 2026). All cells share the same corpus, tokenizer, model family, training schedule, and public BabyLM evaluation wrapper, and every objective–mixer pair receives its own learning-rate sweep so that no family competes at another family's optimizer setting.

The challenge fixes the corpus, so the design freedom that remains is how a block moves in-

Proceedings of the Second BabyLM Workshop, pages 75–92
October 29, 2026 ©2026 Association for Computational Linguistics

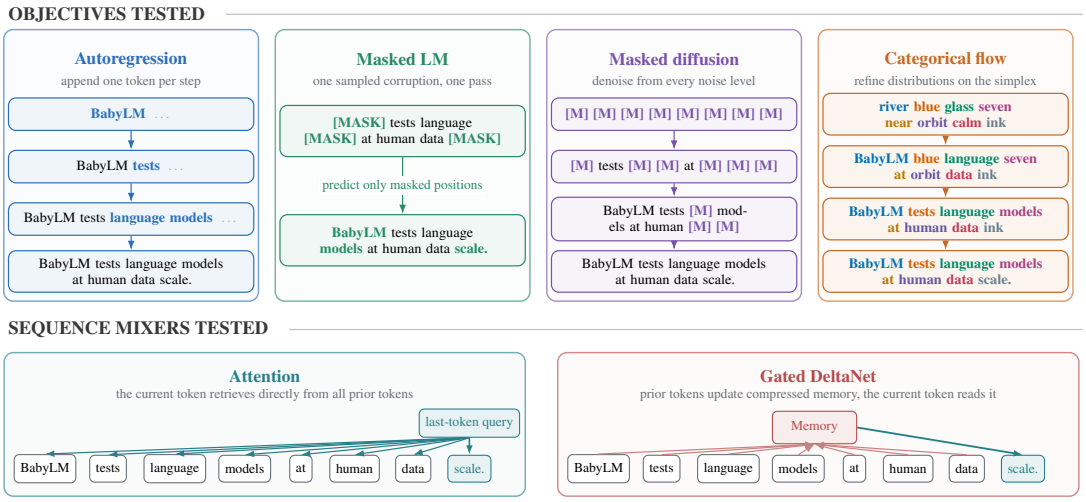


Figure 1: The two experimental axes. **Top:** AR appends tokens from left to right, MLM reconstructs one sampled mask pattern in a single pass, masked diffusion learns denoisers at all noise levels and chains them for generation, CatFlow moves continuous vocabulary states from Gaussian noise toward clean token endpoints (the displayed words are the most likely token per position). **Bottom:** the current position either attends directly to every preceding token or reads the compressed recurrent memory updated by Gated DeltaNet.

formation between positions and what prediction problem turns a sentence into gradient signal. Both axes carry an explicit small-data hypothesis. Gated DeltaNet represents the delta-rule family: its update overwrites the value stored at an addressed key rather than accumulating another association (Yang et al., 2024, 2025), and it is the mixer that has most recently replaced attention layers in a production-scale model (Merrill et al., 2026). Earlier BabyLM work found recurrent and linear-time mixers competitive at these budgets (Haller et al., 2024, 2025), Gated DeltaNet asks whether the current generation of that family still is. On the objective axis, masked diffusion is the natural "more supervision per sentence" hypothesis for scarce text: rather than one corruption level, each sentence yields a family of denoising problems indexed by noise (Austin et al., 2021; Sahoo et al., 2024). MDLM, LLaDA, and SEDD share that construction and differ in one decision each – loss weighting ($1/t$ versus uniform) and parameterization (token cross-entropy versus score entropy) – so together they separate the family from its implementation. MLM is their single-noise-level reference point, and categorical flow is the control that keeps bidirectional denoising but replaces the absorbing mask with a continuous Gaussian-to-one-hot path (Roos et al., 2026), separating what bidirectionality buys from what the mask construction buys.

Under the official public scorer, the autoregressive Transformer remains the strongest default: it wins broad transfer throughout the retained scale sweep, while the corrected matched-base comparison finds local non-AR strengths rather than one consistently strong alternative. Diffusion and categorical flow occasionally improve token overlap on bidirectional WikiText infilling, but neither advantage becomes reliable prompt-conditioned generation across tasks, and Gated DeltaNet helps particular objective–benchmark pairs rather than supplying a uniform advantage. We therefore read the grid as a BabyLM lesson rather than a blanket ranking of model classes: small-data inductive bias helps only when it is aligned with the scoring interface and downstream use. The challenge for future BabyLM systems is not to replace attention with recurrence or autoregression with diffusion, but to design the objective, mixer, sampler, and scorer as one matched system.

2 Background

BabyLM and sample-efficient pretraining. The BabyLM Challenge was introduced to make language-model pretraining experimentally accessible and cognitively motivated (Warstadt et al., 2023). The 2026 round keeps the Strict and Strict-Small tracks and frames the workshop around small-data training, evaluation, and architecture innovation (BabyLM Organizers, 2026). Previous strong BabyLM systems include hybrid causal and masked training recipes such as GPT-BERT (Charpentier and Samuel, 2024), as well as BabyLM-

specific recurrent and linear-time systems such as BabyHGRN and BLaLM (Haller et al., 2024, 2025). Several of these already report matched comparisons, and Păpuşoi and Nisioi (2025) audit elementary baselines under a shared evaluation pipeline. What distinguishes the present study is that each of those works holds one of our two axes fixed: GPT-BERT varies the objective inside an attention backbone, BabyHGRN and BLaLM vary the mixer under a fixed training objective, and the elementary-baseline audit varies the model family under a fixed scorer. We vary both axes jointly, over a generative-objective set that these studies predate, and we do not tune a single submission recipe.

We vary two independent choices: how the sequence mixer exchanges information among positions and what prediction problem supplies the training signal (the *objective*). Figure 1 summarizes the resulting two-by-six grid and the generation process associated with each objective.

2.1 Sequence Mixing in Language Models

Transformer attention. Softmax attention lets each token retrieve a content-weighted combination of the other token representations (Vaswani et al., 2017). We use standard multi-head self-attention: its context is causal for autoregressive training and bidirectional for all denoising objectives.

Linear attention. Instead of retaining every key–value pair, linear attention compresses the prefix into a fixed-size fast-weight state S_t, making the same computation expressible as a recurrent update (Katharopoulos et al., 2020). In its additive form, $S_t = S_{t-1} + v_t k_t^\top$, so writing a new association can overwrite an old one only indirectly. DeltaNet adds an error-correcting write,

$$S_t = S_{t-1} + \beta_t (v_t - S_{t-1} k_t) k_t^\top,$$

which changes the value stored at the addressed key rather than simply adding another value (Yang et al., 2024).

Gated DeltaNet. Gated DeltaNet additionally learns how much of the previous state to retain and how strongly to write at each step (Yang et al., 2025). OLMo Hybrid uses these layers at 7B scale in place of OLMo 3's sliding-window attention and reports better pretraining and mid-training evaluations (Merrill et al., 2026). We use Gated DeltaNet as one representative rather than sweep efficient mixers. It runs left to right for causal objectives, for bidirectional objectives we average forward and reverse passes. This asks whether compressed, explicitly updated memory helps when sequences are short (256 tokens) but linguistic data are scarce.

2.2 Language Modeling Objectives

Autoregression. AR turns a sequence into a fixed series of next-token decisions,

$$L_{\mathrm{AR}} = -\sum_{i=1}^{n} \log p_\theta(x_i \mid x_{<i}),$$

with a beginning-of-sequence symbol before x_1. Generation repeats exactly the operation used in training: predict one token from the prefix, append it, and continue (Radford et al., 2019). This direct match also provides an exact left-to-right sequence score.

Masked language modeling. MLM presents the model with a partly observed sequence and asks it to recover a random 15% of tokens from both left and right context (Devlin et al., 2019):

$$L_{\mathrm{MLM}} = -\sum_{i \in M} \log p_\theta(x_i \mid x_{\setminus M}).$$

It learns one denoising step at one fixed corruption level. Although this loss does not by itself specify a full-sequence generator, it matches the masked candidate scoring used in our non-AR evaluation.

Masked discrete diffusion. Masked diffusion extends the MLM denoising problem across the whole path from a clean sequence to an almost fully masked one. At training time, a sampled timestep t determines how many tokens are replaced by an absorbing mask, the model learns the reverse direction by recovering their clean identities. At generation time it begins from masks and repeats this recovery over decreasing noise levels, so early predictions provide context for later ones. Thus MLM is a useful single-noise-level reference point, while diffusion learns the family of denoisers needed for iterative unmasking (Austin et al., 2021; Sahoo et al., 2024).

MDLM, LLaDA, and SEDD are all instances of this masked diffusion picture. In our MDLM and LLaDA cells, $t \in [\epsilon, 1-\epsilon]$ linearly controls a mask rate from 0.02 to 0.98, and the model predicts the original tokens at masked positions. MDLM weights their cross-entropy by $1/t$ (Sahoo et al., 2024), LLaDA uses uniform weighting (Nie et al.,

2025). SEDD uses the same absorbing-mask endpoints but changes the parameterization: with log-linear noise

$$\sigma(t) = -\log(1 - (1 - \epsilon)t),$$

it predicts log score ratios and optimizes score entropy rather than ordinary token cross-entropy (Lou et al., 2024). The reported SEDD results use the sigma-scaled, mean-token-reduced implementation, earlier uncorrected runs are excluded.

Categorical flow. Flow matching learns motion along a probability path from noise to data (Lipman et al., 2023). Our CatFlow model does not jump between mask tokens. It represents each position by an unnormalized Gaussian vocabulary state z_0 and interpolates directly to the one-hot target e_x,

$$z_t = (1 - t)z_0 + te_x.$$

After timestep-dependent RMS normalization, a learned vocabulary projection maps the continuous state into the model hidden space. The model predicts the clean endpoint with cross-entropy, generation repeatedly uses those endpoint predictions to move the state toward a token. This is the non-distilled endpoint objective underlying Categorical Flow Maps (Roos et al., 2026; Davis et al., 2026), not a finite-state Discrete Flow Matching transition model (Campbell et al., 2024; Gat et al., 2024).

Evaluation. BLiMP tests English acceptability contrasts (Warstadt et al., 2020), COMPS robust property knowledge through conceptual minimal pairs (Misra et al., 2023), and Entity Tracking whether models maintain object-location state through discourse (Kim and Schuster, 2023), fine-tuning covers GLUE and SuperGLUE-style tasks (Wang et al., 2018, 2019). A central point of our results is that these benchmarks reward different inductive biases, so a single average hides a strong opposition between broad transfer and entity-state tracking.

3 Experimental Design

3.1 Shared Model and Data

All cells share the same language-model scaffold. Tokens are represented with a byte-level BPE vocabulary of 16,384 types over sequences of length 256. The retained scale study uses learned absolute position embeddings, the corrected base comparison uses sinusoidal positions so evaluation is not bounded by the training length. Time-conditioned objectives add a learned timestep embedding. The backbone is a stack of pre-norm sequence blocks, each consisting of a sequence mixer followed by a feed-forward network. The language-model head is tied to the input embedding matrix.

We train on the official 2026 Strict-Small and Strict corpora. The Strict-Small track uses a 10M-word budget and the Strict track uses a 100M-word budget. Both are trained for at most ten epochs, following the challenge rule. The base configuration has hidden size 512, 8 layers, 8 heads, and intermediate size 2048. The scale grid keeps the training recipe fixed and evaluates six width- and depth-scaled configurations: hidden sizes 640, 960, and 1344 at eight layers, and 13, 28, and 60 layers at hidden size 512, giving approximate 50M, 100M, and 200M parameter bands (Appendix Table 4).

The final flow comparison is parameter matched within each mixer. Its added vocabulary projection changes the hidden-state budget, so we use hidden size 448 and eight layers, with feed-forward sizes 1414 for attention and 1552 for Gated Delta. Direct counts give 35.706M and 39.951M trainable parameters, within 0.008% and 0.004% of the corresponding AR/MLM backbones (Appendix Table 5).

Learning-rate selection. Fixing one learning rate across families would confound the objective comparison with optimizer sensitivity, so we do not. Each of the twelve objective–mixer cells in the matched base comparison receives its own sweep on Strict-Small, and the selected rate is frozen for both tracks and all three final runs of that cell. Existing cells select by seven-task fine-tuning macro, the rematched CatFlow rerun selects by held-out pretraining loss over seven rates from $2 \cdot 10^{-4}$ to $2 \cdot 10^{-3}$, with both optima interior. Selected rates (Appendix Table 6) span an order of magnitude, from $1 \cdot 10^{-4}$ for SEDD with attention to $1 \cdot 10^{-3}$ for CatFlow with Gated Delta, and no two objectives share the same pair. Every family is therefore compared at its own selected rate rather than at AR's, so the ranking in Section 4.1 is not an artifact of a schedule chosen for autoregression. Two caveats remain: the retained AR/MLM scale study keeps its original fixed schedule, so that slice compares AR against MLM at a shared rate, and the sweep varies only the peak rate, not warmup, weight decay, batch size, or objective-specific settings such as the noise distribution (see Limitations).

We repeat every cell three times. The trustworthy scale slice contains

$$2 \text{ tracks} \times 2 \text{ objectives} \times 1 \text{ mixer} \times 6 \text{ size families} \times 3 \text{ runs} = 72$$

completed runs.

3.2 Evaluation

We run the public BabyLM 2026 evaluation pipeline. Zero-shot evaluation includes BLiMP, BLiMP Supplement, Entity Tracking, COMPS, and reading metrics when available. For the main paper, Zero macro averages BLiMP, Supplement, Entity, and COMPS. Fine-tuning evaluation covers BoolQ, MultiRC, RTE, WSC, MRPC, QQP, and MNLI. FT macro averages the primary metric for these seven tasks, with F1 used for MRPC and QQP and accuracy for the others.

Autoregressive models are scored with causal likelihood. Non-AR objectives are scored by placing mask tokens at candidate positions and reading the logits at those positions. This evaluation choice is necessary for compatibility, and it can change the conclusion. A generative objective can be strong under its own sampler and still weak under a masked candidate-scoring interface.

We therefore add HellaSwag, PIQA, WinoGrande, and ARC-Easy in two forms (Zellers et al., 2019; Bisk et al., 2020; Sakaguchi et al., 2020; Clark et al., 2018). The standard form ranks supplied candidates by causal conditional likelihood for AR and leave-one-out masked pseudo-likelihood otherwise. The paired generative form removes the candidates and asks each model to produce the gold continuation, following the direct-answer motivation of ARC-DA (Bhakthavatsalam et al., 2021). Candidate ranking covers each complete validation split, generation uses the same deterministic 256 examples per task for every model and reports exact match, positional token accuracy, normalized token F1, and ROUGE-L under a gold-length oracle. LAMBADA tests last-word continuation directly (Paperno et al., 2016). A matched WikiText diagnostic (Merity et al., 2017) fixes the same 16-word target span and supplies either left context alone (completion) or both left and right context (infilling). The latter deliberately favors bidirectional objectives, the contrast is meant to expose interface alignment, not to define a neutral overall ranking.

AR decodes greedily. MLM has no native sampler, it uses deterministic left-to-right mask prediction, one target per pass, as an evaluation heuristic. For diffusion we vary native reverse-process length, tokens committed per model call, and commitment order (confidence, left-to-right, right-to-left, random). For CatFlow we vary 1–128 integration steps, progressive versus terminal commitment, one to 16 tokens per event, and the same four orders. We compare a fixed decoder with one selected on LAMBADA/WikiText, the latter need not generalize to every task. The sweep is not inference-compute matched: every reverse or integration step is another full model evaluation, whereas AR advances one token at a time and caches its prefix. At 128 flow steps and a mean target length of 13.8 tokens this is roughly nine full-sequence evaluations per target token. The step budgets test whether extra denoising compute repairs generation, not whether the resulting sampler is efficient.

4 Experiments and Results

4.1 What Survives the Public Scorer?

Table 1 gives the system-level view. AR/attention has the best fine-tuning macro, while local winners differ: AR/Gated Delta leads Entity, MDLM/attention leads Supplement among the denoising systems, and SEDD/Gated Delta is the strongest non-AR BLiMP cell. Corrected CatFlow is no longer near chance – its attention cell reaches 64.70 BLiMP and 59.21 Supplement – but its 61-point fine-tuning macro stays below AR's 68.

The external interface makes the broader advantage clearer (Table 2). AR/attention leads candidate ranking and common seven-task generation. Decoder tuning raises CatFlow/attention from 6.46 to 8.05 token F1, but not to AR's 9.06. Its WikiText completion and infilling scores exceed AR, while its 3.52 LAMBADA score does not. Bidirectional reconstruction is therefore a real local strength, not a general replacement for causal continuation. Appendix Table 7 gives the complete official matrix for both tracks.

Individual LAMBADA cases go both ways – CatFlow exactly recovers final tokens that AR misses – but the aggregate is lopsided: AR is exact on 31 of 256 cases, versus 9 for tuned CatFlow/attention and 0 for CatFlow/Gated Delta. AR trains repeatedly on the same prefix-to-next-token decision LAMBADA evaluates, whereas denoisers spread signal across corruption levels and positions.

The ranking is also not a scale artifact. Fig-

Objective	Mixer	FT macro	BLiMP	Supp.	Entity	COMPS
AR	Attention	**68.0**±0.0	75.05±0.24	**60.87**±1.81	17.08±0.16	**55.20**±0.16
AR	Gated Delta	66.0±1.0	**75.25**±0.31	59.02±2.13	**21.40**±0.87	55.06±0.41
MLM	Attention	64.0±1.0	56.67±6.55	52.65±6.77	16.91±0.15	50.43±1.70
MLM	Gated Delta	64.0±0.0	71.33±3.57	58.99±0.94	21.23±1.03	52.89±0.43
CatFlow	Attention	61.0±1.0	64.70±0.56	59.21±0.65	20.79±0.89	52.81±0.52
CatFlow	Gated Delta	62.0±0.0	65.48±2.34	57.80±1.37	19.48±0.67	54.04±0.57
MDLM	Attention	64.0±0.0	64.44±2.37	59.53±2.33	18.07±0.40	52.62±0.35
MDLM	Gated Delta	64.0±0.0	66.69±4.33	58.76±3.44	19.66±1.26	52.50±0.47
LLaDA	Attention	64.0±1.0	63.11±2.91	58.13±1.77	18.75±1.32	52.28±0.62
LLaDA	Gated Delta	63.0±1.0	64.04±5.02	58.34±1.29	19.74±0.46	52.43±0.55
SEDD	Attention	59.0±1.0	53.13±3.14	49.82±2.30	16.67±0.41	50.64±0.67
SEDD	Gated Delta	60.0±0.0	67.40±2.75	58.05±1.08	19.56±0.37	52.57±0.57

Table 1: Corrected parameter-matched Strict results, mean and standard deviation over three seeds. Bold marks the best cell per column. Fine-tuning macro is shown as a percentage.

Objective	Mixer	Rank	Fixed	Tuned	LAMBADA	WT comp.	WT infill	Δctx
AR	Attention	**40.24**	**9.06**	—	**12.50**	8.77	8.77	0.00
AR	Gated Delta	39.70	7.34	—	5.08	7.86	7.86	0.00
MLM	Attention	38.84	2.05	—	0.39	2.67	6.08	**+3.41**
MLM	Gated Delta	38.34	5.47	—	0.39	7.61	8.67	+1.06
CatFlow	Attention	39.17	6.46	**8.05**	3.52	**11.51**	**12.41**	+0.90
CatFlow	Gated Delta	37.56	4.16	5.69	0.00	8.62	10.29	+1.67
MDLM	Attention	37.63	5.80	5.20	1.56	8.74	9.86	+1.12
MDLM	Gated Delta	38.14	3.88	4.34	0.00	6.58	8.02	+1.44
LLaDA	Attention	38.90	3.18	3.91	0.39	6.55	8.63	+2.08
LLaDA	Gated Delta	37.04	4.54	4.92	0.00	7.21	8.75	+1.54
SEDD	Attention	36.87	2.26	3.72	0.00	7.21	8.23	+1.02
SEDD	Gated Delta	38.16	3.34	5.85	0.00	8.82	9.69	+0.87

Table 2: External Strict evaluation (percent). Rank is normalized accuracy averaged over HellaSwag, PIQA, WinoGrande, and ARC-Easy. Fixed is seven-task generation token F1 under the common decoder, Tuned uses each non-AR family's selected decoder. The WikiText columns use that selected decoder (the fixed decoder for AR/MLM). Δctx is the infilling minus completion difference: the same model and the same target span, with right context added. It is the one column that varies only the scoring interface, and it is positive for all ten non-AR cells and exactly zero for AR by construction. Bold marks the best value in each column.

ure 2 shows FT macro curves for the trustworthy AR/MLM attention slice. AR stays above MLM across width and depth. Every shown 50M model is within 1.2 points of the best larger member of its scaling family, and the 50M model is often best. The inconsistent gains from 100M and 200M models, especially under depth scaling, suggest that optimization and data are already binding near the bottom of this parameter range.

4.2 How Much of the Ranking Is the Interface?

Objective and scoring interface differ together between AR and non-AR cells, so the headline ranking cannot by itself attribute the gap to either. A fully crossed experiment is unavailable under a shared recipe – our AR models have no masked pseudo-likelihood head and our denoisers no exact causal factorization – but two manipulations in Table 2 vary the interface while holding the trained model fixed, and they bound how much of the ranking the interface can explain.

The first is the Δctx column. The WikiText diagnostic fixes the same 16-word target span and supplies either left context alone or left and right context, nothing about the model, objective, or target changes. Adding right context raises token F1 for all ten non-AR cells, by $+1.51$ on average and by $+3.41$ for MLM with attention, and by construction leaves AR unchanged. The reordering is substantial: AR/attention falls from third of twelve to fifth, AR/Gated Delta from sixth to eleventh, and the number of non-AR cells above AR/attention doubles from two to four. A left-to-right interface therefore scores bidirectional objectives below their capability, by one to three token-F1 points.

The second is candidate ranking versus option-free generation. Asked only to compare supplied strings, the twelve cells span 36.87 to 40.24, a coefficient of variation of 0.03, asked to produce the continuation themselves, the same twelve checkpoints span 2.05 to 9.06, a coefficient of variation of 0.42. Almost all apparent separation between objectives lives in the generative interface, not the discriminative one.

Together these controls support a bounded claim rather than a strong one. Matching the interface

to the objective demonstrably lifts non-AR scores, and the choice of interface demonstrably sets how large the between-objective gap looks. But the corrections are one to three points, whereas AR leads MLM by four and SEDD by nine points of fine-tuning macro (Table 1) and still leads LAMBADA by 8.98 points over the best non-AR cell after decoder tuning. Interface mismatch is a measurable part of the gap, on this evidence it is not all of it. Settling the remainder needs each objective scored under its own native likelihood (Section 5).

4.3 What Does the Stateful Mixer Buy?

Averaging the six objectives in Table 1, replacing attention with Gated Delta changes fine-tuning macro by -0.17 points and Entity Tracking by $+2.13$. That is the trade-off in one line: at Strict scale the recurrent state buys entity-state accuracy and pays for it with a small loss of broad transfer.

Why the recurrent state suits entity tracking. Read literally, the delta rule of Section 2.1 is a keyed overwrite: it replaces whatever value the state associates with k_t rather than adding a second association beside it, and the gate learns how much of the old state to discard. Entity Tracking is a sequence of exactly this operation – "the coin is in the cup," then "the coin is in the box," where the second mention must supersede the first at the same address. Attention has no overwrite primitive: both mentions stay in the key–value cache and the model must learn at read time to down-weight the stale one, a discrimination it acquires from data rather than inherits from the update rule. At a 100M-word budget that difference is worth points.

Two observations support this reading and bound it. The Entity gains concentrate in the two objectives with the sharpest per-position supervision, AR ($+4.32$) and MLM ($+4.32$), and shrink or reverse where the signal is spread over corruption levels (MDLM $+1.59$, LLaDA $+0.99$, CatFlow -1.31): a state only sometimes asked to be current is a weaker overwrite. The effect is also budget dependent. On Strict-Small (Appendix Table 7) the same averages are -0.24 on Entity and $+0.17$ on fine-tuning macro, so at 10M words the advantage disappears entirely.

What the state costs. The complementary cost is retrieval: a fixed-size state must write down in advance whatever a later token needs, whereas attention retrieves any earlier key–value pair on demand. BLiMP contrasts and sentence-pair transfer both need that arbitrary lookup. The averages need care here. Gated Delta raises BLiMP by $+5.52$ points on average, but that figure is carried by MLM ($+14.66$) and SEDD ($+14.27$), the two cells whose attention runs have by far the largest seed variance (± 6.55 and ± 3.14), across the four objectives whose attention cell trains stably the mean effect is $+1.04$. What looks like a grammatical advantage for recurrence is largely an optimization-stability effect – Gated Delta rescues configurations in which attention fails to train reliably at this budget rather than learning grammar better where attention already works. Mixer effects thus stay objective- and benchmark-specific: SEDD with Gated Delta reaches the best corrected non-AR BLiMP score (67.40), MDLM with attention the best Supplement score (59.53), and Strict Entity spans a narrow 16.4–19.7 range across the diffusion objectives.

4.4 Why the Non-AR Results Are Still Informative

The local non-AR wins above occur on different scoring interfaces and do not combine into one consistently strong system. Prompt-conditioned generation in particular remains weaker. Across 1–128 flow steps, one to 16 committed tokens per event, four orders, and progressive or terminal commitment, the selected CatFlow decoders reach 9.15% (attention) and 6.30% (Gated Delta) token F1 over LAMBADA and the two WikiText tasks on the full 256-example set, still below AR/attention's 10.01%. Appendix Figure 4 shows the non-monotonic curves: more integration is not a sufficient repair. The corrected samples are diverse and preserve every prompt, but local English often fails to form a coherent continuation.

Masked diffusion is better and responds differently to commitment size (Figure 3). Mean normalized token F1 rises from 3.99% with 16 tokens committed per call to 5.50% with one, and top-five token concentration falls from 80.3% to 41.4%, the best cell is MDLM with attention at 64 native steps (7.73%). Exact matches nevertheless remain almost absent, so this is a real diversity and overlap improvement rather than a recovery of reliable answers.

4.5 What the Overall Scores Miss

To see behavior hidden by the overall benchmark scores, we ran a 92-example controlled reasoning probe on a representative matched Strict slice, us-

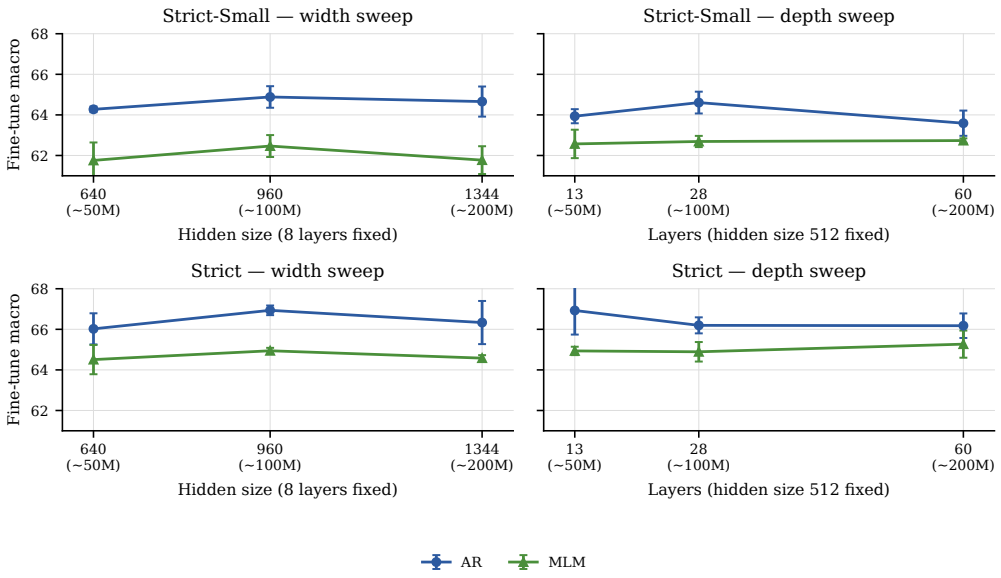


Figure 2: Fine-tuning macro for the trustworthy 72-run AR and MLM Transformer scale slice. The width panels vary hidden size over 640, 960, and 1344 while fixing depth at 8 layers, the depth panels vary 13, 28, and 60 layers while fixing hidden size at 512. Parenthetical labels give the approximate parameter band, error bars show standard deviation over three runs. Performance largely flattens after the 50M family.

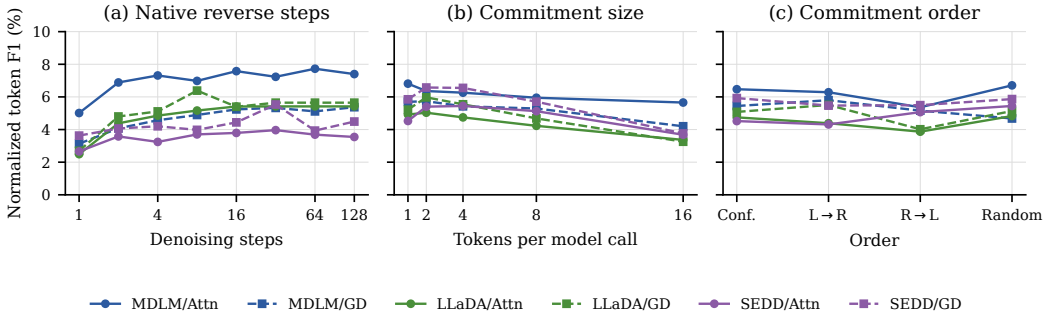


Figure 3: Masked-diffusion prompt-conditioned generation under decoder controls. Normalized token F1 is averaged over LAMBADA and paired WikiText completion and infilling. Panel (a) uses each objective's native reverse process, panels (b) and (c) average the fixed-commit grid over the other displayed controls. More native steps help initially, smaller commitments improve overlap on average, while the best order depends on the objective and sequence mixer.

ing forced-choice items rather than task fine-tuning. A model is correct when the gold continuation has higher mean log probability than the foil, under causal likelihood for AR and masked pseudo-likelihood otherwise. Table 3 pairs probe with public scores for the same checkpoints. Appendix Table 8 summarizes the slices, Tables 9 and 10 give the full matrix and inventory, and Appendix D.1 lists the exact items.

The probe shows why a BabyLM paper should not rely on a single average across tasks. MDLM Gated Delta is strongest on counterfactual state rollback and causal counterfactuals, and CatFlow Gated Delta reaches 62.5% on both counterfactual state and comparison cases even though its public fine-tuning score trails AR. The likely explanation is that the probe uses tightly matched forced choices, while the official suite additionally demands calibration, task transfer, and generation-compatible scoring.

Individual pairs also show how brittle an objective-level explanation would be. For "cutting the wire makes the alarm ring," AR with attention prefers the correct *stay silent* once the cut is removed, whereas five of six diffusion cells prefer *ring*. On the parallel item "adding the clear drop makes the liquid turn red," the same AR model prefers *red* after the drop is removed, while all

Model	Public FT	Public Entity	Probe all	CF state	Causal	WH	Role	NPI
MLM-T	64.0	16.91	**0.576**	0.438	0.625	0.750	**0.700**	**0.833**
AR-T	**68.0**	17.08	0.554	0.438	0.750	0.375	0.600	0.750
CatFlow-GD	62.0	19.48	0.533	0.625	0.500	0.500	0.300	0.500
MDLM-GD	64.0	19.66	0.533	**0.688**	**0.875**	**1.000**	**0.700**	0.083
LLaDA-GD	63.0	**19.74**	0.424	0.375	0.250	0.000	0.300	**0.833**

Table 3: Controlled probe scores beside corrected matched Strict results. Public columns are three-run BabyLM means, probe columns use representative matched checkpoints and report forced-choice accuracy over all 92 items or selected slices. CF state denotes counterfactual state rollback.

six diffusion cells correctly prefer *not red*. Such adjacent reversals point to lexical and calibration sensitivity rather than a reasoning ability cleanly supplied by one objective.

5 Discussion

Lesson 1: the scoring interface is an experimental variable. AR optimizes exactly the causal score BabyLM reads off, and the same model initializes sequence classifiers, bidirectional objectives reach the suite only through masked scores or fine-tuned heads. Section 4.2 shows the consequence is measurable and directional, but also that it does not explain the whole gap: the corrections we can measure are one to three points against a four- to nine-point fine-tuning deficit. Any BabyLM comparison across objective families thus reports an objective–scorer composite, and papers should say which part they intend to be reading. The control we leave undone is the fully crossed one – train each objective to admit the other family's scorer (an any-order likelihood for denoisers, a masked pseudo-likelihood head for AR) and report the full objective $\times$ scorer matrix. That is a training-side change, which is why it does not fit inside a fixed shared recipe.

Lesson 2: state tracking is not broad language learning. Gated recurrent state fits Entity Tracking: at Strict scale Gated Delta is worth $+2.13$ Entity points on average and -0.17 points of fine-tuning macro, and the gain is largest exactly where supervision is sharpest per position. BLiMP and the fine-tuning suite reward retrieval of arbitrary earlier material, which a fixed-size state has to anticipate and attention does not. Statefulness does not imply broad language learning, and at the 10M-word budget it does not even buy the entity-state advantage.

Lesson 3: what a matched non-AR BabyLM recipe would look like. Weak transfer through the BabyLM interface does not rule out diffusion or flow, it says the surrounding recipe is borrowed from autoregression. Four concrete changes follow. (i) *Matched scoring*: report any-order or permutation-marginal likelihoods for denoisers alongside masked pseudo-likelihood, so ranking uses the model's own factorization, Δctx is the cheap version of this test and it already moves results. (ii) *Adaptive commitment*: our diffusion sweep improves monotonically as tokens committed per model call fall from 16 to one (3.99% to 5.50% token F1), pointing at entropy-thresholded or semi-autoregressive block commitment rather than a fixed schedule – reported against an inference-compute-matched AR baseline, since at 128 flow steps we spend roughly nine full-sequence evaluations per emitted token. (iii) *Hybrid training*: GPT-BERT shows one model can carry both a causal scorer and a bidirectional encoder (Charpentier and Samuel, 2024), and the same construction around a diffusion denoiser would keep the bidirectional training signal while retaining a native left-to-right score. (iv) *Objective-native fine-tuning*: attaching classifiers to a masked score discards the denoiser's conditional structure, whereas conditioning the denoiser on the label would test transfer without routing it through an AR-shaped interface. Our results do not say diffusion cannot work at BabyLM scale, they say no part of the pipeline can be inherited unexamined.

Limitations

The comparison is a shared-recipe comparison. Holding the training stack fixed is what makes the grid interpretable, but it also means each objective is measured under a recipe whose defaults come from autoregressive practice. We tune the peak learning rate independently for all twelve objective–mixer cells (Appendix Table 6) and sweep the decoder for every non-AR family, so the two hyperparameters most likely to be mis-set are not shared. Everything else is: warmup and weight decay, batch size, the 256-token sequence length, the

ten-epoch cap, the tokenizer, the depth of time conditioning, and the masking-rate and noise schedules, which we take from each objective's published default rather than re-tuning at BabyLM scale. Diffusion and flow methods in particular are known to benefit from objective-specific schedules and architectural adaptations. Our numbers should therefore be read as characterizing these objectives *under a shared BabyLM recipe*, not at each objective's own optimum, and a negative result here is evidence about the transferability of a recipe as much as about an objective.

The scorer control is partial. Section 4.2 varies the interface while holding the model fixed, but we do not report the fully crossed objective $\times$ scorer matrix, which would require training each family to admit the other's likelihood. Claims about how much of AR's advantage is interface rather than objective are correspondingly bounded rather than settled.

The probe is small and not frequency matched. The controlled probe contains 92 hand-written forced-choice items, and its smallest slices contain eight. Its lexical material is not matched for frequency against the training corpora, so a preference could reflect familiarity rather than the structure the item targets. Adjacent items reverse (Section 4.5), which is what one expects at this sample size. The probe supports item-level diagnosis of where an inductive bias appears to help, it does not support rankings, and it does not license claims about what the public benchmarks over- or underrepresent. The apparent non-AR advantage on NPI licensing rests on 12 items and motivates a larger frequency-controlled study rather than a conclusion.

The generative tasks are deliberately stringent. The option-free counterparts to multiple-choice tasks are diagnostics, not replacement leaderboards. They supply one reference continuation even when other answers would be valid, and their gold-length oracle removes length prediction from the problem. They isolate content generation and interface alignment rather than measure unconstrained language quality. The decoder sweep is likewise not inference-compute matched, so it tests whether extra denoising compute repairs generation, not whether the resulting sampler is practical.

Scope. We report public rather than hidden BabyLM tasks, English rather than the 2026 multilingual track, and two mixers rather than a survey of efficient architectures. Gated DeltaNet is one representative of the delta-rule family and results here should not be read as covering linear-attention mixers generally.

6 Conclusion

Autoregressive Transformers transfer best throughout the retained scale sweep, already near saturation at 50M parameters. Matched-base experiments nevertheless find local non-AR wins on official metrics, bidirectional infilling, and controlled probes, and varying only the interface shows that part of AR's margin is the scorer, not the objective – though not all of it. Those wins do not combine, every tested diffusion or flow decoder stays behind AR on prompt-conditioned generation. Small-data progress requires testing objectives, mixers, decoders, and scorers together.

Acknowledgments

We thank all reviewers for their valuable comments. Alan Akbik and Patrick Haller are supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Emmy Noether grant "Eidetic Representations of Natural Language" (project number 448414230). Alan Akbik is furthermore supported under Germany's Excellence Strategy "Science of Intelligence" (EXC 2002/1, project number 390523135). Jonas Golde is supported by the German Federal Ministry of Economic Affairs and Climate Action (BMWK) as part of the project ENA (KK5148001LB0). The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project "Pre-Training SLMs via Knowledge Distillation". NHR funding is provided by federal and Bavarian state authorities.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993.

BabyLM Organizers. 2026. BabyLM challenge: Sample-efficient pretraining on a developmentally

plausible corpus. https://babylm.github.io/. Accessed 2026-07-01.

Sumithra Bhakthavatsalam, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, and Peter Clark. 2021. Think you have solved direct-answer question answering? try ARC-DA, the direct-answer AI2 reasoning challenge. *Preprint*, arXiv:2102.03315.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. 2024. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 5453–5512. PMLR.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Oscar Davis, Anastasiia Filippova, Pierre Ablin, Victor Turrisi, Amitis Shidani, Marco Cuturi, and Louis Béthune. 2026. Scaling categorical flow maps. *Preprint*, arXiv:2605.07820.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. 2024. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 82–94, Miami, FL, USA. Association for Computational Linguistics.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*, pages 175–191, Suzhou, China. Association for Computational Linguistics.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2023. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32819–32848. PMLR.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *The Fifth International Conference on Learning Representations*.

William Merrill, Yanhong Li, Tyler Romero, Anej Svete, Caia Costello, Pradeep Dasigi, Dirk Groeneveld, David Heineman, Bailey Kuehl, Nathan Lambert, Jacob Morrison, Luca Soldaini, Finbarr Timbers, Pete Walsh, Noah A. Smith, Hannaneh Hajishirzi, and Ashish Sabharwal. 2026. OLMo hybrid: From theory to practice and back. *Preprint*, arXiv:2604.03444.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong

Wen, and Chongxuan Li. 2025. Large language diffusion models. *Advances in Neural Information Processing Systems*, 38:50608–50646.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Rareş Păpuşoi and Sergiu Nisioi. 2025. A comparison of elementary baselines for BabyLM. In *Proceedings of the First BabyLM Workshop*, pages 218–225, Suzhou, China. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Daan Roos, Oscar Davis, Floor Eijkelboom, Michael Bronstein, Max Welling, İsmail İlkan Ceylan, Luca Ambrogioni, and Jan-Willem van de Meent. 2026. Categorical flow maps. In *Forty-third International Conference on Machine Learning*.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32:3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025. Gated delta networks: Improving Mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. 2024. Parallelizing linear transformers with the delta rule over sequence length. *Advances in Neural Information Processing Systems*, 37:115491–115522.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Configuration Details

Table 4 gives the controlled scale families used in the retained sweep and Table 5 the parameter matching for the corrected categorical-flow comparison.

B Selected Learning Rates

Table 6 lists the peak learning rate selected for each of the twelve objective–mixer cells in the matched base comparison. Each cell was swept independently on Strict-Small, the selected value was then frozen for both tracks and all three seeds of that cell. No two objectives share the same pair of rates, and the selected values span an order of magnitude.

C Objective and Mixer Interactions

Table 7 complements the Strict matrix in Table 1 with the corrected Strict-Small results.

D Controlled Probe Details

Table 8 summarizes the composition of the probe and Table 9 reports the full 12-model by 9-slice probe matrix. The main paper reports a selected subset of these columns to focus on the contrast between state tracking and broad transfer.

Table 10 gives the item inventory used to build the 92-case probe.

D.1 Probe Items

Two negation-scope items use commas here in place of the semicolons used by the scoring file.

1. NPI 1. Gold: Only the nurse has ever touched the red folder. Foil: Even the nurse has ever touched the red folder.
2. NPI 2. Gold: Only the pilot had ever opened the silver hatch. Foil: Even the pilot had ever opened the silver hatch.
3. NPI 3. Gold: No student has ever entered the blue room. Foil: Every student has ever entered the blue room.
4. NPI 4. Gold: No visitor has ever borrowed the green badge. Foil: Every visitor has ever borrowed the green badge.
5. NPI 5. Gold: Hardly any baker has ever used the cracked bowl. Foil: Many bakers have ever used the cracked bowl.
6. NPI 6. Gold: Hardly any dancer had ever missed the final cue. Foil: Many dancers had ever missed the final cue.
7. NPI 7. Gold: Did the artist ever paint the mural? Foil: The artist ever painted the mural.
8. NPI 8. Gold: Had the captain ever moved the compass? Foil: The captain had ever moved the compass.
9. NPI 9. Gold: The guard did not give anyone the key. Foil: The guard gave anyone the key.
10. NPI 10. Gold: The child did not see anything in the drawer. Foil: The child saw anything in the drawer.
11. NPI 11. Gold: Few reporters have ever met the senator. Foil: Most reporters have ever met the senator.
12. NPI 12. Gold: Only Mira would ever question the rule. Foil: Even Mira would ever question the rule.
13. WH 13. Gold: The detective revealed who the witness had accused. Foil: The detective revealed that the witness had accused.
14. WH 14. Gold: Maya remembered what the teacher had placed on the shelf. Foil: Maya remembered that the teacher had placed on the shelf.
15. WH 15. Gold: The clerk forgot which package the courier had delivered. Foil: The clerk forgot that package the courier had delivered.
16. WH 16. Gold: The singer announced which song the choir would perform. Foil: The singer announced that song the choir would perform.
17. WH 17. Gold: The judge asked who the lawyer had called. Foil: The judge asked that the lawyer had called.
18. WH 18. Gold: The parent knew what the toddler had hidden under the bed. Foil: The parent knew that the toddler had hidden under the bed.
19. WH 19. Gold: Which book did the editor praise? Foil: That book did the editor praise?
20. WH 20. Gold: Who did the reporter interview after lunch? Foil: That did the reporter interview after lunch?
21. State 21. Context: At first, the coin was in the cup. Nora moved the coin from the cup to the box. Question: Where is the coin now? Answer: Gold: the box. Foil: the cup.
22. State 22. Context: At first, the key was in the drawer. Liam moved the key from the drawer to the jar. Question: Where is the key now? Answer: Gold: the jar. Foil: the drawer.
23. State 23. Context: At first, the ticket was in the wallet. Ava moved the ticket from the wallet to the envelope. Question: Where is the ticket now? Answer: Gold: the envelope. Foil: the wallet.
24. State 24. Context: At first, the marble was in the basket. Omar moved the marble from the basket to the closet. Question: Where is the marble now? Answer: Gold: the closet. Foil: the basket.
25. State 25. Context: At first, the ring was in the shelf. Mina moved the ring from the shelf to the pouch. Question: Where is the ring now? Answer: Gold: the pouch. Foil: the shelf.
26. State 26. Context: At first, the button was in the tray. Ezra moved the button from the tray to the cabinet. Question: Where is the button now? Answer: Gold: the cabinet. Foil: the tray.

Scaling family	Configurations varied	Architectural coordinate held fixed
Width	hidden sizes 640 (∼50M), 960 (∼100M), 1344 (∼200M)	8 layers
Depth	13 (∼50M), 28 (∼100M), 60 (∼200M) layers	hidden size 512

Table 4: Controlled scale families, parenthetical labels are approximate parameter bands.

Mixer	Hidden	Layers	FFN	Flow params	Reference params
Attention	448	8	1414	35,705,712	35,708,416
Gated Delta	448	8	1552	39,951,072	39,952,768

Table 5: Parameter matching for the corrected categorical-flow comparison. References are the corresponding AR/MLM backbones, differences are 0.0076% for attention and 0.0042% for Gated Delta.

Objective	Attention	Gated Delta
AR	5	3
MLM	3	3
CatFlow	7	10
MDLM	3	2
LLaDA	5	3
SEDD	1	3

Table 6: Selected peak learning rates for the matched base comparison, in units of 10^{-4}. Existing cells select by seven-task fine-tuning macro, the rematched CatFlow rerun selects by held-out pretraining loss.

27. State 27. Context: At first, the card was in the folder. Iris moved the card from the folder to the tin. Question: Where is the card now? Answer: Gold: the tin. Foil: the folder.

28. State 28. Context: At first, the spoon was in the sink. Theo moved the spoon from the sink to the drawer. Question: Where is the spoon now? Answer: Gold: the drawer. Foil: the sink.

29. State 29. Context: At first, the map was in the desk. Jules moved the map from the desk to the box. Jules then moved the map from the box to the shelf. Question: Where is the map now? Answer: Gold: the shelf. Foil: the box.

30. State 30. Context: At first, the photo was in the album. Lea moved the photo from the album to the drawer. Lea then moved the photo from the drawer to the bag. Question: Where is the photo now? Answer: Gold: the bag. Foil: the drawer.

31. State 31. Context: At first, the shell was in the bowl. Noah moved the shell from the bowl to the basket. Noah then moved the shell from the basket to the cup. Question: Where is the shell now? Answer: Gold: the cup. Foil: the basket.

32. State 32. Context: At first, the note was in the book. Rosa moved the note from the book to the tin. Rosa then moved the note from the tin to the pocket. Question: Where is the note now? Answer: Gold: the pocket. Foil: the tin.

33. CF state 33. Context: At first, the map was in the desk. Jules moved the map from the desk to the box. Jules then moved the map from the box to the shelf. Question: If the first move had not happened, where would the map be? Answer: Gold: the desk. Foil: the shelf.

34. CF state 34. Context: At first, the photo was in the album. Lea moved the photo from the album to the drawer. Lea then moved the photo from the drawer to the bag. Question: If the first move had not happened, where would the photo be? Answer: Gold: the album. Foil: the bag.

35. CF state 35. Context: At first, the shell was in the bowl. Noah moved the shell from the bowl to the basket. Noah then moved the shell from the basket to the cup. Question: If the first move had not happened, where would the shell be? Answer: Gold: the bowl. Foil: the cup.

36. CF state 36. Context: At first, the note was in the book. Rosa moved the note from the book to the tin. Rosa then moved the note from the tin to the pocket. Question: If the first move had not happened, where would the note be? Answer: Gold: the book. Foil: the pocket.

37. CF state 37. Context: At first, the map was in the desk. Jules moved the map from the desk to the box. Jules then moved the map from the box to the shelf. Question: If the second move had not happened, where would the map be? Answer: Gold: the box. Foil: the shelf.

38. CF state 38. Context: At first, the photo was in the album. Lea moved the photo from the album to the drawer. Lea then moved the photo from the drawer to the bag. Question: If the second move had not happened, where would the photo be? Answer: Gold: the drawer. Foil: the bag.

39. CF state 39. Context: At first, the shell was in the bowl. Noah moved the shell from the bowl to the basket. Noah then moved the shell from the basket to the cup. Question: If the second move had not happened, where would the shell be? Answer: Gold: the basket. Foil: the cup.

40. CF state 40. Context: At first, the note was in the book. Rosa moved the note from the book to the tin. Rosa then moved the note from the tin to the pocket. Question: If the second move had not happened, where would the note be? Answer: Gold: the tin. Foil: the pocket.

41. CF state 41. Context: At first, the coin was in the cup. Nora moved the coin from the cup to the box. Question: If that move had not happened, where would the coin be? Answer: Gold: the cup. Foil: the box.

42. CF state 42. Context: At first, the key was in the drawer. Liam moved the key from the drawer to the jar. Question: If that move had not happened, where would the key be? Answer: Gold: the drawer. Foil: the jar.

Objective	Mixer	FT macro	BLiMP	Supp.	Entity	COMPS
AR	Attention	63.0±0.0	64.33±0.29	55.76±0.39	18.31±0.53	50.73±0.38
AR	Gated Delta	61.0±1.0	64.27±0.27	53.19±2.35	17.71±0.24	50.27±0.07
MLM	Attention	59.0±1.0	54.04±1.72	46.99±1.96	16.45±0.32	50.21±0.14
MLM	Gated Delta	61.0±1.0	54.10±0.52	51.10±2.71	16.77±0.38	49.63±0.40
CatFlow	Attention	61.0±0.0	55.82±0.49	48.12±1.20	17.20±0.39	49.80±0.23
CatFlow	Gated Delta	62.0±0.0	57.11±1.55	51.59±0.88	17.50±0.58	50.28±0.43
MDLM	Attention	59.0±1.0	54.32±0.09	45.86±1.69	17.22±0.60	49.87±0.06
MDLM	Gated Delta	59.0±1.0	55.14±1.13	46.10±0.74	16.40±0.29	49.83±0.21
LLaDA	Attention	59.0±1.0	55.12±1.00	45.59±1.39	17.26±0.16	50.11±0.47
LLaDA	Gated Delta	60.0±1.0	54.13±1.18	46.19±2.04	17.17±0.38	49.97±0.49
SEDD	Attention	59.0±1.0	48.34±3.93	48.60±1.78	17.15±0.27	50.43±0.18
SEDD	Gated Delta	58.0±1.0	54.86±1.80	45.47±0.72	16.62±0.34	50.09±0.40

Table 7: Corrected parameter-matched Strict-Small results, mean and standard deviation over three seeds. Fine-tuning macro is shown as a percentage.

Slice	n	What it tests	Example gold / foil contrast
NPI licensing	12	Licensing of negative-polarity items such as *ever* and *anything*.	Only Mira would ever question the rule / Even Mira would ever question the rule.
WH filler-gap	8	Preference for a wh filler that licenses a gap over a local declarative foil.	who the witness had accused / that the witness had accused.
Ordinary state	12	Tracking an object's actual location after one or two observed moves.	final location / previous location.
Counterfactual state	16	Rolling back an omitted move while ignoring the observed final location.	drawer / bag, when the second move to the bag is counterfactually removed.
Causal counterfactual	8	Removing an observed effect when its cause is counterfactually absent.	invalid / valid, when stamping a card is counterfactually removed.
Comparison	8	Transitive ordering and reversed-order counterfactuals.	Ava / Cara, after comparing a three-person height order.
Negation scope	8	Resolving exceptives and contrastive negation against local distractors.	the bear / the train, after all toys except the bear are packed.
Semantic role	10	Binding agents, patients, and recipients across active and passive forms.	the student / the teacher, after the teacher gives the student a pencil.
Novel binding	10	Binding novel words to newly introduced properties and negated properties.	the dax / the wug, after dax is defined as the tool that cuts paper.

Table 8: Composition of the controlled probe. Each item is a matched forced-choice comparison scored by preference for the gold continuation over the foil. Examples are abridged only for table length.

Model	All	State	CF state	Causal	Comp.	Neg.	NPI	WH	Role	Novel
AR-Attn	55.4	41.7	43.8	75.0	37.5	75.0	75.0	37.5	60.0	60.0
AR-GD	46.7	41.7	56.2	0.0	25.0	62.5	58.3	75.0	50.0	40.0
MLM-Attn	57.6	58.3	43.8	62.5	50.0	62.5	83.3	75.0	70.0	20.0
MLM-GD	44.6	50.0	37.5	25.0	37.5	62.5	25.0	25.0	70.0	70.0
SEDD-Attn	44.6	41.7	50.0	25.0	50.0	62.5	41.7	12.5	50.0	60.0
SEDD-GD	46.7	50.0	50.0	25.0	62.5	50.0	58.3	25.0	50.0	40.0
LLaDA-Attn	40.2	50.0	56.2	25.0	50.0	62.5	25.0	0.0	20.0	60.0
LLaDA-GD	42.4	41.7	37.5	25.0	50.0	62.5	83.3	0.0	30.0	40.0
MDLM-Attn	42.4	41.7	62.5	25.0	50.0	75.0	41.7	0.0	20.0	50.0
MDLM-GD	53.3	25.0	68.8	87.5	37.5	25.0	8.3	100.0	70.0	70.0
CatFlow-Attn	52.2	41.7	62.5	75.0	37.5	62.5	75.0	25.0	40.0	40.0
CatFlow-GD	53.3	50.0	62.5	50.0	62.5	62.5	50.0	50.0	30.0	60.0

Table 9: Full controlled-probe matrix. Values are forced-choice accuracies in percent for representative matched Strict checkpoints. State is ordinary state tracking, CF state is counterfactual state rollback, Novel is novel-word compositional binding.

43. CF state 43. Context: At first, the ticket was in the wallet. Ava moved the ticket from the wallet to the envelope. Question: If that move had not happened, where would the ticket be? Answer: Gold: the wallet. Foil: the envelope.

44. CF state 44. Context: At first, the marble was in the basket. Omar moved the marble from the basket to the closet. Question: If that move had not happened, where would the marble be? Answer: Gold: the basket. Foil: the closet.

45. CF state 45. Context: At first, the ring was in the shelf. Mina moved the ring from the shelf to the pouch. Question: If that move had not happened, where would the ring be? Answer: Gold: the shelf. Foil: the pouch.

46. CF state 46. Context: At first, the button was in the tray. Ezra moved the button from the tray to the cabinet. Question: If that move had not happened, where would the button be? Answer: Gold: the tray. Foil: the cabinet.

47. CF state 47. Context: At first, the card was in the folder. Iris moved the card from the folder to the tin. Question: If that move had not happened, where would the card be? Answer: Gold: the folder. Foil: the tin.

48. CF state 48. Context: At first, the spoon was in the sink. Theo moved the spoon from the sink to the drawer.

Slice	Subcategory	n	Construction
NPI licensing	licensor present	12	Licensed *ever*, *anyone*, or *anything* under negation, questions, *only*, *few*, or related downward-entailing contexts, paired with an unlicensed foil.
WH filler-gap	wh versus that	8	Wh fillers such as *who*, *what*, and *which* license a gap, while the foil replaces the filler with a local declarative form.
Ordinary state	one move	8	An object starts in one container and is moved once. The gold answer is the post-move location.
Ordinary state	two moves	4	An object is moved twice. The gold answer is the final location rather than the intermediate location.
Counterfactual state	undo single move	8	A one-step move is counterfactually removed. The gold answer is the initial location.
Counterfactual state	undo second move	4	A two-step story removes only the second move. The gold answer is the intermediate location.
Counterfactual state	undo first move	4	A two-step story removes the first move. The gold answer is the initial location.
Causal counterfactual	prevent effect	8	A local rule maps an observed cause to an observed effect. The counterfactual question removes the cause and the gold answer removes the effect.
Comparison	transitive order	6	Two pairwise comparisons define a three-item order. The question asks for an endpoint of the order.
Comparison	reversed order	2	The same ordering format asks what would happen if the order were reversed.
Negation scope	exception or contrast	8	Exceptives and contrastive negation override a nearby distractor answer.
Semantic role	agent-patient binding	10	Active, passive, and ditransitive sentences test whether the model binds agents, patients, and recipients.
Novel binding	positive property	7	Novel words are assigned new properties. The question asks which novel item has a target property.
Novel binding	negated property	3	Novel words are assigned contrastive positive and negated properties. The question asks for the item satisfying the negated relation.

Table 10: Full item inventory for the controlled probe. Counts sum to 92 matched forced-choice examples.

Question: If that move had not happened, where would the spoon be? Answer: Gold: the sink. Foil: the drawer.

49. Causal 49. Context: In this game, pressing the blue switch opens the gate. Sam pressed the blue switch. Question: If Sam had not pressed the switch, what would happen to the gate? Answer: Gold: it would stay closed. Foil: it would open.

50. Causal 50. Context: In this lab, adding the clear drop makes the liquid turn red. Dana added the clear drop. Question: If Dana had not added the drop, what color would the liquid be? Answer: Gold: not red. Foil: red.

51. Causal 51. Context: In this machine, cutting the wire makes the alarm ring. The worker cut the wire. Question: If the worker had not cut the wire, what would the alarm do? Answer: Gold: stay silent. Foil: ring.

52. Causal 52. Context: In this game, placing the token on star makes the door lock. Priya placed the token on star. Question: If Priya had not placed the token on star, what would the door do? Answer: Gold: remain unlocked. Foil: lock.

53. Causal 53. Context: In this kitchen, turning the knob starts the fan. Alex turned the knob. Question: If Alex had not turned the knob, what would the fan do? Answer: Gold: stay off. Foil: start.

54. Causal 54. Context: In this puzzle, pulling the rope lowers the bridge. Kim pulled the rope. Question: If Kim had not pulled the rope, what would the bridge do? Answer: Gold: stay raised. Foil: lower.

55. Causal 55. Context: In this rule system, stamping the card marks it valid. The clerk stamped the card. Question: If the clerk had not stamped it, how would the card be marked? Answer: Gold: invalid. Foil: valid.

56. Causal 56. Context: In this device, tapping the screen saves the file. The user tapped the screen. Question: If the user had not tapped the screen, what would happen to the file? Answer: Gold: it would not be saved. Foil: it would be saved.

57. Comp 57. Context: Ava is taller than Ben. Ben is taller than Cara. Question: If the order were reversed, who would be tallest? Answer: Gold: Cara. Foil: Ava.

58. Comp 58. Context: Milo arrived before Vera. Vera arrived before June. Question: If the arrival order were reversed, who would arrive last? Answer: Gold: Milo. Foil: June.

59. Comp 59. Context: Ava is taller than Ben. Ben is taller than Cara. Question: Who is tallest? Answer: Gold: Ava. Foil: Cara.

60. Comp 60. Context: The red box is heavier than the green box. The green box is heavier than the blue box. Question: Which box is lightest? Answer: Gold: the blue box. Foil: the red box.

61. Comp 61. Context: Milo arrived before Vera. Vera arrived before June. Question: Who arrived last? Answer: Gold: June. Foil: Milo.

62. Comp 62. Context: The oak tree is older than the pine tree. The pine tree is older than the birch tree. Question: Which tree is oldest? Answer: Gold: the oak tree. Foil: the birch tree.

63. Comp 63. Context: The silver key is smaller than the brass key. The brass key is smaller than the black key. Question: Which key is largest? Answer: Gold: the black key. Foil: the silver key.

64. Comp 64. Context: Lena scored more points than Kai. Kai scored more points than Omar. Question: Who scored the fewest points? Answer: Gold: Omar. Foil: Lena.

65. Neg 65. Context: Nina packed every toy except the bear. Question: Which toy was not packed? Answer: Gold: the bear. Foil: the train.

66. Neg 66. Context: All cups except the red cup are clean. Question: Is the red cup clean? Answer: Gold: no. Foil: yes.

67. Neg 67. Context: The key is not in the left drawer, it is in the right drawer. Question: Where is the key? Answer: Gold: the right drawer. Foil: the left drawer.

68. Neg 68. Context: The note was not under the mat, it was behind the lamp. Question: Where was the note? Answer: Gold: behind the lamp. Foil: under the mat.

69. Neg 69. Context: Every child but Mira received a sticker. Question: Who did not receive a sticker? Answer: Gold: Mira. Foil: every child.

70. Neg 70. Context: The dog chased the cat, not the rabbit. Question: What did the dog chase? Answer: Gold: the cat. Foil: the rabbit.

71. Neg 71. Context: The red button starts the motor, but the green button does not. Question: Which button starts the motor? Answer: Gold: the red button. Foil: the green button.

72. Neg 72. Context: The square is inside the circle, not outside it. Question: Where is the square? Answer: Gold: inside the circle. Foil: outside the circle.

73. Role 73. Context: The doctor read the book to the child. Question: Who read? Answer: Gold: the doctor. Foil: the book.

74. Role 74. Context: The teacher gave the student a pencil. Question: Who received the pencil? Answer: Gold: the student. Foil: the teacher.

75. Role 75. Context: The chef was praised by the waiter. Question: Who was praised? Answer: Gold: the chef. Foil: the waiter.

76. Role 76. Context: The musician was followed by the reporter. Question: Who followed someone? Answer: Gold: the reporter. Foil: the musician.

77. Role 77. Context: The letter was written by the sailor. Question: Who wrote? Answer: Gold: the sailor. Foil: the letter.

78. Role 78. Context: The artist showed the critic a painting. Question: What was shown? Answer: Gold: a painting. Foil: the critic.

79. Role 79. Context: The nurse called the patient after dinner. Question: Who was called? Answer: Gold: the patient. Foil: the nurse.

80. Role 80. Context: The captain was warned by the guard. Question: Who gave the warning? Answer: Gold: the guard. Foil: the captain.

81. Role 81. Context: The child carried the basket to the porch. Question: What was carried? Answer: Gold: the basket. Foil: the porch.

82. Role 82. Context: The driver thanked the mechanic. Question: Who was thanked? Answer: Gold: the mechanic. Foil: the driver.

83. Novel 83. Context: A dax is a tool that cuts paper. A wug is a tool that stirs soup. Question: Which tool cuts paper? Answer: Gold: the dax. Foil: the wug.

84. Novel 84. Context: A blicket is a fruit that tastes sour. A toma is a fruit that tastes sweet. Question: Which fruit tastes sweet? Answer: Gold: the toma. Foil: the blicket.

85. Novel 85. Context: A fep is a machine that prints labels. A lorp is a machine that folds boxes. Question: Which machine folds boxes? Answer: Gold: the lorp. Foil: the fep.

86. Novel 86. Context: A zup is a bird that sings at night. A niff is a bird that sleeps at night. Question: Which bird sings at night? Answer: Gold: the zup. Foil: the niff.

87. Novel 87. Context: A mip is a stone that glows in water. A tane is a stone that floats in water. Question: Which stone floats in water? Answer: Gold: the tane. Foil: the mip.

88. Novel 88. Context: A vorn is a plant that needs shade. A pilk is a plant that needs sun. Question: Which plant needs shade? Answer: Gold: the vorn. Foil: the pilk.

89. Novel 89. Context: A blicket is sour but not sweet. A toma is sweet but not sour. Question: Which item is not sour? Answer: Gold: the toma. Foil: the blicket.

90. Novel 90. Context: A dax cuts paper but does not stir soup. A wug stirs soup but does not cut paper. Question: Which item stirs soup? Answer: Gold: the wug. Foil: the dax.

91. Novel 91. Context: A fep prints labels but does not fold boxes. A lorp folds boxes but does not print labels. Question: Which item does not print labels? Answer: Gold: the lorp. Foil: the fep.

92. Novel 92. Context: A zup sings at night but does not sleep at night. A niff sleeps at night but does not sing at night. Question: Which item does not sleep at night? Answer: Gold: the zup. Foil: the niff.

E Categorical-Flow Decoder Sweep

Figure 4 shows the full categorical-flow decoder grid summarized in Section 4.4.

F Generation Diagnostics

Table 11 reports the generated-sample diagnostics used in Section 4.4. AR models use continuation sampling. Masked and diffusion objectives use full-unmask and prompt-unmask samplers. CatFlow also uses unconditional flow sampling.

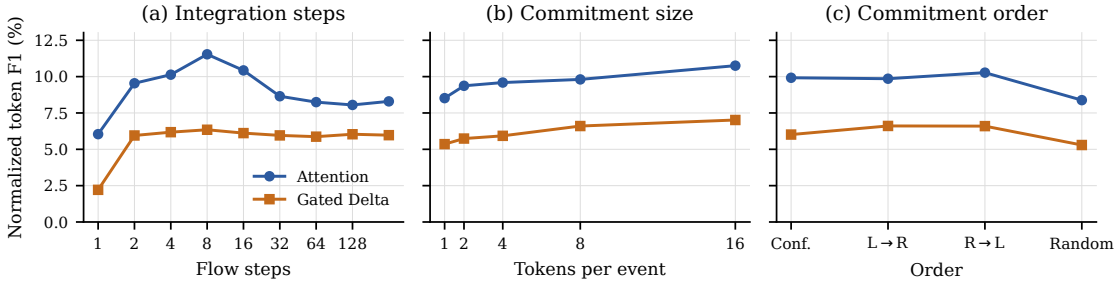


Figure 4: Categorical-flow prompt-conditioned generation under decoder controls. Normalized token F1 is averaged over LAMBADA and paired WikiText completion and infilling. Panel (a) holds progressive left-to-right commitment fixed, panels (b) and (c) average the fixed-commit grid over the other displayed controls. More integration steps, fewer committed tokens, and alternative orders do not materially repair generation for either sequence mixer.

Obj.	Mix.	Mode	Tokens	Dist-2	Rep-4
AR	GD	continuation	28.3	0.824	0.048
AR	Attn	continuation	25.3	0.807	0.071
CatFlow	GD	flow-unconditional	59.3	0.705	0.019
CatFlow	GD	full-unmask	58.8	0.717	0.014
CatFlow	GD	prompt-unmask	62.3	0.681	0.020
CatFlow	Attn	flow-unconditional	57.0	0.613	0.169
CatFlow	Attn	full-unmask	57.2	0.625	0.159
CatFlow	Attn	prompt-unmask	60.9	0.670	0.071
LLaDA	GD	full-unmask	31.0	0.121	0.696
LLaDA	GD	prompt-unmask	50.1	0.334	0.444
LLaDA	Attn	full-unmask	12.0	0.024	0.975
LLaDA	Attn	prompt-unmask	51.5	0.102	0.883
MDLM	GD	full-unmask	54.0	0.127	0.852
MDLM	GD	prompt-unmask	54.0	0.285	0.606
MDLM	Attn	full-unmask	1.0	0.000	0.000
MDLM	Attn	prompt-unmask	43.4	0.116	0.871
MLM	GD	full-unmask	32.0	0.194	0.450
MLM	GD	prompt-unmask	55.1	0.330	0.479
MLM	Attn	full-unmask	33.0	0.016	0.984
MLM	Attn	prompt-unmask	39.5	0.342	0.591
SEDD	GD	full-unmask	12.0	0.238	0.474
SEDD	GD	prompt-unmask	49.2	0.287	0.597
SEDD	Attn	full-unmask	1.0	1.000	0.000
SEDD	Attn	prompt-unmask	12.2	0.884	0.112

Table 11: Generated-sample diagnostics on representative matched Strict models. Tokens is mean decoded length over 64 samples, Dist-2 is distinct bigrams and Rep-4 is the repeated four-gram rate. Corrected CatFlow rows replace the legacy softmax-input runs.

Model	Mode	Representative output
AR-Attn	continuation	Prompt: The child found. Output: The child found it hard to stop.
AR-GD	continuation	Prompt: The child found. Output: The child found a way to get into the little girl's arms and hold her tight.
LLaDA-Attn	prompt-unmask	Prompt: A scientist explained that. Output: A scientist explained that.....
MDLM-GD	prompt-unmask	Prompt: Yesterday the teacher said. Output: Yesterday the teacher said, "I didn't know you." I didn't. "I didn't know."
MDLM-Attn	full-unmask	Output: .
MLM-Attn	prompt-unmask	Prompt: Yesterday the teacher said. Output: Yesterday the teacher said that.
SEDD-Attn	prompt-unmask	Prompt: The child found. Output: The child found.
CatFlow-GD	prompt-unmask	Prompt: A scientist explained that. Output: A scientist explained that, problem the it there may be a key Earth with medicine.
CatFlow-Attn	prompt-unmask	Prompt: In the box there was. Output: In the box there was no sign to show them the ones they had the clubs out for dinner.

Table 12: Representative generated outputs. Examples are selected from the same sample-analysis run as Table 11 and are shortened only for space.

Do Association-Reweighted BPE Merge Rules Help Data-Efficient Language Modeling?

Zy Li Phu Han Trinh Tran Lam Duong Andrew Fang
Department of Computer Science, Seattle University
{zli10, phan, ktran24, lduong, afang}@seattleu.edu

Abstract

Subword tokenization determines a language model's symbol inventory and sequence length, yet standard BPE selects merges by frequency alone. We test whether association-reweighted merges help on BabyLM Strict-Small. Holding the merge budget and architecture fixed and approximately matching word exposure, we train 97M-parameter causal decoders with three seeds using standard BPE, the S-BPE merge score with fixed-budget stopping, and exponential positive-PMI reweighting (PMI-BPE). Neither alternative reliably improves held-out bits-per-byte (BPB) or BLiMP. PMI-BPE's mean BPB is slightly higher than standard BPE's, but no pairwise difference is statistically reliable at three seeds. Both alternatives alter the merge tables and measurably change compression, ruling out a trivial no-intervention explanation. Yet intrinsic association rankings depend on scoring measure and aggregation. Type-uniform rankings reverse between post-merge PMI and thresholded held-out PPMI, while application-weighted PPMI is nearly identical across tokenizers. These results do not rule out association-aware tokenization; they cover one merge budget, one operating point per score, and two evaluation axes. Within that scope, the tested scores do not reliably help, and intrinsic merge association does not consistently align with the BPB and BLiMP point-estimate orderings.

1 Introduction

Subword tokenization is the first learned component of a language model and one of its few discrete design choices. Byte-Pair Encoding (BPE) (Gage, 1994; Sennrich et al., 2016) and its relatives fix the input and output vocabulary. This shapes sequence lengths, the granularity of the units a model must predict, and how the finite training budget is spent. This choice is especially consequential in the data-efficient setting, where the corpus is small and fixed. The question is then how much can be learned per word rather than per FLOP, and no additional data can compensate for a poor segmentation.

Standard BPE is greedy and purely frequency-driven: at each step it merges the adjacent pair with the highest raw bigram count, $\arg\max_{(a,b)} N(a,b)$. This objective is agnostic to whether a pair coheres as a linguistic unit or merely co-occurs often. A recent line of methods revisits exactly this merge rule, reweighting it by a statistical association score so that merges reflect *cohesion* rather than raw frequency. We use *cohesion* as shorthand for statistical association under the stated metric, not as a direct measure of morphological or semantic well-formedness. Most build on pointwise mutual information (Church and Hanks, 1990). S-BPE weights the merge count by a pointwise mutual information term (Vilar and Federico, 2021), and related proposals reweight or gate merges by significance or attribution signals (Nouri, 2026; Foroutan et al., 2026; Chizhov et al., 2026). The shared intuition is to prefer merges whose parts predict each other rather than merely co-occur. A broader literature likewise argues that tokenization quality is not reducible to compression (Zouhar et al., 2023; Goldman et al., 2024; Schmidt et al., 2024; Bostrom and Durrett, 2020), motivating tokenizers optimized for something other than sequence length.

The underlying question has a narrow, testable form: *does association-aware tokenization actually improve data-efficient language modeling?* We answer it in the stringent low-resource BabyLM Strict-Small setting (Warstadt et al., 2023; Choshen et al., 2026), a fixed 10M-word corpus of developmentally plausible text. We train small language models from scratch under three tokenizers held to a common merge budget and a common *word* budget. The three are standard greedy BPE and two association-based merge criteria. The first is the S-BPE merge score with fixed-budget stopping (Vilar and Federico, 2021), which weights the merge

Proceedings of the Second BabyLM Workshop, pages 93–104
October 29, 2026 ©2026 Association for Computational Linguistics

count by a pointwise mutual information term. The second is an exponential positive-PMI reweighting $N(a,b)\exp(\beta\,\mathrm{PPMI}(a,b))$, which we call PMI-BPE and adopt as a representative instantiation of the PMI-reweighting family (published members use related but distinct functionals (Nouri, 2026), and $\beta{=}0$ recovers greedy BPE). All three use a common implementation so that only the merge criterion differs. We evaluate on bits-per-byte (BPB) over a held-out slice and on BLiMP zero-shot grammatical judgments (Warstadt et al., 2020), with three seeds per tokenizer.

Our answer is negative. Both alternatives substantially alter the merge table, ruling out a trivial no-intervention explanation. Intrinsic *association* does not provide a stable tokenizer ranking in this experiment, because whether a reweighted tokenizer scores higher than standard BPE depends on the association measure. Under the type-uniform post-merge-PMI summary both reweighted tokenizers exceed standard BPE. Under the type-uniform thresholded-PPMI summary neither does, and PMI-BPE, which uses positive PMI in its merge score, ranks last. Moreover, weighting each merge by how often its rule is applied narrows the positive-PMI spread to a fraction of a percent, and weighting by final-token frequency leaves standard BPE, not the reweighted tokenizers, slightly ahead. These diagnostics provide little reason to expect substantial separation on either evaluation axis, and we detect none reliably at three seeds (Table 3). The larger of the two differences from standard BPE is PMI-BPE's higher mean BPB, reflecting a compression–prediction tradeoff: the easier per-token predictions its finer segmentation affords do not offset the additional predictions per byte it requires. Our conclusion rests on the two evaluation axes, not on the intrinsic association scores. Those scores rank the tokenizers differently under different measures, so we treat them as descriptive and do not offer them as an explanation of the negative result.

We make the following contributions:

- **A controlled data-efficient study.** We compare standard BPE, PMI-BPE, and S-BPE (Vilar and Federico, 2021) at a fixed merge budget and approximately matched word exposure, training a 97M-parameter Llama-style decoder from scratch with three seeds per tokenizer (nine runs), and report BPB and BLiMP in the BabyLM Strict-Small setting (Choshen et al., 2026).
- **An aggregation analysis of merge association.** We compare type-uniform, merge-application-weighted, and final-token-weighted association summaries. On this corpus, their rankings depend on the association measure. Under thresholded held-out PPMI, much of the difference is associated with rarely applied or unseen merges.
- **A negative result.** Neither association-reweighted merge rule reliably improves BPB or BLiMP. PMI-BPE has the higher mean BPB of the two alternatives, a compression–prediction tradeoff (Section 5).

Consistent with the analysis-oriented aims of the BabyLM effort (Warstadt et al., 2023), we report this negative result. We include the intrinsic measurements that document the tokenizer-level changes and how the intrinsic ranking depends on the association measure. Future work can then test whether association concentrated in frequent final tokens, or another model-relevant property, affects data-efficient learning.

2 Related Work

Association-based merge criteria. Standard BPE (Gage, 1994; Sennrich et al., 2016) grows a vocabulary by repeatedly merging the most frequent adjacent pair. This purely frequency-driven rule draws no distinction between statistically bound units (e.g. *morphemes*) and accidental within-word collocations, and a long line of work has replaced or augmented it with an *association* measure. WordPiece (Schuster and Nakajima, 2012; Wu et al., 2016) selects the merge that most increases training-corpus likelihood, a likelihood-ratio rule closely related to pointwise mutual information (PMI) (Church and Hanks, 1990). Vilar and Federico (2021) propose a statistical extension of BPE (S-BPE) that scores a pair using its count and a smoothed lower-bound term on the likelihood increase from merging it. Nouri (2026) combines a z-statistic under an independence null with an explicit compression-aware gain. The PMI-BPE tokenizer we study reweights each candidate's frequency by an exponential of its *positive* PMI, with a strength β that recovers greedy BPE at $\beta{=}0$. S-BPE is a prior method. PMI-BPE is our controlled exponential instantiation of the broader association-reweighting idea, introduced as an experimental condition rather than a tokenizer contribution of

this paper. A parallel strand modifies the merge objective for goals orthogonal to cohesion: Parity-Aware BPE (Foroutan et al., 2026) applies a fair-max rule to reduce cross-lingual tokenization inequality, and SA-BPE (Chizhov et al., 2026) regularizes code-tokenizer training via source attribution and merge skipping. These works share our premise, that the greedy frequency rule is a design choice rather than a fixed default, but target fairness and code rather than data-efficient language modeling of English.

Data-efficient pretraining and BabyLM. The BabyLM challenge (Warstadt et al., 2023; Choshen et al., 2026) asks whether pretraining can be made sample-efficient on a fixed, developmentally plausible corpus of roughly 10M or 100M words, and it explicitly welcomes analysis and negative results. We use the Strict-Small (10M-word) corpus and training setting, and measure grammatical competence with BLiMP (Warstadt et al., 2020), a suite of minimal-pair paradigms scored zero-shot through the language-model evaluation harness (Biderman et al., 2024). Most BabyLM entries hold the tokenizer fixed and vary architecture, data curation, or objective. We instead make the tokenizer the independent variable and test whether association-based merging helps in this low-resource setting.

Tokenization and language-model quality. Whether tokenizer choice measurably affects downstream quality remains contested. Gallé (2019) attributes BPE's effectiveness to the shorter sequences it produces, Rust et al. (2021) show that tokenizer fit strongly affects multilingual models, and Bostrom and Durrett (2020) find BPE suboptimal relative to a unigram tokenizer for pretraining. On intrinsic evaluation, Zouhar et al. (2023) cast tokenization as a noiseless channel, while Goldman et al. (2024) and Schmidt et al. (2024) caution that compression correlates only weakly with downstream accuracy. Mielke et al. (2021) survey the broader design space. Our aggregation analysis compares type-uniform, merge-application-weighted, and final-token-weighted summaries. It contributes an intrinsic measure tied to how often an association-reweighted merge rule is applied during encoding, and to how sensitive that diagnostic is to the association measure used to score it.

3 Method

We compare three tokenizers that differ only in the criterion used to score candidate merges during BPE training. All three share the same greedy agglomerative procedure of Gage (1994) and Sennrich et al. (2016). Starting from a Unicode-character vocabulary, we repeatedly select the highest-scoring adjacent pair, add it as a new symbol, and re-count. We run 15,598 merge operations, corresponding to a target vocabulary size of $K = 16\text{k}$. The merge score is the only deliberate intervention. Any downstream difference follows from which merges are formed and in what order (and their consequences for vocabulary size, sequence length, and optimizer-step count), not from the encoding mechanism: each produces a standard merge table applied greedily at encode time, identically across all three.

3.1 Three merge criteria

Let $N(a, b)$ be the count of the adjacent pair (a, b) at the current stage of training, tallied within words only and weighted by word frequency, so no pair spans a word boundary. We define the positive pointwise mutual information (PPMI) of a pair as

$$\mathrm{PPMI}(a, b) = \max\left(0,\ \log \frac{P(a, b)}{P_L(a)\, P_R(b)}\right), \tag{1}$$

the standard clipping of pointwise mutual information (Church and Hanks, 1990), where P_L and P_R are the left and right positional pair-marginals. Every observed pair remains eligible for merging. Let $\mathcal{R} = \{(x, y) : N(x, y) \geq 5\}$ be the retained pairs and $T = \sum_{(x,y)\in\mathcal{R}} N(x, y)$. For $(a, b) \in \mathcal{R}$, the three probabilities are estimated only over $\mathcal{R}$: $P(a, b) = N(a, b)/T$, $P_L(a) = \sum_{y:(a,y)\in\mathcal{R}} N(a, y)/T$, and $P_R(b) = \sum_{x:(x,b)\in\mathcal{R}} N(x, b)/T$. A pair outside $\mathcal{R}$ (that is, with $N(a, b) < 5$) receives $\mathrm{PPMI} = 0$, so its score reduces to the frequency term $N(a, b)$. The held-out merge-association diagnostic below applies the same count-five threshold independently to development counts. All three criteria pick the next merge as $\arg\max_{(a,b)} s(a, b)$, differing only

in the score s:

$$\text{Standard BPE:} \quad s(a,b) = N(a,b), \tag{2}$$

$$\text{PMI-BPE:} \quad s(a,b) = N(a,b)\, e^{\beta\, \mathrm{PPMI}(a,b)}, \tag{3}$$

$$\text{S-BPE:} \quad s(a,b) = N(a,b)\big[\log p_n(ab) - \log p_n(a) - \log p_n(b)\big]. \tag{4}$$

PPMI clips PMI at zero. In Eq. 4, $N(a,b)$ is the number of adjacent (a,b) occurrences the merge consumes, which equals the count of the newly formed unit ab (for overlapping same-symbol pairs we use the non-overlapping application count). The $p_n(\cdot)$ are unigram probabilities computed from the *post-merge* counts, with add-one (Laplace) smoothing, following Vilar and Federico (2021).

Standard BPE (Eq. 2) is the greedy compression baseline: it merges the most frequent pair, a purely count-driven objective that ignores whether the two symbols are statistically associated beyond what their marginal frequencies predict.

PMI-BPE (Eq. 3) reweights the frequency score by a Boltzmann/exponential factor in positive PMI, so that among pairs of comparable frequency the more *cohesive* pair (higher PPMI) is preferred. A strength parameter β controls the cohesion prior: $\beta = 0$ reduces Eq. 3 exactly to standard BPE (Eq. 2), and larger β trades compression for association. We take this exponential form as a representative instantiation of the PMI-reweighting family. Published members reweight or gate merges by related association signals (Nouri, 2026) through different specific functionals. We fix $\beta = 0.5$ and do not sweep it on the BabyLM corpus (a limitation). This fixed, untuned operating point tests whether a cohesion prior provides any benefit on a 10M-word budget.

S-BPE (Eq. 4) is the statistical BPE extension of Vilar and Federico (2021). It scores the merge of (a,b) by a lower-bound term on the corpus log-likelihood increase from treating ab as a unit: the pair count times the change in unigram log-probability, computed from the post-merge unigram counts with add-one smoothing (their Eq. 7 and Appendix A). We faithfully implement the published S-BPE merge score. A common shortcut instead scores pairs by their pre-merge bigram PMI. That is a distinct approximation, and we found the two yield measurably different merge tables, so we report the faithful criterion here. Under our fixed-budget stopping, all 15,598 selected merges have a positive S-BPE score, so fixed-budget stopping does not take the algorithm beyond the zero-score point.

3.2 Merge-association metric and its aggregations

To characterize what each merge table encodes about association, we score every merge by the PPMI of its pair *at merge time*, evaluated on held-out text. For the m-th merge, forming pair (a_m, b_m), we compute $c_m = \mathrm{PPMI}(a_m, b_m)$ using bigram and unigram statistics of (a_m, b_m) on the held-out dev slice under the partial merge table in effect just before merge m is applied. To suppress unstable estimates for rare held-out pairs, we score a pair whose held-out bigram count is below five as zero (Appendix E reports the ranking's sensitivity to this choice). This measures, for each merge the tokenizer chose to create, how strongly its two constituents co-occur beyond chance on text the tokenizer did not train on.

We aggregate the per-merge scores $\{c_m\}$ three ways. The **type-uniform** aggregation is the plain mean over merges,

$$\bar{c}_{\text{type}} = \frac{1}{M} \sum_{m=1}^{M} c_m, \tag{5}$$

which counts every merge *type* once, regardless of how often it is used. The **merge-application-weighted** aggregation weights each merge by how often its rule is applied when the dev text is tokenized,

$$\bar{c}_{\text{app}} = \frac{\sum_m w_m\, c_m}{\sum_m w_m}, \tag{6}$$

where w_m is the number of times merge rule m is applied while encoding the dev slice. This is *not* the number of final token occurrences: a final token can absorb several merge applications, base symbols involve none, and the model consumes only final token IDs, never the merge rules. As an intrinsic proxy for the token level, we also report a **final-token-weighted** aggregation,

$$\bar{c}_{\text{token}} = \frac{\sum_t f_{\text{dev}}(t)\, g(t)}{\sum_t f_{\text{dev}}(t)}, \tag{7}$$

where $f_{\text{dev}}(t)$ is the dev frequency of final token type t and $g(t)$ is the association c_m of the merge that forms t, using the highest-priority creating

merge when a string is reachable by more than one merge order (base tokens contribute g=0). Even this treats t as an atomic ID and so remains an intrinsic proxy.

What the aggregations do and do not measure. None of these statistics is a quantity the language model observes or optimizes: the model is trained on final token IDs, not on merge rules or their derivations. A merge rule applied often during encoding need not correspond to a frequent final token, so we report all three as intrinsic diagnostics, with the scoring measure stated, rather than as "what the model sees." The gap between $\bar{c}_{\text{type}}$ and the usage-weighted forms is nonetheless diagnostic: if a tokenizer's association advantage lives in rarely-applied merges, it contributes little to $\bar{c}_{\text{app}}$ or $\bar{c}_{\text{token}}$. All three aggregations, under both held-out scoring measures, are reported in Table 2; the type-uniform ordering reverses between them (Section 6). Our two evaluation axes, BPB and BLiMP, are reported in Table 3.

4 Experimental Setup

Data. We use the BabyLM-2026 Strict-Small corpus (Choshen et al., 2026), a fixed budget of 10M words drawn from six developmental and naturalistic English sources: CHILDES (2.84M words), Project Gutenberg (2.56M), OpenSubtitles (2.28M), Simple Wikipedia (1.53M), the spoken portion of the BNC (0.76M), and Switchboard (0.02M). This mixture mirrors the quantity and character of language a child is exposed to. The Strict-Small setting fixes the word budget, so all methods are compared under identical data. The release includes no official development split. We therefore hold out every 20th corpus line as a deterministic 5% development slice. These lines are short, about nine words on average, i.e. sentence- or utterance-sized rather than document-sized. This gives 9.50M training words ($\sim$1.05M lines) and 0.50M held-out development words. Because we split by line rather than by source document, adjacent lines from the same source, for instance one CHILDES conversation, can fall on opposite sides of the split. The held-out slice may therefore share local context with training. We use it only for BPB, and treat that number as indicative rather than as an unbiased generalization estimate. BLiMP, our main external measure, is a minimal-pair benchmark and is unaffected. The dev slice is never seen during tokenizer training or language-model optimization. All three tokenizers and all language models share this exact split.

Tokenizers. We compare the three merge criteria of Section 3: standard BPE, PMI-BPE (β=0.5), and S-BPE (Vilar and Federico, 2021). All three are trained with a common budget of 15,598 merge operations (target $K = 16$k) on the 9.50M-word training split, from an identical 402-character base. This yields realized vocabularies of 15,982 / 16,000 / 15,986 token IDs (Appendix C). Realized vocabulary sizes differ by under 0.2%, and optimizer steps vary as a consequence of sequence length. Table 1 reports intrinsic properties of the three tokenizers on the dev slice; Section 5 reads them off.

Language model. For each tokenizer we train a 97M-parameter Llama-style decoder from scratch: $d_{\text{model}} = 768$, 12 layers, 12 attention heads, context length 1024, a SwiGLU feed-forward network with hidden width 2048, rotary position embeddings, RMSNorm, and tied input/output embeddings. The tied embedding is sized to each tokenizer's realized vocabulary, so totals range 97.23–97.24M parameters, differing by under 0.02% across tokenizers. Training uses AdamW (peak learning rate 3×10^{-4}, cosine decay to $0.1\times$ peak, warmup set to 10% of each run's optimizer steps ($\sim$185), weight decay 0.1, gradient clipping 1.0) in bfloat16 on a single NVIDIA RTX 5090, with an effective batch of 65,536 tokens per step (micro-batch 16 $\times$ context 1024 $\times$ gradient accumulation 4). Each model performs $\sim$10 passes over the fixed training text, roughly 1,850 steps. We set `max_steps` *per tokenizer* so that every tokenizer is trained on the fixed 9.50M-word text the same number of times ($\sim 10\times$), matched up to a dropped final partial block (Appendix A). Because PMI-BPE encodes the corpus into more tokens, holding the word budget fixed means its longer sequences cost proportionally more optimizer steps. This is a compute cost of the longer sequences. We do not correct for it, because the quantity we hold constant is the linguistic data. Appendix A reports the exact per-tokenizer step counts (PMI-BPE takes about 3.5% more steps than standard BPE). We train 3 random seeds per tokenizer, for 9 language-model runs in total, and report mean and sample standard deviation across seeds.

Evaluation. We evaluate on two axes. First, *BPB* on the held-out 0.50M-word dev slice, computed as the summed per-token negative log-likelihood

Tokenizer	B/tok ↑	Fert.	%split
Standard BPE	**4.24**	1.255	16.3
PMI-BPE (β=0.5)	4.09	1.300	18.0
S-BPE	4.20	1.266	16.4

Table 1: Compression properties on the held-out dev slice (target K=16k). Fert. is tokens per whitespace word and %split the fraction of words split into more than one token. Higher bytes/token is better compression. Merge-association summaries are in Table 2.

Measure	Standard	PMI-BPE	S-BPE
PPMI, type-unif.	4.08	3.82	3.97
PPMI, app-wtd	2.131	2.133	2.130
PPMI, token-wtd	3.40	3.22	3.16
Post-merge PMI, type-unif.	5.66	6.24	6.00
Post-merge PMI, app-wtd	11.34	11.42	11.51
Post-merge PMI, token-wtd	4.98	4.91	4.75

Table 2: Mean merge association under two held-out scoring measures (thresholded PPMI and post-merge PMI), aggregated type-uniformly, by merge-application count, and by final-token frequency. Only the type-uniform ordering fully reverses between the two measures (see text).

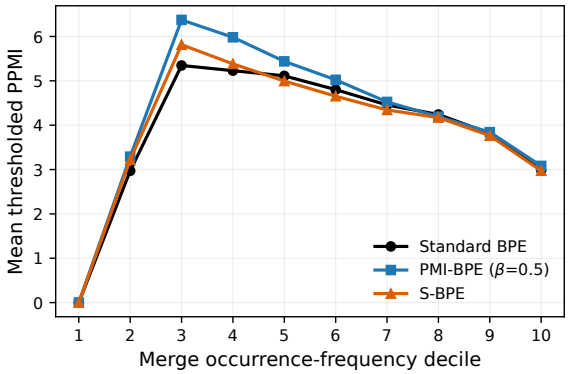


Figure 1: Mean held-out thresholded PPMI (minimum count five) of merges that fire on the dev slice, binned into deciles by application count (1 = rarest applied, 10 = most). Bins pool merges across tokenizers. PMI-BPE leads across most deciles, with standard BPE slightly ahead around the eighth. S-BPE is lowest in the more frequent deciles. See text for how this relates to the type-uniform means of Table 2.

divided by (bytes $\times \ln 2$). BPB normalizes by bytes rather than tokens, so it compares models on a common scale despite their different sequence lengths and vocabulary segmentations. Second, *BLiMP* zero-shot accuracy (Warstadt et al., 2020): we score the stock `blimp` task (67 minimal-pair paradigms) in the lm-evaluation-harness (Biderman et al., 2024) via a model adapter for our from-scratch checkpoints. This is not the official `blimp_filtered` variant used for the BabyLM leaderboard, so our accuracies are not directly comparable to the official baseline. BLiMP is part of the BabyLM evaluation suite (Warstadt et al., 2023; Choshen et al., 2026). BPB is an additional tokenizer-comparable held-out language-modeling metric. BPB and BLiMP results (mean ± sample SD over 3 seeds per tokenizer) are reported in Table 3.

5 Results

Intrinsic metrics. Table 1 reports the three tokenizers on the held-out dev slice. The reweighting changes which merges are formed: the PMI-BPE and S-BPE merge tables share only 38% and 43% of their merges with standard BPE (Jaccard over the set of ordered merge rules). Standard BPE compresses best and splits the fewest words; PMI-BPE compresses worst, with the highest fertility, and S-BPE sits close to standard.

Table 2 reports mean merge association under both held-out scoring measures and all three aggregations. No aggregation makes the reweighted tokenizers uniformly more associated. Under thresholded held-out PPMI, standard BPE ranks highest type-uniformly and PMI-BPE last, despite PMI-BPE using PPMI in its own training-time merge score, and the final-token weighting ranks standard BPE highest as well. Under merge-application weighting the spread across all three collapses to below 0.3%. Scored instead by post-merge PMI, the type-uniform ordering reverses and both reweighted tokenizers exceed standard BPE (Section 6). Figure 1 bins the merges that fire on the dev slice into deciles by application count and plots each decile's mean PPMI. It conditions on fired merges, so it excludes merges that never fire on dev, which the type-uniform mean of Table 2 includes as zeros. The reweighted tokenizers place more merges in that zero-scored tail, which is why PMI-BPE ranks above standard BPE across most deciles here yet below it in the type-uniform mean.

Language modeling. Table 3 reports BPB on the held-out slice and BLiMP zero-shot accuracy, each averaged over three seeds.

First, *no association-based tokenizer reliably changes BPB relative to standard BPE*. The three means are close: standard 1.640±0.003, S-BPE 1.636±0.007, PMI-BPE 1.647±0.003. With three seeds, all primary 95% confidence intervals for the

tokenizer differences include zero (Welch intervals over the three independent runs per tokenizer; Appendix B). S-BPE's mean is fractionally lower than standard's, but the difference (-0.004) is mixed across seeds and small relative to across-seed variation. The larger of the two differences from standard BPE is PMI-BPE's higher BPB, worse than standard by 0.007, but its 95% confidence interval still includes zero. PMI-BPE fragments most (lowest bytes/token) and also has the highest mean BPB. S-BPE fragments slightly more than standard BPE yet has a slightly lower mean BPB, so the compression-to-BPB relationship is not monotone across the three.

Per-token loss is *not* comparable across tokenizers: each scores a different number of targets over the same text, so a finer segmentation mechanically lowers the loss per token. We list NLL/token in Table 3 only as a factor of the identity BPB $=$ (NLL/token) / (bytes/scored token $\cdot \ln 2$), and draw no tokenizer-quality conclusion from it. BPB normalizes by *bytes* and is the tokenizer-comparable quantity. In that identity, PMI-BPE spends more token predictions per byte, and the mechanically easier per-token predictions do not offset them, so its mean BPB sits above standard BPE's; whether they offset is an empirical balance, not a consequence of segmentation alone, as S-BPE shows.

Second, *BLiMP differences are inconclusive*. All three tokenizers score within about one point of one another (57.3–57.8%), with overlapping across-seed variation. On the point estimate PMI-BPE is highest and S-BPE lowest, the reverse of the BPB ordering. But with three seeds we detect no reliable difference (a paradigm-level paired bootstrap gives 95% intervals that include zero; Appendix B). BLiMP is a minimal-pair benchmark, not normalized by bytes and less directly coupled to compression. It too provides no reliable evidence of an association benefit.

Tokenizer	NLL/tok†	BPB ↓	BLiMP ↑
Standard BPE	4.82	1.640 ± 0.003	57.5 ± 0.5
PMI-BPE (β=0.5)	4.67	1.647 ± 0.003	57.8 ± 0.9
S-BPE	4.77	1.636 ± 0.007	57.3 ± 0.2

Table 3: Language-model results: BPB and BLiMP zero-shot accuracy, mean ± sample standard deviation over 3 seeds. †NLL/token (nats; across-seed mean) is *not* comparable across tokenizers and is listed only as a factor of the identity BPB $=$ (NLL/token) / (bytes/scored token $\cdot \ln 2$) (Appendix C). No value is boldfaced: no pairwise difference is significant at three seeds (Section 5).

6 Discussion

Our result is negative, within the scope we test: on the 10M-word Strict-Small corpus (9.5M words for training), at one merge budget, one operating point per association score, and two evaluation axes, association-reweighted tokenization does not reliably improve data-efficient language modeling. With three seeds neither alternative differs reliably from standard BPE on either axis (Appendix B). On BPB, S-BPE's slightly lower mean falls within across-seed variation, and PMI-BPE's higher mean is the larger of the two differences from standard BPE, but its 95% interval still includes zero. On BLiMP the differences are inconclusive.

A single intrinsic diagnostic cannot explain the negative result, because the intrinsic evidence is too measure-dependent. The reweighting does reshape the merge table, but whether that counts as higher association depends on the measure. Scored by the post-merge PMI component of S-BPE's score (with no count threshold) on held-out text, both reweighted tokenizers exceed standard BPE type-uniformly, and both remain above standard under application weighting, though PMI-BPE and S-BPE exchange order (Table 2). Under our thresholded held-out PPMI diagnostic (count threshold five), the type-uniform ordering reverses: standard BPE scores highest, and PMI-BPE ranks last despite incorporating PPMI in its training-time merge score. The application-weighted spread across all three tokenizers then falls to a fraction of a percent (Eq. 6). The two measures disagree partly because the reweighted tokenizers concentrate merges on rare pairs assigned high scores by the smoothed post-merge measure: those pairs are assigned zero by our minimum-count threshold under thresholded positive PMI, and either way their rules are applied seldom. Intrinsic association is therefore too sensitive to the scoring measure to explain the absence of a reliably detected difference on either axis, and type-uniform association in particular is an unreliable summary.

The observed PMI-BPE point-estimate difference reflects a compression–prediction tradeoff. BPB is the tokenizer-comparable held-out metric used here: unlike per-token loss, which is not comparable across segmentations because each tokenizer scores a different number of targets, BPB is

expressed per byte. PMI-BPE's finer segmentation spends more token predictions per byte, the easier per-token predictions it affords do not offset them, and by BPB it has the worse mean (Table 3).

Two implications follow for small-data tokenizer design. First, intrinsic association is an unreliable predictor of measured model quality. Reported type-uniformly, it can point in either direction depending on the association measure. Its application-weighted form differs by about 0.14% across tokenizers (PMI-BPE marginally highest), and its final-token-weighted form ranks standard BPE highest. No intrinsic summary matches both point-estimate orderings: application-weighted PPMI happens to match the BLiMP point-estimate ordering, but its spread is below 0.3% and the BLiMP differences are inconclusive. Intrinsic association should be reported and read with the scoring measure stated. Second, we test PMI-BPE at a single fixed strength (β=0.5) and S-BPE at the published merge score with fixed-budget stopping. A strength or stopping rule calibrated on larger corpora should not be assumed to carry over, and the useful setting, if any, would need to be re-tuned for each corpus size. Association-reweighted tokenization is not ruled out in principle. One possibility is to concentrate association in the merges that become frequent final tokens, rather than in the rare pairs a PMI prior most rewards.

7 Conclusion

At the tested operating points (a shared 15,598-merge budget targeting K=16k, PMI-BPE at β=0.5, and S-BPE with fixed-budget stopping), neither association-reweighted merge rule yields a reliably detected improvement over standard BPE for our 97M-parameter models on the BabyLM Strict-Small corpus, on either held-out BPB or BLiMP. With three seeds we detect no significant difference among the three tokenizers on BPB or BLiMP. PMI-BPE's higher mean BPB is the larger of the two differences from standard BPE, but its 95% interval includes zero. The easier per-token predictions its finer segmentation affords do not offset the additional token predictions per byte it requires. The reweighting does change the tokenizer, so the absence of a reliably detected difference on either axis is not explained by identical tokenizations. We were unable to trace it to a specific intrinsic cause, because the tokenizers' ranking by association varies with both the measure and the aggregation: the type-uniform ordering reverses between post-merge PMI and thresholded PPMI. For small-data tokenizer design, this cautions against treating intrinsic association, and type-uniform association in particular, as a predictor of language-model quality. One possibility is to concentrate association in the merges that become frequent final tokens.

Limitations

Our study is narrow in scope. It covers a single language (English) and a single model scale (97M parameters). Larger models, or morphologically richer languages where association may align better with meaningful units, could behave differently. We compare three merge criteria at one merge budget (target K=16k). Vocabulary size is an influential hyperparameter that we neither swept nor tuned per merge criterion, so we cannot exclude that the three rank differently at another budget. We performed no hyperparameter tuning of any kind, on the tokenizers or the language models. PMI-BPE is tested at a single strength (β=0.5) and S-BPE at a fixed-budget stopping rule rather than its published likelihood-gain stopping criterion, so we cannot exclude that another operating point behaves differently on this corpus. With only three runs per tokenizer, the confidence intervals are wide. The zero-spanning intervals do not establish equivalence and remain compatible with small effects in either direction. Our intrinsic association findings are measure-relative: the ordering of tokenizers by merge association reverses between a thresholded positive-PMI measure and post-merge PMI, so we report the BPB and BLiMP results as primary and treat the intrinsic diagnostics as descriptive. The training-time PMI bonus uses a fixed count-five cutoff that we do not sweep. Appendix E varies only the held-out diagnostic cutoff and shows the intrinsic ranking is sensitive to that evaluation choice. The held-out development slice is split by line rather than by source, so its BPB may share local context with training (Section 4). BLiMP, our main external measure, is unaffected by this train/development split. Our two evaluation axes are held-out language modeling (BPB) and grammatical minimal-pair accuracy (BLiMP); neither is a broad downstream-task evaluation. Other BabyLM evaluations, including the fine-tuning suite, are left to future work. Finally, WordPiece and other likelihood-based association

criteria are discussed but not trained end-to-end in the language-modeling comparison.

References

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, and 11 others. 2024. Lessons from the trenches on reproducible evaluation of language models. *arXiv preprint arXiv:2405.14782*.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624.

Pavel Chizhov, Egor Bogomolov, and Ivan P. Yamshchikov. 2026. From where words come: Efficient regularization of code tokenizers through source attribution. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 39053–39073.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Negar Foroutan, Clara Meister, Debjit Paul, Joel Niklaus, Sina Ahmadi, Antoine Bosselut, and Rico Sennrich. 2026. Parity-aware byte-pair encoding: Improving cross-lingual fairness in tokenization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7514–7538.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

Matthias Gallé. 2019. Investigating the effectiveness of BPE: The power of shorter sequences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1375–1381.

Omer Goldman, Avi Caciularu, Matan Eyal, Kris Cao, Idan Szpektor, and Reut Tsarfaty. 2024. Unpacking tokenization: Evaluating text compression and its correlation with model performance. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2274–2286.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *arXiv preprint arXiv:2112.10508*.

Azam Nouri. 2026. Significance-gain pair encoding for LLMs: A statistical alternative to frequency-based subword merging. *arXiv preprint arXiv:2603.19261*.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135.

Craig W. Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. 2024. Tokenization is more than compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 678–702.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

David Vilar and Marcello Federico. 2021. A statistical extension of byte-pair encoding. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 263–275.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 1–34.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, and 12 others. 2016.

Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. 2023. Tokenization and the noiseless channel. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5184–5207.

A Per-tokenizer exposures

Table 4 gives the exact training exposure of each tokenizer. All three are trained for ∼10 passes over the same 9.50M-word training text, so *word* exposure is matched up to a dropped final partial block (87, 371, and 700 tokens for standard BPE, PMI-BPE, and S-BPE, each <0.006% of tokens, with the dropped suffix differing by segmentation). They are not matched on compute. A tokenizer that fragments more encodes the fixed word budget into more tokens, and therefore performs more optimizer steps at a fixed effective batch of 65,536 tokens. PMI-BPE takes 1,884 steps to standard BPE's 1,820 (about 3.5% more), and S-BPE 1,834 (about 0.8% more). We hold the linguistic data constant and do not correct for this. The extra steps are a cost of longer sequences. BPB does not normalize for this additional training compute. The comparison is word-exposure-matched rather than compute-matched.

Tokenizer	Train tok.	Vocab	Steps	Pos.	Ep.
Standard BPE	11.93M	15,982	1,820	119.3M	10.0
PMI-BPE (β=0.5)	12.35M	16,000	1,884	123.5M	10.0
S-BPE	12.02M	15,986	1,834	120.2M	10.0

Table 4: Per-tokenizer training exposure. "Train tok." is the encoded training token count. "Steps" is the optimizer steps for 10 epochs at an effective batch of 65,536 tokens. "Pos." is the total token positions processed (steps × batch). "Ep." is the epochs over the fixed 9.50M-word text. Word exposure is matched up to a sub-block tail. Optimizer steps are not.

B Per-seed results and significance

Table 5 lists all nine runs. Because the three tokenizers have different vocabularies, matched seed indices do not guarantee matched initialization or data order. Our primary analysis therefore uses unpaired Welch intervals over the three independent runs per tokenizer, and reports the matched-seed (paired) differences only as exploratory. All primary 95% confidence intervals for the two differences from standard BPE include zero (Welch t intervals with Welch–Satterthwaite degrees of freedom). For BPB, the Welch mean difference from standard BPE is $+0.007$ for PMI-BPE (95% CI $[-0.001, +0.015]$) and -0.004 for S-BPE (95% CI $[-0.019, +0.011]$). For BLiMP, the Welch differences are $+0.4$ points for PMI-BPE (95% CI $[-1.5, +2.3]$) and -0.2 for S-BPE (95% CI $[-1.5, +1.1]$). A paradigm-level paired bootstrap (10,000 resamples over the 67 paradigms, averaged across seeds, which does not propagate seed uncertainty) gives tighter but still zero-spanning intervals, $[-0.9, +1.7]$ and $[-1.1, +0.6]$ respectively. The remaining pair, PMI-BPE versus S-BPE, also spans zero (BPB $+0.011$, 95% CI $[-0.004, +0.026]$; BLiMP $+0.6$ points, 95% CI $[-1.7, +2.8]$), so no pairwise difference is reliable. All Welch intervals are computed from unrounded per-seed scores. The exploratory matched-seed analysis agrees: the only directionally consistent effect is PMI-BPE's higher BPB, higher at each of the three matched seed indices, whose interval nonetheless includes zero.

C Tokenizer implementation

All three tokenizers share one implementation, differing only in the merge score (Section 3). **Initialization:** the base vocabulary is the set of Unicode characters observed in the training corpus. Each whitespace-delimited word is suffixed with a four-character end-of-word marker `</w>` and split into characters, so the marker is itself built from the characters `<`, `/`, `w`, `>`. The base is identical across all three tokenizers (402 characters). **Normalization:** runs of whitespace are collapsed to single spaces and leading/trailing whitespace is stripped. There is no lowercasing or Unicode normalization, so case is preserved. **Word boundaries:** BPE is applied within words only and merges never cross whitespace. Decoding concatenates symbols and replaces `</w>` with a space. **Special tokens:** the tokenizer adds none, and at encoding time an unseen symbol maps to id 0. Seven dev word-tokens (of ∼0.50M) contain a character unseen in training and are mapped this way. Excluding those lines changes BPB by under 0.0001 (they are 0.004% of dev bytes), below the precision we report. BLiMP contains no character outside the training base, so no BLiMP token is out-of-vocabulary. The corpus contains no literal occurrence of the marker string `</w>`. **Merge selection:** at each step the highest-

scoring adjacent pair is merged, with ties broken deterministically by symbol-id order (a min-heap keyed by $(-s(a,b), a, b)$). There is no candidate-elimination floor, so every observed pair stays eligible. In PMI-BPE, a pair with $N(a,b) < 5$ receives $\mathrm{PPMI} = 0$, reducing its score to the ordinary frequency term, and the held-out diagnostic independently applies the same count-five cutoff. **Logarithms** are natural throughout (the merge scores and the association metric). The PPMI uses a count threshold of five and clips at zero on the held-out dev slice. **Realized vocabulary:** all three target $K = 16{,}000$ from the identical 402-character base, but the realized number of distinct token IDs differs slightly (15,982 / 16,000 / 15,986). A few merges produce a symbol string reachable by more than one merge order and so add no new ID. We report the realized sizes. The $<0.2\%$ difference is small relative to the merge budget rather than provably without effect. **BLiMP scoring:** we use lm-evaluation-harness v0.4.11 over the standard `blimp` group (67 paradigms, 1,000 minimal pairs each). We sum each sentence's conditional token log-likelihoods from the second token onward. Because the model has no BOS, EOS, or special token, the first token is unscored, and the delimiter space is dropped by whitespace pre-tokenization. A pair counts as correct when the grammatical sentence has the higher summed log-likelihood, and accuracy is the equally-weighted mean over paradigms. In all 67,000 pairs the two sentences share the same first token under every tokenizer, so the unscored first token cancels exactly and cannot bias the comparison.

Language-model inputs. The encoded training corpus is a single concatenated token stream, chunked into non-overlapping 1024-token blocks. Attention is causal within a block and does not cross block boundaries, so a block may span multiple corpus lines and sources. Training blocks are shuffled and the final incomplete block is dropped, with no padding. For BPB we process the concatenated dev stream in 1024-token contexts with stride 1024, each context predicting its following 1024 tokens (so the first prediction in each window uses short context, identically for all three tokenizers). The last target of a window is the first context token of the next, so no target is scored twice, and the final partial window is retained with its padded positions masked out of the loss. Every dev token except the first is therefore scored exactly once: 623,151, 645,468, and 628,317 scored targets for standard BPE, PMI-BPE, and S-BPE. NLL per token is the summed target NLL divided by these scored targets. BPB is the summed target NLL in bits divided by the UTF-8 byte count of the scored targets: the total dev byte count minus the first token's five bytes, 2,640,322 for all three tokenizers (the dev text and its shared first token are identical across tokenizers). Only the first token is unscored, so the Table 3 decomposition BPB = (NLL/token) / (bytes/scored token $\cdot \ln 2$) is an exact identity. The model uses no dropout. RMSNorm uses ϵ=10^{-5}, and AdamW uses β_1=0.9, β_2=0.95, ϵ=10^{-8}. The corpus is the Hugging Face dataset `BabyLM-community/BabyLM-2026-Strict-Small` (`default` configuration, revision `c92ab16`).

Tokenizer	Seed	BPB ↓	BLiMP (%) ↑
Standard BPE	0	1.6378	57.41
	1	1.6440	58.03
	2	1.6385	56.97
PMI-BPE	0	1.6489	56.78
	1	1.6488	58.38
	2	1.6428	58.38
S-BPE	0	1.6280	57.13
	1	1.6393	57.24
	2	1.6414	57.45

Table 5: Per-seed BPB and BLiMP accuracy for all nine runs.

D Merge association under multiple measures

Table 2 reports each tokenizer's mean merge association under the two scoring measures we use: thresholded held-out PPMI (count threshold five, clipped at zero), and post-merge PMI (the association component of S-BPE's score, $\log p_n(ab) - \log p_n(a) - \log p_n(b)$, with no threshold), each aggregated type-uniformly, by merge-application count ($\bar{c}_{\text{app}}$), and by final-token frequency ($\bar{c}_{\text{token}}$). The intrinsic ranking depends on both the measure and the aggregation. Under PPMI, standard BPE has the highest association type-uniformly and by final-token weight, and application weighting reduces the spread to near zero. Under post-merge PMI the type-uniform ordering fully reverses. Under application weighting, both reweighted tokenizers move above standard BPE, with S-BPE highest. The final-token aggregation again ranks standard BPE highest (4.98 vs. 4.91 and 4.75). The final-token aggregation is the closest of the

three proxies to the model's final input units. It thus favors standard BPE under both measures, yet standard BPE does not reliably win downstream (Table 3). No intrinsic summary matches both downstream point-estimate orderings: application-weighted PPMI happens to match the BLiMP point-estimate ordering (PMI-BPE > standard > S-BPE), but its cross-tokenizer spread is below 0.3% and the BLiMP differences are inconclusive. Intrinsic association therefore describes the tokenizers but does not consistently align with downstream point estimates in this experiment. The PPMI/post-merge-PMI disagreement is partly driven by rare pairs that receive positive post-merge PMI but are assigned zero by our minimum-count rule under the thresholded held-out PPMI diagnostic. PMI-BPE places the most merges in the zero-scored tail: 1.5% never fire on dev and 19.5% fire below the count threshold, versus 0.2%/10.5% for standard BPE and 0.5%/12.9% for S-BPE. Figure 1 conditions on merges that fire (w_m>0). Its within-bin means do not determine the type-uniform mean, because the tokenizers place different proportions of their merge inventories in each frequency range, including the never-applied group.

E Count-threshold sensitivity

The type-uniform held-out PPMI ranking is itself sensitive to the minimum-count threshold (Table 6). At thresholds 1 and 3 the reweighted tokenizers score highest, led by PMI-BPE. At 5 (our setting) and 10 standard BPE does. Because the reversal occurs within the range of thresholds examined here, this reinforces that the type-uniform ranking is parameter-dependent and should not be read as a predictor of downstream quality.

Min. count	Standard	PMI-BPE	S-BPE
1	4.23	4.81	4.44
3	4.32	4.44	4.37
5	4.08	3.82	3.97
10	2.75	2.60	2.68

Table 6: Type-uniform held-out PPMI by minimum-count threshold. The threshold-5 row matches Table 2.

Retrieval Without Relevance: Prompt Dependence in Small Machine Translation Models

Heesu Kim[1], **Shota Sato**[2], **Mamoru Komachi**[2]
[1]Sungkyunkwan University [2]Hitotsubashi University
k.heesu@skku.edu
{shota,komachi}@scl.sds.hit-u.ac.jp

Abstract

We study how a small machine translation model should allocate a fixed data budget between parametric training and retrieval memory. We define a shared 100M-token corpus-access ledger that charges pretraining, fine-tuning, and retrieval-datastore tokens alike. Using ≈42M-parameter models, we compare three uses of the same per-pair datastore across 12 directed pairs of English, Chinese, Japanese, and Korean: leaving it unused, training on it, or retrieving from it into the prompt. We evaluate retrieval additivity across all 12 pairs and analyze retrieval-conditioned behavior in a four-pair core set. Retrieval shows no consistent additive gain over simply not using the datastore, a result that persists under an off-budget dense retriever and pools up to approximately 1000× larger. On the four-pair core set, retrieval-trained models are nevertheless highly context-dependent: removing retrieved examples causes large score drops, while replacing them with query-unrelated, count-matched same-pair examples recovers nearly the full score. This suggests that the learned dependence is primarily on exemplar-shaped context rather than on query-specific information carried by the retrieved neighbors. Our training and evaluation code is available at https://github.com/kheesu/retrieval-without-relevance.

1 Introduction

Retrieval-augmented systems are often evaluated by holding a model's training data fixed while granting access to an additional datastore. Such comparisons show whether retrieval helps with extra corpus access, but not whether the same data would be more useful if incorporated into the model's parameters. Prompt-level retrieval introduces a second attribution problem: any observed effect may arise from the information in the selected neighbors or from the model's dependence on the presence of exemplar-formatted context. It therefore remains unclear whether retrieval functions as useful external memory or merely changes the prompt interface that the model learns to expect.

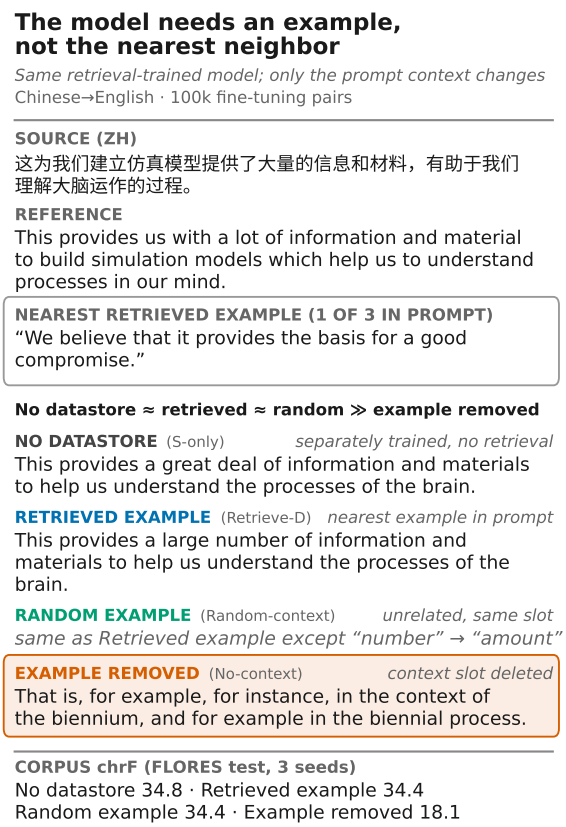


Figure 1: **Query-matched neighbor identity is unnecessary in this example, but exemplar-shaped context is required.** For a held-out FLORES zh→en example, the retrieval-trained RETRIEVE-D model translates correctly whether conditioned on its nearest datastore match or on a query-unrelated, count-matched same-pair example, but degenerates once the demonstration slot is removed. An independently trained S-ONLY model matches these outputs without ever seeing the datastore. Selection protocol in Appendix H.

We study these questions in machine transla-

Proceedings of the Second BabyLM Workshop, pages 105–122
October 29, 2026 ©2026 Association for Computational Linguistics

tion, where parallel sentence pairs can be used directly either for supervised training or as retrieved prompt context. Using ≈42M-parameter decoder-only models, we evaluate all 12 directed pairs among English, Chinese, Japanese, and Korean under a shared 100M-token corpus-access budget covering pretraining, fine-tuning, and datastore tokens. The budget is the BabyLM Challenge's 100M-word data limit (Warstadt et al., 2023), extended so that a retrieval datastore is charged like any other corpus (Section 3.2); we ask the challenge's question—what a fixed, small corpus is best spent on—for a system that can hold data outside its weights. For each pair, the same datastore is either left unused, incorporated through supervised training, or retrieved into the prompt. To distinguish neighbor information from interface dependence, we additionally evaluate the retrieval-trained checkpoints after removing the retrieved context or replacing it with count-matched random examples.

This evaluation yields three main findings. First, prompt-level retrieval exceeds the no-datastore control by at most +0.5 chrF across 36 pair–budget cells and is non-positive in 30. This pattern persists with a dense retriever and retrieval pools approximately three orders of magnitude larger. Training on the datastore, by contrast, usually performs best in the most data-scarce regime, improving over the no-datastore control by +5.8 to +18.1 chrF on 10 of 12 pairs at the 1k-pair budget, with gains diminishing as supervised data increases.

Second, on the four-pair core set where we ran inference-time probes, retrieval-trained models nevertheless depend strongly on their prompt interface: removing context causes losses of up to 16.4 chrF, whereas query-unrelated same-pair examples—count-matched on all four pairs and exactly length-matched on en↔zh—recover nearly the full score. The models therefore learn to require exemplar-shaped context rather than to use relevant neighbors.

Third, whenever retrieval outperforms training on the datastore, the latter falls below the no-datastore control, showing how baseline degradation can be mistaken for retrieval value. Our claims are limited to prompt-level retrieval in small MT models trained from scratch.

Our contributions are:

- a shared corpus-access ledger and no-datastore control for comparing training and retrieval under a fixed data budget;
- evidence across 12 directed translation pairs that prompt-level retrieval shows no consistent gain over leaving the datastore unused, whereas incorporating the datastore through fine-tuning primarily helps when supervision is scarce;
- random-context controls on the four-pair core set, plus an exact length-matched control on en↔zh, showing strong dependence on retrieval-shaped context but weak dependence on query-relevant neighbors; and
- an artifact-qualified attribution analysis illustrating how a degraded datastore-trained baseline can produce an apparent retrieval gain.

2 Related Work

Retrieval as external memory. kNN-LM (Khandelwal et al., 2020) and kNN-MT (Khandelwal et al., 2021) interpolate a parametric model with a nearest-neighbor datastore at the token level; RAG (Lewis et al., 2020) and REALM (Guu et al., 2020) retrieve passages into a generation or masked-LM objective; RETRO (Borgeaud et al., 2022) conditions a language model on retrieved chunks during pretraining; and MassiveDS (Shao et al., 2024) treats datastore scale itself as an axis of improvement. These results are typically reported without charging the datastore against the model's training data budget, so a gain from retrieval and a gain from more data are not distinguished. We adopt the substitution framing but make the accounting explicit: under a fixed corpus budget, datastore tokens are charged to the same ledger as training tokens, so retrieval competes with training for data rather than adding to it.

Retrieval-augmented machine translation. Translation memory and example retrieval have a long history in MT: neural systems have retrieved and fused matched translation-memory segments (He et al., 2021), retrieved monolingual target-side memory (Cai et al., 2021), and used contrastive retrieval to select memory examples (Cheng et al., 2022). Hao et al. (2023) show that naively retrieved translation examples can backfire in low-resource settings, motivating our no-datastore and random-content controls rather than an assumption that retrieval is neutral-or-better by default. Dictionary-augmented NMT (Ghazvininejad et al., 2023) targets rare or unseen words specifically; we adapt this as a secondary, more narrowly scoped arm.

Random demonstrations as a control. Min



one 100M-token ledger per directed pair

P: Wikipedia pretraining (78–84M tokens)
S: fine-tuning pairs, 1k $\subset$ 10k $\subset$ 100k (+1k val.)
D: datastore, disjoint from S, capped to remainder
L: dictionary (secondary arm, App. F)

Condition	S	D	Context
S-ONLY	train	—	none
TRAIN-ON-D	train	train	none
RETRIEVE-D	train	prompt	top-3 retrieved
NO-CONTEXT	RETRIEVE-D ckpt		none
RANDOM-CONTEXT	RETRIEVE-D ckpt		3 random same-pair

Figure 2: **Data pools and conditions.** Top: resources accessible to each condition are charged against a single 100M-token ledger; the diagram shows the pretraining pool P, fine-tuning pool S, datastore D, and (dictionary arm only) lexicon L. Bottom: how each condition uses S and D. All conditions share P through one pretrained base per pair. NO-CONTEXT and RANDOM-CONTEXT are inference-only probes on the trained RETRIEVE-D checkpoint.

et al. (2022) show that in-context learning is often insensitive to the correctness of demonstration labels: replacing gold labels with random ones barely hurts few-shot performance, implicating the format, input distribution, and label space of the demonstrations rather than their content. This complements work that instead retrieves or selects demonstrations to improve in-context learning (Liu et al., 2022; Rubin et al., 2022); our random-context control asks the analogous relevance question for neighbor identity in a retrieval-conditioned regime and finds the same qualitative answer.

3 Evaluation Framework

3.1 Allocation Systems and Contrasts

Each directed pair has three disjoint data pools, all charged to one ledger (Figure 2): a pretraining pool P, a parallel fine-tuning pool S, and a parallel datastore D. We compare three uses of D: leaving it unused (S-ONLY), incorporating it into supervised fine-tuning (TRAIN-ON-D), or using it for prompt-level retrieval during fine-tuning and inference (RETRIEVE-D). The three trained systems share the same architecture, tokenizer, and frozen decoding configuration (Section 4), but differ in how they use the datastore. NO-CONTEXT and RANDOM-CONTEXT reuse the trained RETRIEVE-D checkpoint and modify only its evaluation-time context.

RETRIEVE-D − S-ONLY measures the additive value of making the datastore available through retrieval. TRAIN-ON-D − S-ONLY measures the effect of incorporating it into supervised fine-tuning, while TRAIN-ON-D − RETRIEVE-D is the direct allocation contrast. NO-CONTEXT measures dependence on the trained prompt interface, whereas RANDOM-CONTEXT tests whether that dependence requires query-relevant neighbors. These contrasts assume a shared accounting of data, which we define in the next section.

3.2 Shared data ledger

All model-facing data are charged to one ledger: pretraining tokens, supervised fine-tuning tokens, datastore tokens, and dictionary tokens.

$$B_c \;=\; N(P) + N(S_c) + I_c^D\, N(D) + I_c^L\, N(L) \;\le\; 100\text{M},$$

where $N(X)$ is the number of BPE token occurrences in the deduplicated records of resource X, counted over both source and target sides; P, S_c, D, and L are the pretraining, fine-tuning, datastore, and dictionary resources; and I_c^D, I_c^L indicate whether condition c has access to D, L at all. This ledger is our extension of the BabyLM data budget (Warstadt et al., 2023) to systems that hold data outside the weights; its relation to the official word-based rule is discussed in the Limitations section. Token occurrences are charged once per accessible resource per condition, so an epoch-trained model and a repeatedly queried datastore are comparable. This makes retrieval an allocation decision rather than an extra-data decision: the same datastore tokens can be left unused, used for training, or held in a lookup table. We keep these uses mutually exclusive across our five conditions so that no condition both trains on D and retrieves from it. The ledger's own accounting would not forbid a combined condition, and we leave that hybrid to future work.

3.3 Retrieval and prompting

RETRIEVE-D's retriever uses TF-IDF over the datastore's source side, with language-aware tokenization for all languages, shared with the dictionary arm of Appendix F (kiwipiepy using the default model for Korean, jieba with its default dictionary for Chinese, and fugashi with the unidic-lite dictionary for Japanese), and prepends the top-k=3 retrieved examples to the source sentence before a translation marker and the target; the fine-tuning

loss is computed on target tokens only, so retrieval shapes conditioning without itself being a prediction target. This lexical retriever is budget-clean by construction; we additionally report a LaBSE (Feng et al., 2022) dense retriever as an off-budget upper-reference and find it indistinguishable from the lexical retriever (Section 5.3). The rationale for treating LaBSE as off-budget and the prompt template are given in Appendix C.

4 Experimental Setup

Languages and pairs. We study English (en), Chinese (zh), Japanese (ja), and Korean (ko), evaluating directed translation across all 12 directed pairs: both directions of en↔zh, en↔ja, en↔ko, zh↔ja, zh↔ko, and ja↔ko. A core set of four pairs (en↔zh, ko↔zh) receives the full 3-seed design (S-ONLY, TRAIN-ON-D, RETRIEVE-D, plus the two inference-only probes) and anchors the main results; the S-ONLY control is extended to the remaining 8 pairs to test generalization (Section 5.1); and a breadth grid crosses TRAIN-ON-D, a secondary dictionary-retrieval arm (DICT-RETRIEVE; Appendix F), and a LaBSE-retriever RETRIEVE-D over all 12 pairs for external validity (Appendix B). Chinese is normalized to Simplified throughout. Cross-pair comparisons are descriptive only, since data sources and domains differ by pair.

Data. Each directed model is charged for pretraining, fine-tuning, and datastore tokens under the 100M-token ledger of Section 3.2. Pretraining uses language-specific Wikipedia. Parallel data are pair-specific: the Korean-pivot CJK pairs (ko↔zh, ko↔ja) use AI-Hub corpora; en↔ko combines AI-Hub with WikiMatrix (Schwenk et al., 2021); en↔zh combines MultiUN (Eisele and Chen, 2010), WikiMatrix, and Tatoeba; and en↔ja and zh↔ja use WikiMatrix alone. Within each pair, fine-tuning examples and the retrieval datastore are disjoint; for the Korean-pivot CJK pairs they additionally come from different registers of the same corpus (literary books for fine-tuning, broadcast captions for the datastore), a mismatch Section 5.4 returns to. The test set is FLORES-200 devtest (NLLB Team et al., 2022) (1,012 sentences per pair), held out and not counted against the budget; a confirmatory re-evaluation additionally uses FLORES-200 dev (997 sentences per pair), likewise held out and off-budget (Section 5.2, Appendix E). Datastores are token-capped per pair so that every trained system remains within budget. Full provenance, licenses, exact source mixtures and counts, per-pair budget accounting, and the contamination audit are in Appendix A.

Model. Every condition and pair uses the same architecture: a GPT-2-style (Radford et al., 2019) decoder-only Transformer, 8 layers, 512 hidden, ≈42M parameters, trained from scratch, with a corpus-driven 32k-vocabulary BPE (Sennrich et al., 2016) tokenizer trained on the pretraining corpus. Each translation pair is fine-tuned from a pretrained base whose BPE covers the pair's scripts; all conditions for a directed pair share the same base, so any difference between conditions is attributable to how the datastore is used rather than to architecture or tokenization. Pretraining is capped at 78–84M tokens per base so that pretraining, fine-tuning, and datastore tokens jointly fit under the 100M-token ledger (Appendix A).

Training and decoding. Supervised fine-tuning budgets are nested, 1k $\subset$ 10k $\subset$ 100k pairs, with a shared held-out validation split. Training uses 3 fine-tuning seeds per core-set cell, and the breadth grid uses 3 seeds throughout; in the extended no-datastore control (Section 5.1) the TRAIN-ON-D and RETRIEVE-D arms use 3 seeds, the S-ONLY control uses 3 seeds on a subset of borderline cells and a single seed elsewhere (Appendix C), and other descriptive sweeps are single-seed where noted. All conditions, budgets, and seeds share one frozen decoding configuration (beam search, beam width 4), so decoding differences can never confound a cross-condition comparison. Because TRAIN-ON-D runs the same number of epochs over $S \cup D$, it receives more optimizer updates and cumulative token exposure than the other trained conditions. All seed variation is in supervised fine-tuning; every condition for a given pair shares one pretraining run, so cross-seed confidence intervals capture fine-tuning randomness and understate total run-to-run variance, and results are conditional on one pretrained base per pair (see Limitations). Optimizer settings, checkpoint selection, beam-search details, the seed-allocation rule for the extended S-ONLY control, and a repetition-control decision made before the evaluation grid ran are given in Appendix C.

Metrics and significance. The headline metric is plain chrF (Popović, 2015) (sacreBLEU (Post, 2018) `CHRF, word_order=0`); BLEU (Papineni et al., 2002), TER (Snover et al., 2006), COMET (Rei et al., 2020), and chrF++ (Popović, 2017) are secondary and reported in Appendix D,

pair	budget	S-only	Train-D	Retr-D	Rand-ctx	No-ctx	ΔTrain	ΔRetr
zh-en	1k	17.9	36.0	16.7	17.3	15.6	+18.1	−1.2
zh-en	10k	26.9	35.8	25.4	24.9	16.9	+8.9	−1.5
zh-en	100k	34.8	36.6	34.4	34.4	18.1	+1.9	−0.3
en-zh	1k	3.8	16.9	3.6	2.5	2.2	+13.1	−0.2
en-zh	10k	9.9	16.9	9.1	8.4	4.4	+7.0	−0.8
en-zh	100k	15.9	17.7	15.8	15.5	5.0	+1.8	−0.2
ko-zh	1k	5.0	3.7	4.6	4.3	3.3	−1.3	−0.4
ko-zh	10k	10.0	8.3	9.8	9.7	4.9	−1.7	−0.2
ko-zh	100k	14.4	13.5	14.3	14.3	5.1	−0.9	−0.2
zh-ko	1k	6.0	5.4	6.0	5.7	4.4	−0.6	0.0
zh-ko	10k	11.3	6.4	11.0	11.0	6.4	−4.9	−0.3
zh-ko	100k	15.1	14.1	14.9	14.8	4.1	−1.0	−0.2

Table 1: Core-set attribution results (3-seed means, plain chrF). S-only = S-ONLY, Train-D = TRAIN-ON-D, Retr-D = RETRIEVE-D, Rand-ctx = RANDOM-CONTEXT, No-ctx = NO-CONTEXT (Figure 2). ΔTrain = Train-D − S-only; ΔRetr = Retr-D − S-only. Deltas are computed from unrounded seed means. ko↔zh TRAIN-ON-D values use the original datastore; values after removing its markup artifact are in Appendix G. All twelve ΔTrain 3-seed Student-t 95% CIs exclude zero; six of twelve ΔRetr CIs do, all negative (Appendix D).

where the core-set results are also confirmed metric-robust. Primary contrasts are within-pair paired deltas on the same FLORES sources, aggregated across seeds with Student-t 95% confidence intervals, with false-discovery-rate control across the family of per-seed tests (Appendix D).

The dictionary-retrieval arm is confined to the appendices (Appendix F; Table 7) because coverage collapsed with budget and a tokenizer mismatch prevented a clean retrieval-quality interpretation.

5 Results

Unless otherwise stated, all scores are plain chrF on FLORES-200 devtest ($n = 1{,}012$), decoded with the frozen configuration of Section 4, and reported as 3-seed means; single-seed cells are marked explicitly.

5.1 Retrieval adds no value over the no-datastore control

Table 1 compares the no-datastore control (S-ONLY) with TRAIN-ON-D (datastore baked into weights) and RETRIEVE-D (datastore retrieved, budget-clean lexical retriever), together with the two inference-only probes.

Retrieval does not beat the no-datastore control on the core set. ΔRETRIEVE-D lies between −1.5 and 0.0 chrF in every core-set cell. This holds for the budget-clean lexical retriever, for the LaBSE upper-reference, and under every secondary metric (Appendix D). Holding the datastore in a retriever is at best neutral, and typically slightly negative, relative to not having the datastore at all.

TRAIN-ON-D gains diminish with additional supervision. On en↔zh, ΔTRAIN-ON-D is +13 to +18 chrF at 1k and shrinks monotonically to +1.8 and +1.9 at 100k, never turning negative. ko↔zh is the outlier: ΔTRAIN-ON-D is negative at every budget (−0.6 to −1.7), with one deeper, seed-unstable cell (zh-ko 10k, −4.9) partly traced to a datastore artifact in Appendix G.

The null generalizes to all 12 pairs. Extending the S-ONLY control to the remaining 8 pairs (Table 2) yields ΔRETRIEVE-D $\leq$ +0.5 chrF in all 24 new cells; 21 of 24 are $\leq$ +0.1. Across all 36 pair–budget cells, 30 deltas are non-positive and 34 are at most +0.2. In these 24 cells RETRIEVE-D is the LaBSE variant from the breadth grid, since the lexical retriever is instantiated only on the core set; the extension therefore tests the qualitative pattern rather than the budget-clean accounting. Where both retrievers were run, they differ by at most 0.1 chrF (Section 5.3). The largest positive delta (en-ja at 1k, +0.5) survives a 3-seed re-run (S-ONLY range across seeds 0.11; per-seed Δ +0.42 to +0.50) but sits at an absolute chrF of ≈5, well below usable translation quality. Nowhere does retrieval beat the no-datastore control at a budget where the systems are usable.

5.2 Context is required; retrieval relevance is not

The analysis in this section uses only the four core pairs, on which the inference-time probes were run; the eight-pair extension of Section 5.1 tests the null, not the mechanism.

pair	budget	ΔTRAIN-ON-D	ΔRETRIEVE-D
en-ja	1k/10k/100k	+12.4 / +7.6 / +2.0	+0.5 / −0.9 / −0.3
ja-en	1k/10k/100k	+15.7 / +9.2 / +2.1	−0.1 / −1.6 / −0.6
en-ko	1k/10k/100k	+10.3 / +6.4 / +1.7	+0.1 / −0.7 / −0.1
ko-en	1k/10k/100k	+16.6 / +8.9 / +2.2	+0.3 / −1.7 / −0.7
ja-ko	1k/10k/100k	+9.8 / +3.3 / +1.0	−0.4 / −0.3 / −0.3
ko-ja	1k/10k/100k	+8.0 / +2.1 / +0.4	−0.7 / −0.3 / 0.0
ja-zh	1k/10k/100k	+5.8 / +3.4 / +1.0	−0.4 / +0.1 / +0.1
zh-ja	1k/10k/100k	+6.9 / +4.9 / +0.7	−0.3 / 0.0 / +0.2

Table 2: No-datastore control extended to the remaining 8 pairs, all vs. S-ONLY (plain chrF). TRAIN-ON-D and RETRIEVE-D are 3-seed means; S-ONLY is 3-seed on the 10 borderline cells and single-seed elsewhere (Appendix C). RETRIEVE-D here is the LaBSE-retriever variant (Section 5.3). With Table 1 this covers all 12 directed pairs.

Removing context collapses performance. Stripping retrieved context from the trained RETRIEVE-D checkpoints (NO-CONTEXT, same frozen decoding) costs 16.4 chrF at zh-en 100k (34.4 → 18.1), 10.8 at en-zh 100k, 9.2 at ko-zh 100k, and 10.8 at zh-ko 100k (Table 1). The model is strongly conditioned on retrieval-shaped input; it is not ignoring the datastore.

Replacing query-matched neighbors with random same-pair examples costs almost nothing. RANDOM-CONTEXT feeds the same checkpoints count-matched random datastore pairs and recovers essentially the full RETRIEVE-D level at 10k and 100k: within 0.3 chrF on every 100k cell and within 0.7 at 10k. At 1k the gaps are larger relative to the low absolute scores but mixed in sign: −1.1 on en-zh, +0.6 on zh-en. The dependency decomposes cleanly:

$$\text{S-ONLY} \approx \text{RETRIEVE-D} \approx \text{RANDOM-CONTEXT} \gg \text{NO-CONTEXT}.$$

The model therefore depends strongly on the retrieval-trained exemplar format, while the identity of the query-matched neighbor contributes little. Importantly, RANDOM-CONTEXT draws examples from the same directed language pair: it breaks query relevance while preserving language-pair identity and parallel-example structure. We therefore interpret this control as evidence against strong dependence on nearest-neighbor relevance, not as evidence that arbitrary context would suffice. "Exemplar-shaped context" should accordingly be read as the retrieval-time format together with whatever pair-level information same-pair exemplars carry; separating the two would require cross-pair or non-parallel contexts, which we did not run (see Limitations). For a model fine-tuned into the dependency, this parallels the random-demonstration result of Min et al. (2022). The ordering also holds under BLEU, TER, and COMET (Appendix D); the headline is not a chrF artifact.

The result survives exact length matching. RANDOM-CONTEXT matches the *number* of retrieved exemplars but not their length, so we confirm the result with a length-matched control (LM-RANDOM): in a confirmatory re-evaluation on FLORES-200 dev, with the same checkpoints and frozen decoding, the three retrieved exemplars are replaced by random datastore pairs whose rendered BPE length exactly matches the retrieved ones' (Appendix E). The 100k ordering replicates on this confirmatory split: RETRIEVE-D−S-ONLY stays within $[-0.19, +0.13]$ chrF on all four core-set pairs, RANDOM-CONTEXT stays within −0.2 of RETRIEVE-D, and NO-CONTEXT again collapses (Table 8, Appendix E). At 100k, LM-RANDOM sits only 0.14 (en→zh) and 0.25 (zh→en) chrF below RETRIEVE-D, against losses of 10.9–16.9 chrF on these two pairs when context is removed: most retrieval-conditioned performance does not depend on neighbor identity even with exemplar length held fixed. On zh→ko dev, a sizable fraction of stripped-context outputs are empty strings rather than fluent-but-wrong translations (Appendix E); NO-CONTEXT reflects dependence on the retrieval-conditioned interface, not productive use of retrieved information.

5.3 Robustness: retriever quality and datastore size

Two potential confounds could explain the null of Section 5.1: a retriever that is too weak, or a datastore that is too small.

Retriever quality. Table 3 compares the budget-clean lexical retriever against the off-budget LaBSE-dense reference on five cells. Differences range from −0.07 to +0.09 chrF, within cross-seed variation. Two consequences follow: the budget-clean retriever gives up nothing measurable relative to the off-budget one, so no accounting exception for a pretrained retriever is needed; and the non-additivity of Section 5.1 is not attributable to retriever weakness in these cells. This is consistent with the null being about neighbor relevance rather than retriever strength (Section 5.2), though it does not bound an oracle or task-trained retriever.

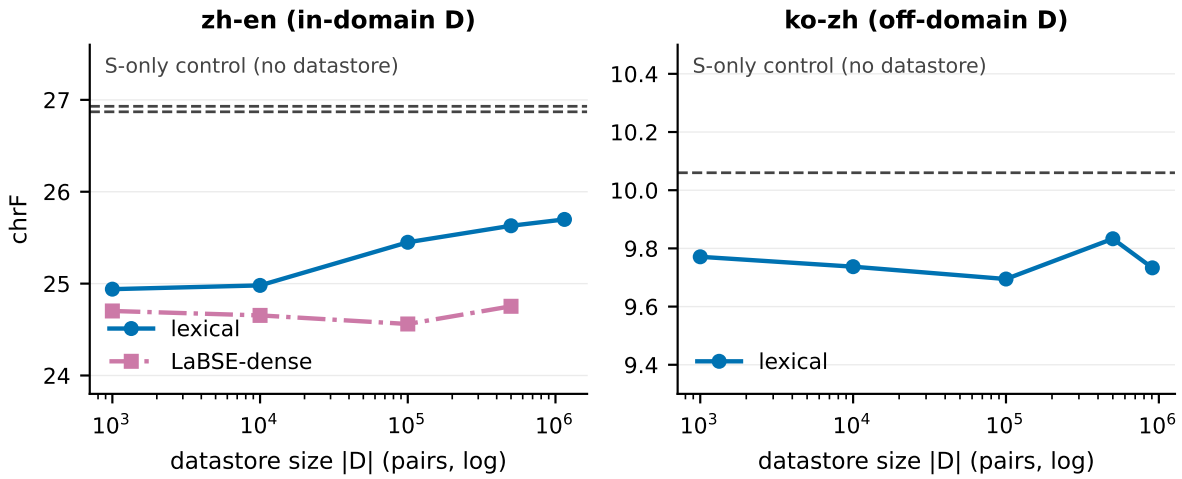


Figure 3: RETRIEVE-D chrF vs. datastore size $|D|$ (log scale, single seed per point), fixed $S = 10$k, against the S-ONLY control. Every curve stays below its control level across a ∼1,000× range of $|D|$; only the in-domain zh-en lexical sweep shows a mild, plateauing slope.

pair	budget	lexical	LaBSE	lex.−LaBSE
zh-ko	10k	10.97	11.02	−0.05
zh-ko	100k	14.90	14.98	−0.07
ko-zh	100k	14.25	14.25	0.00
zh-en	100k	34.44	34.46	−0.02
en-zh	100k	15.75	15.66	+0.09

Table 3: RETRIEVE-D with the budget-clean lexical retriever vs. the off-budget LaBSE-dense upper-reference (3-seed, plain chrF). Two decimals are reported because the contrasts are below 0.1 chrF.

Datastore size. Table 4 and Figure 3 report inference-only sweeps on fixed RETRIEVE-D checkpoints ($S = 10$k), growing the datastore from 1k pairs to the full disjoint pool (off-budget by construction). We sweep both retrievers on zh-en, whose datastore is partly Wikipedia-derived (WikiMatrix) and therefore closest in domain to FLORES, and the lexical retriever on ko-zh, whose broadcast-caption datastore is off-domain. Every curve sits below the no-datastore control at every size. The only monotone gain is the zh-en lexical sweep (+0.76 chrF over a 1,150× pool increase, plateauing); the dense retriever is flat (±0.2, non-monotone), and the ko-zh sweep is flat (±0.1 over ∼900×). Even the largest tested datastore does not reach the no-datastore control. This is unsurprising given Section 5.2: if neighbor identity barely matters, better neighbors drawn from a larger pool cannot help.

5.4 Apparent retrieval wins decompose into baseline harm

The budget-matched allocation contrast, TRAIN-ON-D vs. RETRIEVE-D, is misleading without the no-datastore control. Table 5 lists every one of the 36 pair–budget cells in which RETRIEVE-D significantly beats TRAIN-ON-D (3-seed CIs exclude zero): the six ko↔zh cells. The contrast is significantly negative in the remaining 30. All six decompose the same way: RETRIEVE-D ≈ S-ONLY (retrieval adds nothing), while TRAIN-ON-D falls below the control level (training on the datastore actively hurts). The apparent retrieval gain is therefore an avoided baseline loss, not added retrieval value: "retrieval beats weights" here means that retrieval avoids a harm that baking-in caused, and for these cells the best allocation is to spend no budget on the datastore at all.

The general form is that when a baseline's treatment can be harmful, a comparison against that baseline conflates the competitor's gain with the baseline's loss. Confidence intervals that exclude zero do not rule this out: significance validates the contrast that was run, not the attribution inferred from it.

Domain match does not explain the sign pattern. Across all 12 pairs at the 1k budget, ΔTRAIN-ON-D is positive on 10 of 12 (+5.8 to +18.1); only ko↔zh is negative. This invites a domain reading: en↔zh's WikiMatrix/MultiUN datastore is in-domain for FLORES and gains, while ko↔zh's AI-Hub captions are off-domain and lose. We spec-

sweep	$\|D\|$ 1k	10k	100k	500k	full pool	S-ONLY control
zh-en, lexical	24.94	24.98	25.45	25.63	25.70 (1.15M)	**26.93**
zh-en, LaBSE-dense	24.70	24.65	24.56	24.75	—	**26.87**
ko-zh, lexical	9.77	9.74	9.69	9.83	9.73 (907k)	**10.06**

Table 4: Datastore-size sweep on fixed RETRIEVE-D checkpoints ($S = 10$k, plain chrF). Each sweep uses a single fine-tuning seed and the S-ONLY column reports that seed's control (seed 1 for the two lexical sweeps, seed 2 for the LaBSE sweep), which is why the two zh-en rows show different control values. The LaBSE full-pool point was not run (—).

cell	Retr−Train	95% CI	Retr−S	Train−S	reading
ko-zh 1k	+0.91	[0.28, 1.53]	−0.4	−1.3	baseline harm
ko-zh 10k	+1.49	[1.13, 1.86]	−0.2	−1.7	baseline harm
ko-zh 100k	+0.73	[0.39, 1.06]	−0.2	−0.9	baseline harm
zh-ko 1k	+0.60	[0.18, 1.02]	0.0	−0.6	baseline harm
zh-ko 10k	+4.60	[3.76, 5.45]	−0.3	−4.9	baseline harm
zh-ko 100k	+0.81	[0.44, 1.18]	−0.2	−1.0	baseline harm

Table 5: Decomposing every positive Retrieve-D vs. Train-D delta among all 12 pairs (paired 3-seed Student-t 95% CIs; all six CIs exclude zero). The six positive cells are core-set cells with the lexical retriever; the contrast is significantly negative in the remaining 30 cells, which include the LaBSE-retriever extension (Table 2). Retr = RETRIEVE-D, Train = TRAIN-ON-D, S = S-ONLY.

ified domain predictions for the four undirected extension pairs before examining their results (positive for en↔ja and zh↔ja, negative for ja↔ko, borderline for en↔ko; Appendix B), and the prediction fails: ja↔ko, trained against the same off-domain caption store as ko↔zh, gains +9.8/+8.0 chrF at 1k, the opposite of the predicted sign. The full grid instead supports a budget-gated boost from extra bitext of nearly any register—large when data-starved, fading to +0.4 to +2.2 by 100k without flipping sign—with ko↔zh a pair-level outlier (Appendix G), not evidence of a domain law. Full predictions and outcomes are given in Appendix B.

Caveat. All six ko↔zh cells in Table 5 use RETRIEVE-D checkpoints trained and evaluated with artifact-bearing datastore context; only TRAIN-ON-D was rerun after cleaning (Appendix G), which isolates the weights-memory pathway but not the conditioning pathway. The risk concentrates in zh-ko 10k, where cleaning shifted TRAIN-ON-D by +2.2 chrF (3-seed mean) and altered seed-level stability; the other cells shift by at most 0.3 chrF. The decomposition is therefore diagnostic rather than fully clean causal evidence for ko↔zh, but the central RETRIEVE-D-vs.-S-ONLY result of Section 5.1 is unaffected.

6 Discussion

Dependency without value. The substitution thesis (Borgeaud et al., 2022; Khandelwal et al., 2021; Shao et al., 2024) predicts that datastore tokens can do the work of tokens trained into weights. Under our shared-ledger accounting, the prediction fails in a specific way: retrieval provides no consistent additive gain across the 12 pairs (Section 5.1), yet on the four core pairs where we probed it, it is not ignored either (Section 5.2). Retrieval training creates an input dependency whose system-level payoff is zero: the model learns to need the interface and, with it, recovers only the level of a model trained never to need it. "Small models ignore retrieval" is the wrong summary; retrieval induces dependency without value.

Relation to BabyLM. We make no cognitive claims. Our question is the BabyLM question posed at the systems level: given a corpus on the scale the challenge treats as developmentally plausible, is that corpus better spent internalized in the weights or kept as consultable memory? For translation with ≈42M parameters, the answer is that consultable memory is not used as memory: the model internalizes the retrieval interface but shows little dependence on the query-specific content of the retrieved neighbors, and the data would have been more useful in the weights whenever it was useful at all. Two further points connect to the workshop's aims: the ledger extends BabyLM's data accounting to systems that hold data outside the weights, and the English–Chinese–Japanese–Korean setting exercises the budget on scripts and language pairs beyond English.

Consequences for evaluation design. A retrieval-MT comparison that reports a retrieval-augmented system against a datastore-trained baseline, without a no-datastore control, cannot distinguish a genuine retrieval gain from an avoided baseline loss (Section 5.4). The risk

is largest exactly where noisy or mismatched datastores and small fine-tuning sets are common. We therefore recommend reporting a no-datastore control alongside any retrieval system evaluated against a datastore-trained baseline.

7 Conclusion

We asked how a small machine translation model should spend a fixed corpus budget when the same parallel data can either be trained into its weights or retrieved into its prompt. Under a shared 100M-token ledger that charges pretraining, fine-tuning, and datastore tokens alike, we compared leaving a pair-specific datastore unused, training on it, and retrieving from it, across 12 directed pairs among English, Chinese, Japanese, and Korean, with ≈42M-parameter models trained from scratch.

Three findings follow. First, prompt-level retrieval never exceeds the no-datastore control by more than +0.5 chrF across 36 pair–budget cells and is non-positive in 30; the null persists under an off-budget dense retriever and retrieval pools up to ∼1,000× larger. Second, training on the same datastore is the better use of the budget when supervision is scarce (+5.8 to +18.1 chrF at 1k pairs on 10 of 12 pairs), with gains fading to +0.4–2.2 chrF by 100k. Third, on the four-pair core set, retrieval-trained models are nonetheless strongly dependent on their prompt interface: removing context costs up to 16.4 chrF, while query-unrelated same-pair examples—count-matched on all four pairs and exactly length-matched on en↔zh—recover nearly the full score. Retrieval induces dependence without value.

These findings carry a diagnostic consequence: every cell in which retrieval "beats" training on the datastore is one where training fell below the no-datastore control, so the apparent win is an avoided baseline loss rather than added retrieval value. Our claims are limited to prompt-level retrieval in small models trained from scratch and to same-pair relevance-breaking controls. They motivate two reporting practices for retrieval evaluation under constrained corpus access: a no-datastore control that separates retrieval gains from baseline harm, and a relevance-breaking control that separates use of retrieved information from dependence on the retrieval-conditioned interface.

Limitations

Mechanism. Our negative result is scoped to prompt-level (in-context) injection. It does not cover kNN-style distribution interpolation (Khandelwal et al., 2021), whose stronger coupling might extract value where prompting does not.

Controls. RANDOM-CONTEXT and LM-RANDOM draw replacement exemplars from the same directed pair, so they break query relevance while preserving language-pair identity and parallel structure. They rule out dependence on query-specific neighbor information, not on same-pair exemplars as such; contexts drawn from another language pair, from non-parallel text, or with shuffled source–target alignment would be needed to separate format from pair-level content, and we did not run them. LM-RANDOM was run on en↔zh at the 100k budget only.

Model, task, and data. All experiments use ≈42M-parameter decoder-only Transformers trained from scratch, evaluated on machine translation prompting with these specific datastores. Larger or differently pretrained models could behave differently.

Seeds. Each pair uses one pretraining seed; all seed variation is in fine-tuning. Cross-seed intervals therefore capture fine-tuning randomness and understate total run-to-run variance.

Accounting. Results are conditional on our corpus-access ledger (Section 3.2), which charges BPE-token occurrences once per accessible resource. It departs from the official BabyLM rule in two ways: we count tokens rather than words, because word boundaries are ill defined for Chinese and Japanese; and we charge fine-tuning and datastore tokens, not only pretraining tokens. Different accounting choices would shift the allocation contrasts.

Excluded arms and open outliers. A secondary dictionary-retrieval arm is disclosed but excluded from retrieval-quality and mechanism conclusions: its coverage collapses with fine-tuning budget, and a tokenizer mismatch confounds the CJK-source evaluations (Appendix F). The ko↔zh pair's residual gap from its control level remains an unexplained pair-level outlier (Appendix G). It does not bear on the central RETRIEVE-D-VS.-S-ONLY result, but the baseline-harm decomposition of Sec-

tion 5.4 is drawn entirely from this pair and is qualified accordingly.

Acknowledgments

This research used datasets from 'The Open AI Dataset Project (AI-Hub, S. Korea)'; all data information can be accessed through AI-Hub (https://www.aihub.or.kr).

References

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, pages 2206–2240. PMLR.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7307–7318, Online. Association for Computational Linguistics.

Xin Cheng, Shen Gao, Lemao Liu, Dongyan Zhao, and Rui Yan. 2022. Neural machine translation with contrastive translation memories. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3591–3601, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Andreas Eisele and Yu Chen. 2010. MultiUN: A multilingual corpus from united nation documents. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 2868–2872.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Marjan Ghazvininejad, Hila Gonen, and Luke Zettlemoyer. 2023. Dictionary-based phrase-level prompting of large language models for machine translation. *arXiv preprint arXiv:2302.07856*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*. PMLR.

Hongkun Hao, Guoping Huang, Lemao Liu, Zhirui Zhang, Shuming Shi, and Rui Wang. 2023. Rethinking translation memory augmented neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2589–2605.

Qiuxiang He, Guoping Huang, Qu Cui, Li Li, and Lemao Liu. 2021. Fast and accurate neural machine translation with translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3170–3180, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, and Al Youngblood. 2022. No language left

behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rulin Shao, Jacqueline He, Akari Asai, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, and Pang Wei Koh. 2024. Scaling retrieval-based language models with a trillion-token datastore. *arXiv preprint arXiv:2407.12854*.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA. Association for Machine Translation in the Americas.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

A Data Sources and Budget Accounting

Pretraining data source. Wikipedia (`wikimedia/wikipedia`, 20231101 snapshot, CC-BY-SA-4.0), restricted to the pretraining base's languages and capped to the per-language budget.

Parallel data sources. Each pair has one deduplicated parallel pool, shuffled with a fixed seed and aligned across the two directions, from which the nested fine-tuning splits and the datastore are carved disjointly. The per-pair composition is: ko↔zh and ko↔ja use AI-Hub Korean-pivot MT corpora (≈1.01M and ≈1.04M pairs respectively), with the literary-book subcorpus supplying the fine-tuning splits and the broadcast-caption subcorpus supplying the datastore pool; en↔ko merges the AI-Hub book subcorpus (150,543 pairs) with WikiMatrix (Schwenk et al., 2021) (112,124) into a pool of 262,667; en↔zh merges MultiUN (Eisele and Chen, 2010) (capped at 500,000), WikiMatrix (681,382), and Tatoeba (74,606) into a pool of 1,255,636 after cross-source deduplication; en↔ja uses WikiMatrix alone (545,302); and zh↔ja uses WikiMatrix alone (256,579), as constructing zh–ja pairs by pivoting through the Korean- or English-sided corpora proved infeasible (the candidate corpora share almost no pivot-side sentences). WikiMatrix pools use the LASER margin ≥ 1.04 release; the Chinese side of the web-sourced pools is converted to Simplified with OpenCC before splitting.

Dictionary data sources. The dictionary resources are krdict (ko↔zh: 108,686 entries; ko↔ja: 168,532), CC-CEDICT (en–zh: 152,467), JMdict (en–ja, capped at 200k), and psyche (the psyche/dictionary_koen Hugging Face dataset, https://huggingface.co/datasets/psyche/dictionary_koen, CC-BY-SA-4.0; derived from the National Institute of Korean Language's Korean Learners' Dictionary and Standard Korean Language Dictionary) (en–ko: 125,906). Reverse directions are constructed by swapping the source and target columns. For zh↔ja, we construct a directional Wiktionary lexicon from Kaikki/Wiktextract using English sense-aligned translation tables and the Chinese-language section of Japanese Wiktionary (Han-only, OpenCC-normalized; 46,999 entries). Coverage is adequate for zh→ja but approximately 0.7% for ja→zh. The dictionary analysis therefore covers 10 of the 12 directed pairs; zh↔ja is evaluated only with TRAIN-ON-D and RETRIEVE-D in both directions: the constructed lexicon is usable in one direction only, so we treat zh↔ja as a pair without a usable open-license dictionary and drop the dictionary family (DICT-RETRIEVE, RANDOM-DICT, ORACLE-DICT) for both directions, including zh→ja despite its adequate coverage; both directions are marked — in the dictionary columns of Table 7.

Evaluation data. FLORES-200 devtest (NLLB Team et al., 2022) contains 1,012 sentences per pair; directed pairs are constructed from the shared multilingual sentence IDs. FLORES is held out and excluded from the data budget. Exact deduplication and a LaBSE-based semantic audit found no exact overlaps or flagged near-duplicates between FLORES and the training resources.

Budget construction. Each pair uses nested fine-tuning subsets of 1k, 10k, and 100k examples, a 1k validation split, and a pair-specific datastore D. For each directed model, the budget is

$$P + S + D + \text{dict} \leq 100\text{M}$$

when measured with that model's 32k BPE over both sides of the deduplicated parallel and dictionary records. FLORES is excluded. Pretraining is capped at approximately 78–84M tokens, depending on the pair. The datastore is capped by token count rather than pair count. An initial 200k-pair cap produced totals of 101.8M tokens for en↔zh and 100.7M for en↔ja because these corpora contain longer sentences. The final data construction sets $|D|$ to $100\text{M} - P - S - \text{dict}$ using a prefix of the shuffled pool shared by both directions of the pair. This yields 10.0M datastore tokens (166,794 pairs; 99.83M total) for en↔zh and 9.5M datastore tokens (182,921 pairs; 99.84M total) for en↔ja; all other pairs were already within budget. This correction affects only TRAIN-ON-D and RETRIEVE-D, which access D; the dictionary conditions are independent of D.

Pretrained base selection. Each directed pair is fine-tuned from a pretrained base whose tokenizer covers both scripts (e.g., the ko–zh base for {ko, zh}). Under the fixed pretraining budget, a bilingual base receives approximately 42M tokens per language; trilingual and four-language bases (approximately 28M and 21M tokens per language) were also pretrained but are not used in any reported result; all reported results use the six bilingual bases. Held-out pretraining loss decreased from 7.47 to 3.99 for the ko–zh base and from 7.33 to 3.81 for the ja–ko base.

B Full Experimental Grid and Additional Results

Table 6 lists the full experimental grid, including the dictionary family and its inference-only controls, with internal run labels and roles.

Domain-match analysis. We evaluated whether the sign of ΔTRAIN-ON-D is explained by domain match between the datastore and FLORES. Before examining the eight-pair extension, we assigned positive effects to the in-domain en↔ja and zh↔ja pairs, negative effects to the off-domain ja↔ko pair, and borderline effects to en↔ko. Table 2 does not support this prediction: ja↔ko, which uses the same off-domain caption datastore as ko↔zh, improves by +9.8/+8.0 chrF at the 1k budget. Across the full grid, TRAIN-ON-D generally provides a large low-budget gain that decreases with additional fine-tuning data. ko↔zh remains the only pair family with consistently negative deltas and is analyzed separately in Appendix G.

Additional 100k-budget results. The breadth grid evaluates {TRAIN-ON-D, DICT-RETRIEVE, RETRIEVE-D-LaBSE} across 12 directed pairs (10 for the dictionary conditions), three fine-tuning budgets, and three seeds; RANDOM-DICT and ORACLE-DICT are added at inference. Because this grid does not contain S-ONLY, it is descriptive with respect to retrieval additivity. Table 7 reports the 100k-budget cells.

Name	Label	Prepended context	Retriever	Role	Train?
S-only	C_S	— none; no datastore at all (fine-tuned on S only)	—	no-datastore control	yes
Train-on-D	C0	— none; datastore baked into weights (fine-tuned on $S \cup D$)	—	weights memory	yes
Retrieve-D	C2	nearest parallel-sentence example(s)	lexical TF-IDF over D; LaBSE-dense comparison	external phrasal memory	yes
No-context	C2−ret	— context stripped from the trained Retrieve-D model	—	dependence probe	inference
Random-context	C2-rand	count-matched random datastore pairs	random, relevance-broken	neighbor-relevance control	inference
Dict-retrieve	C1	dictionary entries (src → tgt) for rare source words	exact lookup + frequency gate	lexical memory (secondary)	yes
Random-dict	C3	random dictionary entries (no source-word overlap)	random, count-matched to Dict-retrieve	prompt-length control	inference
Oracle-dict	C4	dictionary entries whose target appears in the reference	oracle (eval-only)	upper bound for Dict-retrieve	inference

Table 6: Full condition grid, internal labels, and roles. RANDOM-DICT and ORACLE-DICT reuse the DICT-RETRIEVE checkpoint and modify only the inference-time retriever. Reference-dependent oracle retrieval is isolated from all training code (Appendix I).

pair	TRAIN-ON-D	DICT-RETRIEVE	RETRIEVE-D (LaBSE)	RANDOM-DICT	ORACLE-DICT
zh-en	36.6	33.1	34.5	32.1	34.3
ja-en	34.4	31.0	31.7	30.3	32.0
ko-en	35.6	32.6	32.7	31.9	33.2
en-zh	17.7	15.8	15.7	15.8	15.8
en-ja	18.4	16.4	16.1	16.4	15.3
en-ko	16.3	14.5	14.5	14.5	13.7
ko-zh	13.5	13.2	14.3	11.8	13.8
zh-ko	14.1	13.4	15.0	11.5	14.4
ko-ja	23.3	21.6	22.9	19.1	22.3
ja-ko	22.3	14.6	21.0	12.3	18.3
zh-ja	15.5	—	15.0	—	—
ja-zh	13.3	—	12.5	—	—

Table 7: Breadth grid at the 100k budget (3-seed means, plain chrF). The RETRIEVE-D−TRAIN-ON-D confidence interval excludes zero for every pair; the contrast is negative on 10 pairs and positive only on ko↔zh.

C Training, Decoding, and Prompt Construction

Training. Models are pretrained from scratch in bf16 for a fixed number of steps (10 epochs over the packed pretraining corpus at an effective batch of 64 sequences of 512 tokens, i.e. 23,960–25,690 steps depending on the base). Fine-tuning uses 10 epochs, batch size 32 (16 with two gradient-accumulation steps), bf16, dynamic padding, and target-only loss. TRAIN-ON-D is optimized over $S \cup D$, whereas S-ONLY and RETRIEVE-D are optimized over S; consequently, TRAIN-ON-D receives more optimizer updates and cumulative token exposure. Optimization uses AdamW (Loshchilov and Hutter, 2019) ($\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.01), learning rate 10^{-4}, cosine decay, 6% warmup, and gradient clipping at 1.0. Checkpoints are evaluated and saved every 200 steps, and the checkpoint with the lowest validation loss is selected. The learning rate and epoch count were chosen from a small sweep on one TRAIN-ON-D cell and then fixed for all experiments. A 100k-budget S-ONLY fine-tuning run required approximately 56 minutes on an NVIDIA A6000; wall-clock time varies with the training conditions and objectives.

Seed allocation for the extended S-ONLY control. In the 8-pair extension (Appendix B), TRAIN-ON-D and RETRIEVE-D use 3 seeds throughout; the S-ONLY control itself uses 3 seeds on the 10 borderline cells, defined as every extension cell whose initial single-seed ΔRETRIEVE-D was ≥ -0.1 chrF, and a single seed on the remaining 14 cells, where the deltas are far from zero.

Decoding. All systems use beam search with beam width 4, early stopping, length penalty 1.0, no sampling, repetition penalty 1.0, `no_repeat_ngram_size=3`, and at most 256 new

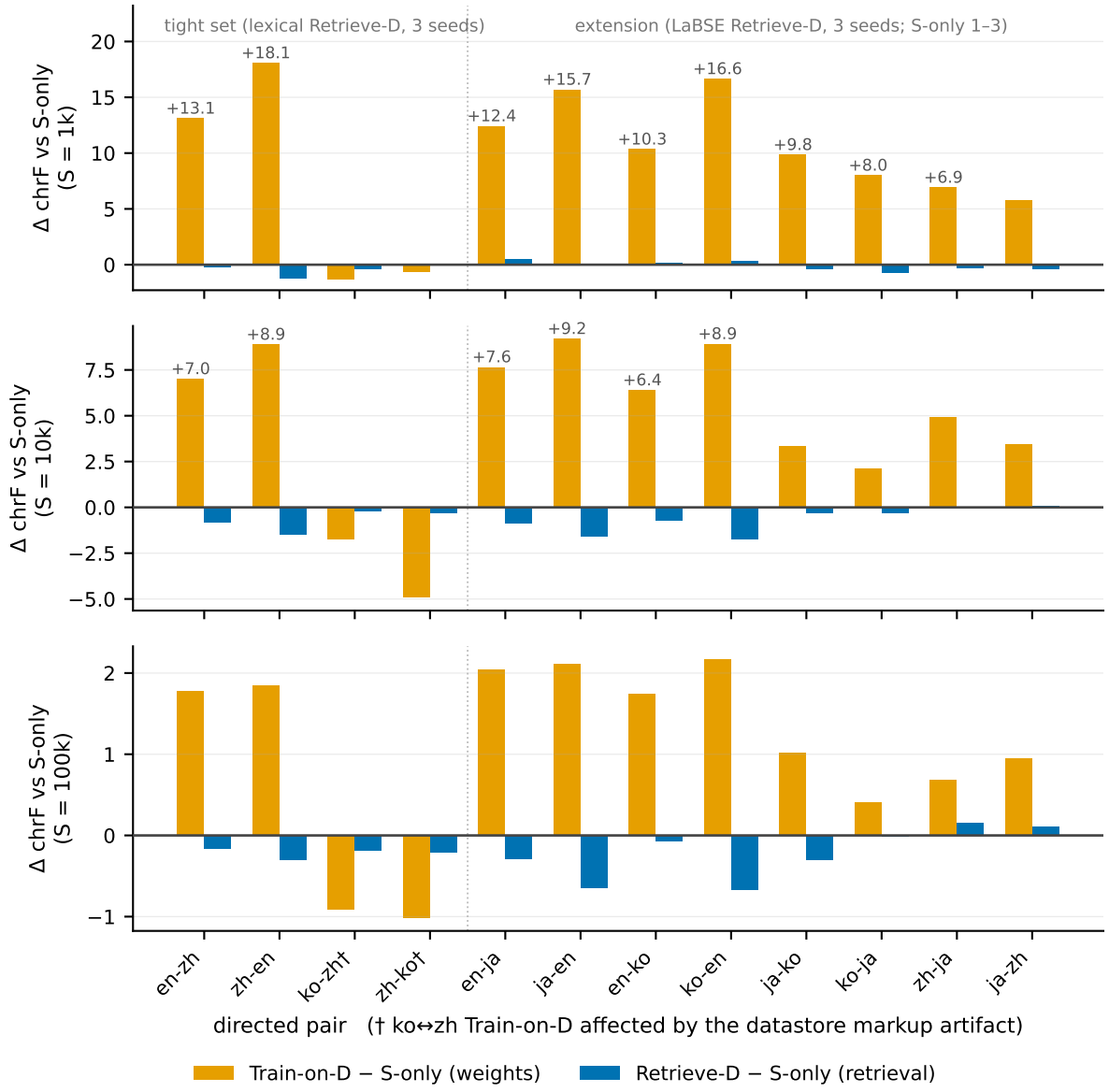


Figure 4: Datastore effects under parametric training and prompt-level retrieval across all 12 directed pairs. Each panel shows TRAIN-ON-D−S-ONLY and RETRIEVE-D−S-ONLY at one fine-tuning budget. The left side uses the lexical retriever on the four-pair core set; the right side uses the LaBSE variant from the eight-pair extension (Table 2). The figure combines the deltas in Tables 1 and 2.

tokens. The no-repeat constraint was selected in a pilot on ko→zh TRAIN-ON-D at the 100k budget after observing degenerate loops. It reduced looped outputs from 27/1,012 to 0 and outputs reaching the 256-token limit from 20 to 0; chrF increased from 13.72 to 14.47 and BLEU from 14.07 to 14.53. The setting was then fixed for all reported evaluations. Because the pilot used FLORES devtest, Section 5.2 additionally reports confirmatory results on FLORES dev.

Prompt construction and retrieval depth. The primary TF-IDF retriever uses no externally pretrained parameters and therefore satisfies the corpus-access accounting. LaBSE is evaluated separately as an externally pretrained dense comparison. Each training instance is rendered as retrieved context, followed by the source sentence, a translation marker, and the target; loss is computed only on target tokens. Sentence-retrieval prompts contain the top $k = 3$ datastore examples. Concretely, each instance is rendered as

```
<|translate|> [SRC] → [TGT]
Examples:
1. [<src1>] → [<tgt1>]
2. [<src2>] → [<tgt2>]
3. [<src3>] → [<tgt3>]
<|src|> <source>
```

```
<|tgt|> <target>
```

where SRC and TGT are upper-cased language codes, the exemplars are numbered `1.`, `2.`, `3.` and rendered as bracketed `[source]` → `[target]` lines with the arrow U+2192, and `<|tgt|>` is the translation marker; the order is exemplars, source, marker, target, with the target following the marker on the same line. RANDOM-CONTEXT replaces them with three count-matched random examples under the same template. Context construction is deterministic for a fixed seed, and random controls use per-sentence deterministic sub-seeds. Figure 1 displays only the highest-ranked example for readability, whereas all reported scores use the complete $k = 3$ prompt.

D Metrics and Statistical Analysis

Metrics. The headline metric is plain chrF (sacreBLEU `CHRF`, `word_order=0`), with BLEU, TER, COMET (`Unbabel/wmt22-comet-da`), and chrF++ (`word_order=2`) as secondary metrics, using language-aware tokenization for BLEU (sacreBLEU `tokenize=zh` for Chinese targets; character-level, `tokenize=char`, for Japanese and Korean). All metrics are computed from saved per-sentence predictions in a single scoring pass per metric across every system.

Metric consistency audit. An initial grid collation labeled chrF++ values as chrF for one subset of systems, whereas the attribution systems used plain chrF. Because the metrics differ by 2–4 points on English and Chinese targets, all systems were rescored from saved per-sentence predictions using a single implementation. Recomputed chrF++ reproduced the original values for all 486 evaluated cells, confirming that the discrepancy was a labeling error rather than a prediction mismatch. All values reported in the paper use the rescored metrics.

Metric robustness on the core set. Rescoring the four core conditions with BLEU, TER, and COMET preserves the qualitative ordering in Section 5.2. At the 100k budget, RETRIEVE-D−S-ONLY is at most +0.02 BLEU and +0.007 COMET for every pair, NO-CONTEXT is 0.19–0.28 COMET below RETRIEVE-D, and RANDOM-CONTEXT is within 0.016 COMET of RETRIEVE-D.

Significance. Primary contrasts are within-pair paired deltas on the same FLORES sources: the attribution deltas RETRIEVE-D−S-ONLY and TRAIN-ON-D−S-ONLY, the allocation delta RETRIEVE-D−TRAIN-ON-D, and the dependence delta RETRIEVE-D−NO-CONTEXT. For each contrast we aggregate per-seed deltas across the 3 seeds with a Student-t 95% confidence interval (significant iff the interval excludes zero) and control the false discovery rate across the family of per-seed tests (Benjamini and Hochberg, 1995); the FDR-adjusted results agree in sign and significance with the Student-t intervals reported in the tables (no test changes status under correction). Per-run paired bootstrap intervals were also computed but not preserved, so only the Student-t and FDR results are reported. Cross-pair comparisons are descriptive only, since data sources and domains differ across pairs.

E Confirmatory Evaluation on FLORES-200 Dev

Motivation for length matching. RANDOM-CONTEXT matches the number of exemplars but not their rendered length. At the 100k budget, mean prompt length under RETRIEVE-D versus RANDOM-CONTEXT is 158 versus 249 tokens for en–zh and 183 versus 251 for zh–en. LM-RANDOM therefore matches the three exemplar lengths exactly.

Run provenance. The confirmatory runs are inference-only evaluations of preserved checkpoints using the decoding configuration in Appendix C. Each run records hashes for the checkpoint, tokenizer, datastore, and evaluation data. On audited zh–en 100k cells, the S-ONLY, RETRIEVE-D, RANDOM-CONTEXT, and NO-CONTEXT paths reproduced all 1,012 saved devtest hypotheses byte-for-byte. TRAIN-ON-D was reproduced within 0.06 chrF at corpus level; its saved hypotheses were generated with a different batch size and are therefore not byte-comparable.

Confirmatory split. FLORES-200 dev was obtained from the official NLLB release. As a data-integrity check, the corresponding devtest files reproduce the original evaluation files exactly after trailing-whitespace normalization. A repository audit found no FLORES-dev download, cache, configuration, code path, or notebook reference predating this analysis; we therefore treat dev as an unused confirmatory split.

LM-RANDOM construction. For each dev source, ordinary lexical retrieval produces three exemplars, each rendered into the prompt as a numbered `[source]` → `[target]` line; the rendered

pair	S-only	Train-D	Retr-D	Rand-ctx	LM-Rand	No-ctx	ΔTrain	ΔRetr
zh-en	35.04	36.78	35.17	35.04	34.92	18.30	+1.75	+0.13
en-zh	15.97	17.82	15.83	15.64	15.69	4.96	+1.93	−0.14
ko-zh	14.21	13.04	14.02	13.95	—	4.64	−1.17	−0.19
zh-ko	14.79	14.10	14.90	14.74	—	3.78	−0.67	+0.11

Table 8: Confirmatory re-evaluation on FLORES-200 dev ($n = 997$, $S = 100$k, plain chrF; conditions as in Figure 2, LM-Rand = LM-RANDOM). S-ONLY, RETRIEVE-D, RANDOM-CONTEXT, NO-CONTEXT are 3-seed means; LM-RANDOM additionally averages five random draws per seed. TRAIN-ON-D is available for seeds 2–3 only (its checkpoint was inadvertently deleted before the re-evaluation; its saved devtest predictions survive and enter Table 1) and is reported descriptively. ΔTrain is the mean paired delta over those two seeds, so it does not equal the difference between the displayed 3-seed S-ONLY mean and the 2-seed TRAIN-ON-D mean; both paired deltas have the same sign in every pair (e.g., en-zh $+1.95/+1.91$), but we do not treat the two-seed interval as a stable uncertainty estimate. All ko↔zh retrieval-conditioned cells inherit the datastore markup artifact of Appendix G (the checkpoints were not retrained) and are not clean causal comparisons, though the observed ordering is stable across splits.

lines are tokenized with the checkpoint's own BPE to give a length vector $\ell = (\ell_1, \ell_2, \ell_3)$. Replacements are drawn from the same per-pair datastore to minimize $\|\ell - \ell'\|_1$, preferring exact per-position matches, and excluding the true retrieved examples, any normalized self-match of the source, and duplicates within a prompt, the same exclusion rules as RANDOM-CONTEXT. With ≈167k candidate pairs per datastore, an exact length-vector match was found for 100% of the 29,910 prompts (median and maximum L_1 distance 0), and the realized mean prompt length is identical to RETRIEVE-D's. Five draws per sentence and checkpoint seed use deterministic hash-derived seeds; draw-level variability is small (per-seed draw standard deviation ≤ 0.11 chrF) and is reported separately from seed variability. One caveat: the draw seed includes the checkpoint seed, so different fine-tuning seeds see different random exemplars, and the across-seed interval for LM-RANDOM−RETRIEVE-D includes residual context-sampling variation in addition to fine-tuning-seed variation.

NO-CONTEXT empty outputs. On zh→ko dev, 238–346 of 997 stripped-context outputs are empty strings (across the three seeds); the other pairs produce none. Empty hypotheses are preserved in the saved predictions and included in all metric computations. Removing context creates an out-of-distribution prompt mismatch, so the NO-CONTEXT score collapse partly reflects generation failure; we read NO-CONTEXT as demonstrating dependence on the retrieval-conditioned interface, not productive use of retrieved information (consistent with the list-marker degeneration on zh→en documented in Appendix H).

F Dictionary Retrieval Analysis

Design. DICT-RETRIEVE tokenizes the source sentence (language-aware: kiwipiepy for Korean, jieba for Chinese, fugashi for Japanese), looks up exact lowercased dictionary matches, keeps only rare words, and caps at a fixed number of entries. "Rare" is a rank-based bottom-quartile gate: within the active fine-tuning budget split, unique source words are ranked by frequency and the rarest 25% are eligible. This same rare-word set drives both the retrieval gate and the rare-word-accuracy metric used in evaluation, so the two are consistent by construction.

Coverage by fine-tuning budget. Eligibility is defined relative to each fine-tuning subset and also requires dictionary coverage. Under training-time augmentation of the fine-tuning split, the fraction of sentences receiving at least one entry falls from approximately 30–60% at 1k to below 3% at 100k across the ten directions with a usable lexicon; the evaluation-time densities reported below for English-source directions (2.4 entries per sentence at 1k) measure the same gate on FLORES. The lowest-frequency quartile of the larger vocabularies contains many hapax forms, proper nouns, and typographical variants not covered by the fixed lexicons. Coverage is already minimal at 1k for ja→en (0.022 entries per sentence) and ja→zh (0.007); the latter is excluded from the dictionary family, together with zh→ja, because the family is dropped for the zh↔ja pair as a whole (Appendix A).

Evaluation-time tokenizer mismatch. Training-time augmentation used the intended language-specific tokenizers. During evaluation, however, the frequency map was reconstructed

with an English tokenizer. Its keys do not match Korean, Chinese, or Japanese lemmas, which disables the rarity gate for the seven CJK-source directions. These directions therefore receive approximately five entries per sentence on nearly all examples at every budget. English-source directions follow the intended density schedule (2.4, 0.6, and 0.08 entries per sentence at 1k, 10k, and 100k). ORACLE-DICT is gated by the reference and remains approximately budget-independent; RANDOM-DICT matches the realized DICT-RETRIEVE count in each cell, so their direct contrast remains paired.

Interpretable subset. The en→CJK directions are the only dictionary conditions for which the training and evaluation tokenizers agree. At the 1k budget, where injection is non-trivial, all three pairs satisfy ORACLE-DICT $\geq$ DICT-RETRIEVE $\geq$ RANDOM-DICT (e.g., 4.20, 4.12, and 3.58 chrF for en–zh). At 10k and 100k, the intended gate injects too few entries for a reliable retrieval-quality comparison. The CJK-source directions instead contain a train–evaluation context-density mismatch of up to approximately 50× at 100k. This mismatch is associated with template copying in ja→ko at 100k (53.7% of outputs) and with inversions between ORACLE-DICT and DICT-RETRIEVE in some cells. We therefore exclude all affected CJK-source dictionary conditions from retrieval-quality and mechanism claims in the main text.

G ko↔zh Datastore Artifact Analysis

ko↔zh is the only pair family for which TRAIN-ON-D is below S-ONLY at every budget. We identified a leading-> markup artifact in approximately 50% of ko↔zh datastore target sentences; the artifact is absent from the fine-tuning subsets and FLORES. It enters TRAIN-ON-D as a training target and enters RETRIEVE-D and RANDOM-CONTEXT only as conditioning context. To isolate its effect on parametric training, we retrained TRAIN-ON-D after removing the marker from D. S-ONLY does not access D, and the retrieval-conditioned checkpoints were not retrained. Table 9 reports the resulting TRAIN-ON-D−S-ONLY deltas.

At zh→ko 10k, cleaning increases the mean delta by 2.2 chrF: two seeds recover to within 1.6 chrF of S-ONLY, while one remains in the low-performing regime. In the other five cells, cleaning changes the delta by at most 0.3 chrF. Residual deficits of approximately 0.4–1.7 chrF remain in the other five cells; zh→ko at 10k remains 2.7 chrF below S-ONLY because one cleaned seed does not recover. Because the retrieval-conditioned checkpoints were not retrained, their ko↔zh results remain artifact-bearing and are interpreted as diagnostic rather than fully controlled comparisons.

pair	budget	dirty	cleaned	moved
ko-zh	1k	−1.3	−1.6	≈0
ko-zh	10k	−1.7	−1.7	≈0
ko-zh	100k	−0.9	−1.0	≈0
zh-ko	1k	−0.6	−0.4	≈0
zh-ko	100k	−1.0	−1.1	≈0
zh-ko	**10k**	**−4.9**	**−2.7**	**+2.2**

Table 9: Δ = TRAIN-ON-D − S-ONLY before and after removing the leading-> marker from the ko↔zh datastore (plain chrF). The zh→ko 10k row is placed last for emphasis. The cleaned 10k cells are 3-seed means; the cleaned 1k and 100k cells are single-seed. At zh→ko 10k, all three original runs have chrF 6.2–6.7; two of three cleaned runs recover to the range of neighboring cells (their mean Δ is −1.6), while the third does not, giving the 3-seed mean of −2.7. Deltas and movements are computed from unrounded means.

H Qualitative Example Selection

Reported values. The score strip in Figure 1 reports three-seed mean corpus chrF for zh→en at S = 100k: 34.8 for S-ONLY, 34.4 for lexical RETRIEVE-D, 34.4 for RANDOM-CONTEXT, and 18.1 for NO-CONTEXT; TRAIN-ON-D is omitted from the figure and reported in Table 1. The displayed sentence outputs are from seed 1. All figure values use the lexical retriever; the ko↔zh allocation contrasts are reported in Table 5.

Selection procedure. For each of the 1,012 zh→en devtest items, we compute sentence-level plain chrF averaged over seeds for S-ONLY, lexical RETRIEVE-D, RANDOM-CONTEXT, and NO-CONTEXT. An item is eligible when (i) $|\text{RETRIEVE-D} - \text{RANDOM-CONTEXT}| \leq 10$, (ii) $\min(\text{RETRIEVE-D}, \text{RANDOM-CONTEXT}) - \text{NO-CONTEXT} \geq 20$, (iii) $|\text{RETRIEVE-D} - \text{S-ONLY}| \leq 10$, and (iv) the reference contains 8–30 words. These criteria yield 200 items with a median context-removal gap of 26.7 chrF. Figure 1 uses the item nearest this median (index 677; gap 26.9). The figure shows only the highest-ranked of the three retrieved examples.

No-context output pattern. Among the 1,012 zh→en NO-CONTEXT outputs, 232 (23%) contain repeated list markers such as "(iii) (ii) (iv)…". Most remaining outputs are fluent but unrelated

to the source, as in the displayed example.

I Validation and Reproducibility Checks

Control checks. For the dictionary analysis, we verify at the 1k budget that oracle retrieval is at least as accurate as non-oracle dictionary retrieval on the rare-word metric and that random dictionary context does not reproduce the same gain. These checks are interpreted only for the English-source directions because the CJK-source conditions contain the tokenizer mismatch described in Appendix F.

Budget validation. Before training, $P + S + D + \mathrm{dict}$ is computed with the directed model's tokenizer over both sides of the deduplicated records. The datastore is then capped at $100\mathrm{M} - P - S - \mathrm{dict}$. This validation identified initial pair-count-based datastores for en↔zh and en↔ja that exceeded the limit by 0.7–1.8M tokens; Appendix A reports the corrected token caps.

Oracle isolation. ORACLE-DICT uses the reference translation and is restricted to evaluation. Module-level, constructor-level, and AST checks prevent the oracle component from being imported or instantiated by the training pipeline.

Determinism. Given a fixed seed, the augmenter produces identical context for a given source sentence across runs. RANDOM-CONTEXT and RANDOM-DICT each use a per-sentence deterministic sub-seed, so their controls are themselves reproducible.

Saved predictions. All metrics are computed from stored per-sentence hypotheses. This permits consistent rescoring across metrics and enabled the metric-label and dictionary-gate audits described in Appendices D and F.

When Does Depth Recursion Pay? A Parameter-Matched Study of Weight-Tied Hybrid Transformers on 100M Unique Words

Aditya Sasidhar
Independent Researcher
telikicherlaadityasasidhar@gmail.com

Abstract

Weight-tied depth recursion trades computation for effective depth without adding learned parameters. We pre-train 21 hybrid Gated DeltaNet (GDN) + grouped-query attention (GQA) transformers from scratch on the BabyLM 2026 Strict corpus under one tokenizer, data order, and optimizer, and use them to ask *on which budget axis* recursion actually pays. Under a matched **data-exposure** budget, recursion is a robust win *on language-model loss*: two exact parameter-matched pairs improve validation loss by 0.0252 and 0.0195 nats (means over three training seeds; seed sd 0.0015 and 0.0005), and a domain-balanced re-evaluation over the full 17.4M-token dev set puts the advantage at 0.0270 and 0.0196 nats, positive in all six dev domains under all three seeds (36 of 36 cells). Grammatical accuracy does not support a claim either way: replicated over the same three seeds the mean zero-shot BLiMP gain is $+0.58$ (sd 2.44) and $+1.84$ (sd 2.05) points, against a seed-noise floor of up to 2.80 points sd for a *single* architecture retrained—so the $+3.31$ and $+4.21$ obtained from one seed are not separable from training noise. Under a matched **compute** budget it is not: the same non-recursive twins trained to 1B tokens, using two thirds of the recursive models' training FLOPs, beat them by 0.0117 and 0.0088 nats. Two further controls sharpen the mechanism. Untying the re-applied layers—same depth, same FLOPs, three times the parameters—recovers a further 0.0467 and 0.0399 nats, pricing what tying costs. Replacing effective-depth residual scaling with unique-depth scaling changes loss by only 0.0034 and 0.0003 nats, so the gain comes from tying rather than from the initialization the two differ in. A 25M-token checkpoint grid locates the recursive models' crossover with their twins at 75–125M tokens in the 2:1 pair, before either model has completed one pass over the corpus, ruling out repeated examples as the mechanism there; the crossover moves later as effective depth grows, consistent with an early optimization cost. Recursion is therefore best understood as a **data-efficiency** mechanism for the fixed-corpus regime this benchmark defines, not a compute- or inference-efficiency one.

1 Introduction

The BabyLM challenge fixes the training data—roughly the number of words a human encounters during development—and asks what modeling choices make the most of it (BabyLM Organizing Committee, 2026). Most entries respond with data curricula or training objectives. We instead use the fixed-data setting for what it does best: making an *architectural* question unusually clean to answer.

> **If parameters are scarce, is it better to spend them on independent layers, or to re-apply a smaller stack of layers several times with tied weights?**

Weight-tied depth recursion—applying the same block of layers repeatedly—decouples a model's *effective depth* from its parameter count. The idea is old (Dehghani et al., 2019; Lan et al., 2020) and now spans from-scratch small models (Liu et al., 2024), large recurrent-depth pretraining (Geiping et al., 2025), conversion of pretrained models (Bae et al., 2024), and adaptive recursion (Bae et al., 2025).

The question is under-specified in a way that matters. "Scarce" can mean scarce *parameters*, *compute*, or *data*, and re-applying a stack three times holds parameters fixed while tripling compute—so a parameter-matched comparison alone can only say that more layer applications beat fewer at equal weight storage, which is close to definitional. We therefore measure the same intervention against three separate budgets, and the answer differs between them.

Proceedings of the Second BabyLM Workshop, pages 123–133
October 29, 2026 ©2026 Association for Computational Linguistics

We run the study inside a modern *hybrid* architecture that interleaves Gated DeltaNet linear-attention layers (Yang et al., 2025) with softmax grouped-query attention layers (Ainslie et al., 2023) in fixed ratios, following the design popularized by Qwen3-Next (Qwen Team, 2025). A hybrid *ratio unit* (e.g., GDN–GDN–GQA) is a self-contained token-mixing block, so recursion has an obvious granularity—repeat the unit.

Contributions.

1. **Recursion pays per word, not per FLOP.** At matched data exposure, two exact parameter-matched pairs favour recursion by 0.0252 and 0.0195 nats. At matched compute, non-recursive twins trained longer win by 0.0117 and 0.0088 nats while using two thirds of the FLOPs. Both directions are far outside training-seed variance (§5.2, §5.3).

2. **Three seeds, six domains, and a mechanism isolated by two controls.** Every headline loss comparison is replicated across three independently seeded runs (seed sd $\leq$0.0015 nats) and re-measured on every dev domain rather than a 2M-token prefix; the advantage is positive in 36 of 36 domain$\times$seed cells, and is largest on written text, smallest on speech (§5.2). An untied-depth arm prices what tying costs, and an initialization ablation shows the gain is not an artifact of effective-depth residual scaling (§5.4, §5.5).

3. **The crossover precedes the first epoch.** A 25M-token checkpoint grid places the 2:1 crossover at 75–125M tokens, before one corpus pass, eliminating repetition as an explanation; crossover time grows with effective depth (§5.6).

4. **BLiMP is too seed-noisy at this scale to carry an architectural claim.** Retraining one architecture under a different seed moves macro-accuracy by up to 5.56 points (sd 2.80), which swamps every architectural contrast in our grid; our own previously reported $+3.31$ does not survive replication. We report the noise floor so others can calibrate against it (§5.7).

Evaluation note. A training-time logger averaged per-*batch* means with batch size tied to a memory setting, biasing four logged losses by about -0.025. That logger has since been fixed to weight by rows; every validation number below is recomputed from checkpoints with uniform per-token weighting regardless. Appendix A documents the pitfall, its fix, and the calibration confirming both.

2 Related Work

Parameter sharing across depth. Universal Transformers (Dehghani et al., 2019) apply one block recurrently with adaptive halting; ALBERT (Lan et al., 2020) ties encoder layers. MobileLLM-LS pre-trains 125M- and 350M-parameter decoder-only models with immediate block-wise sharing (Liu et al., 2024). Geiping et al. (2025) scale a recurrent-depth decoder to 3.5B parameters. Relaxed Recursive Transformers convert pre-trained models and soften ties with layer-wise LoRA (Bae et al., 2024), while Mixture-of-Recursions learns token-specific depths (Bae et al., 2025). These studies establish that sharing works; none separates the parameter, compute, and data axes under a fixed corpus, which is what the BabyLM setting makes possible.

Subquadratic token mixers at BabyLM scale. This is not the first BabyLM entry to replace or dilute softmax attention. Haller et al. (2024) introduced BabyHGRN, which uses the HGRN2 recurrent architecture and outperformed transformer and other subquadratic baselines (LSTM, xLSTM, Mamba) under the 2024 Challenge budgets. Their follow-up BLaLM (Haller et al., 2025) replaces self-attention with a linear-time mLSTM token mixer plus short convolutions, sliding-window attention, and Hedgehog feature maps, reporting that linear attention combined with sliding-window attention improves zero-shot performance and that Muon converges more stably than AdamW. Our setting differs in two ways that matter: we retain a *minority* of full softmax layers rather than eliminating attention, and our object of study is the re-application schedule of a fixed layer inventory rather than the mixer itself. Their results are the relevant prior for the viability of subquadratic mixers at 100M words; ours addresses a question orthogonal to mixer choice.

Linear attention and the delta rule. DeltaNet-style linear attention maintains a fast-weight state updated by the delta rule (Yang et al., 2024). Gated DeltaNet adds a data-dependent decay gate

(Yang et al., 2025). We use the reference Triton kernels from `flash-linear-attention` (Yang and Zhang, 2024).

Hybrid architectures. A minority of softmax layers interleaved with a majority of linear-attention layers preserves most of full attention's quality at a fraction of its inference cost; Qwen3-Next (Qwen Team, 2025) adopts a 3:1 ratio at 80B scale. We evaluate 2:1 and 3:1 at BabyLM scale and use the ratio unit as the unit of recursion.

Sample-efficient pre-training. The BabyLM series (Warstadt et al., 2023; BabyLM Organizing Committee, 2026) fixes the corpus (Strict: 100M words; repetition allowed up to 1B words of exposure). Our primary runs consume ∼295M words each; the compute-matched arms consume ∼589M and ∼884M, all within budget.

3 Models

All variants share one spine: a decoder-only pre-norm transformer with $d_{\text{model}} = 768$, RMSNorm (Zhang and Sennrich, 2019, $\epsilon = 10^{-5}$), SwiGLU feed-forward blocks (Shazeer, 2020) with $d_{\text{ff}} = 2304$ (a fixed $3\times$ multiplier for *all* variants), tied input/output embeddings, and the BabyLM-community BPE tokenizer (vocabulary 16,384).

GQA layers use 12 query heads and 4 KV heads of dimension 64, rotary position embeddings (Su et al., 2021) with base 10,000, and per-head RMS QK-normalization (Henry et al., 2020). Softmax attention runs FlashAttention-3 (Shah et al., 2024) on Hopper with an SDPA fallback.

GDN layers are Gated DeltaNet layers (Yang et al., 2025) as implemented in `flash-linear-attention` v0.5.1: 12 heads of dimension 64, value expansion 1, output gating and short (size-4) depthwise convolutions enabled.

3.1 The variants

A *ratio unit* is (GDN, GDN, GQA) for 2:1 and (GDN, GDN, GDN, GQA) for 3:1. The non-recursive hybrids stack two units. The recursive hybrids contain **two independent super-blocks in sequence, each holding one ratio unit, each applied $R = 3$ times with weights tied within that super-block** (never across super-blocks). Unrolled, Recursive 2:1 computes 18 layer applications with the parameters of 6 layers.

Model	s42	s43	s44	sd	range
GDN 2:1	62.24	65.60	67.80	2.80	**5.56**
Recursive 2:1	65.56	64.22	67.62	1.71	3.39
GDN 3:1	63.27	66.05	65.19	1.42	2.78
Recursive 3:1	67.48	66.77	65.78	0.85	1.70

Table 1: BLiMP macro-accuracy of one architecture across three training seeds. The same models' validation loss varies by $\leq$0.0020 nats.

Dev domain	Tokens	Wins.	2:1	3:1
BNC Spoken	1.74M	424	+0.0267	+0.0207
CHILDES	5.84M	1426	+0.0139	+0.0099
Gutenberg	3.88M	946	**+0.0406**	**+0.0310**
OpenSubtitles	3.39M	828	+0.0238	+0.0190
Simple Wiki	2.29M	559	**+0.0405**	+0.0267
Switchboard	0.25M	60	+0.0164	+0.0100
Domain-balanced	17.39M	4243	**+0.0270**	**+0.0196**

Table 2: Paired recursion advantage in nats by dev domain, mean over three seeds (seed sd $\leq$0.0048 throughout). Positive favours the recursive model in every cell—36 of 36 domain×seed×family combinations.

Table 3 lists the inventory. Within each matched pair the two models have identical learned layer inventories and parameter counts *to the digit*; `src/common/param_count.py` is the executable source of truth and asserts this, along with the untied arms' $3\times$ layer scaling and the ablations' parameter identity.

3.2 Initialization, and what it does not explain

Weights are drawn from $\mathcal{N}(0, 0.02^2)$; residual-output projections are scaled by $1/\sqrt{2D_{\text{eff}}}$, where D_{eff} is the *effective* depth—18 or 24 for the recursive models, the plain layer count otherwise. Using effective rather than unique depth is a principled stability choice because the same projection writes into the residual stream R times.

The matched pairs therefore differ at initialization as well as in execution, so the comparison as originally constructed identifies the recursive *configuration* rather than tying alone. The *unique init* arms separate the two: parameter-identical to the recursive models to the digit, differing only in scaling residual-output projections by $1/\sqrt{2 \cdot \text{unique depth}}$, a $\sqrt{3}$ difference. Initialization accounts for 0.0034 and 0.0003 nats against recursion gains of 0.0252 and 0.0195 (§5.5)—real but small.

The untied deep arms use $1/\sqrt{2n_{\text{layers}}}$ with $n_{\text{layers}} = 18$ or 24, which *equals* their recur-

Variant	Mix	Uniq. layers	Eff. depth	Non-emb. params	Role
GQA baseline	pure GQA	10	10	68,830,208	no-GDN reference
GDN 2:1	2:1	6	6	46,913,888	twin of Recursive 2:1
GDN 3:1	3:1	8	8	63,487,504	twin of Recursive 3:1
Recursive 2:1	2:1	6	18	**46,913,888**	2 SB $\times$ R=3
Recursive 3:1	3:1	8	24	**63,487,504**	2 SB $\times$ R=3
Untied 2:1 deep	2:1	18	18	140,740,128	depth/FLOP-matched, untied
Untied 3:1 deep	3:1	24	24	190,460,976	depth/FLOP-matched, untied
Recursive 2:1 (uniq. init)	2:1	6	18	46,913,888	initialization ablation
Recursive 3:1 (uniq. init)	3:1	8	24	63,487,504	initialization ablation

Table 3: Model inventory. GDN 2:1 / Recursive 2:1 and GDN 3:1 / Recursive 3:1 are exact parameter-matched pairs. The untied arms deliberately are not parameter-matched: they hold depth and FLOPs fixed instead. Compute-matched arms are not new models—they are GDN 2:1 and GDN 3:1 trained to larger token budgets (§4).

sive counterparts' $1/\sqrt{2D_{\text{eff}}}$. Untied and recursive arms therefore share an initialization scale by construction and differ only in whether weights are tied.

4 Experimental Setup

Data. We train on the official BabyLM 2026 Strict corpus (six domain files, ∼100M words) and validate on the official dev split, excluded from the training bins at preprocessing time. Under the shared tokenizer one epoch is 169,741,563 tokens (≈1.70 tokens per whitespace word).

Recipe. One locked recipe for every run: 4,096-token sequences, 32 sequences per optimizer step, AdamW at peak 6×10^{-4} with cosine decay to 10%, and a 499,908,608-token primary budget (≈2.9 epochs ≈295M words). Appendix A gives the full hyperparameters and hardware.

Seeds. Each matched pair is trained at seeds 42, 43 and 44. A seed sets weight initialization *and* data order together, so the control holds within a seed group while the spread across groups is genuine run-to-run training variance.

Compute-matched arms. A recursive model executes ∼3× the layer FLOPs per token of its twin, so at 500M tokens it consumes ≈1.5B token-equivalents of training compute. We therefore train GDN 2:1 and GDN 3:1 to **1.5B tokens** (11,444 steps, 750 warmup, ≈884M words—exact FLOP parity) and to **1B tokens** (7,629 steps, 500 warmup, ≈589M words—two thirds of parity), each with the cosine schedule stretched to its own horizon and the warmup *fraction* held at ≈6.5%. Both remain inside the Strict track's 1B-word exposure cap. The 1B arm exists because the 1.5B arm overtrains (§5.3), so reporting only the FLOP-parity endpoint would flatter recursion.

Evaluation protocol. All validation numbers are recomputed from retained checkpoints over the same 488 non-overlapping 4,096-token windows, with uniform per-token weighting and an identical code path, for all 21 runs and 255 checkpoints. The 2M-token prefix was fixed to bound cost, not selected by model performance; because sorted-file concatenation makes it BNC Spoken plus part of CHILDES, final checkpoints are additionally scored on all six dev domains (§5.2).

5 Results

5.1 Overall picture

Read down Table 4's loss column and the ordering is not by parameters, not by effective depth, and not by tokens—it is by how the budget was spent. The rest of this section separates those axes.

5.2 Recursion at matched exposure

At the shared 500M-token budget the recursive member wins both pairs, and the effect replicates (Table 5). Within seed 42, a paired 100,000-resample bootstrap over the 488 aligned windows gives 0.0261 [0.0248, 0.0274] and 0.0195 [0.0183, 0.0207] nats, with recursion winning 480/488 and 462/488 windows (exact sign tests $p = 1.92 \times 10^{-130}$ and 2.64×10^{-104}). The published single-seed values sit essentially on the three-seed means. Establishing the seed floor at $\leq$0.002 nats is what makes the smaller effects below interpretable rather than suggestive.

The advantage holds in every dev domain. Because preprocessing concatenates the six dev files in sorted-filename order, the fixed 2M-token prefix used above is 87% BNC Spoken and 13% CHILDES; four of six domains contribute nothing to it. We therefore re-scored the final check-

Model	Non-emb. params	Tokens	Val loss ↓	Val ppl ↓	bpt ↓	BLiMP ↑	Suppl. ↑
Untied 3:1 deep	190.5M	500M	**3.0527**	**21.17**	**4.404**	67.50	59.53
Untied 2:1 deep	140.7M	500M	3.0605	21.34	4.415	68.17	59.26
GDN 3:1 (1B, annealed)	63.5M	1B	3.0838	21.84	4.449	67.71	55.95
Recursive 3:1	63.5M	500M	3.0926	22.03	4.462	67.48	55.17
Recursive 3:1 (uniq. init)	63.5M	500M	3.0929	22.04	4.462	67.54	54.55
GDN 2:1 (1B, annealed)	46.9M	1B	3.0955	22.10	4.466	65.58	57.95
Recursive 2:1	46.9M	500M	3.1072	22.36	4.483	65.56	53.30
GDN 2:1 (1.5B)	46.9M	1.5B	3.1081	22.38	4.484	67.56	**60.68**
Recursive 2:1 (uniq. init)	46.9M	500M	3.1106	22.43	4.488	65.75	56.08
GDN 3:1 (1.5B)	63.5M	1.5B	3.1120	22.47	4.490	66.82	56.32
GDN 3:1	63.5M	500M	3.1121	22.47	4.490	63.27	53.93
GQA baseline	68.8M	500M	3.1244	22.75	4.508	**71.07**	59.59
GDN 2:1	46.9M	500M	3.1333	22.95	4.520	62.24	52.28

Table 4: All arms on the fixed dev slice, uniform per-token weighting, seed 42. Every model trained on the same 100M-word Strict corpus; the Tokens column is exposure, not corpus size. The GQA baseline used the identical 500M-token budget as the other primary runs—only the arms marked 1B and 1.5B consumed more.

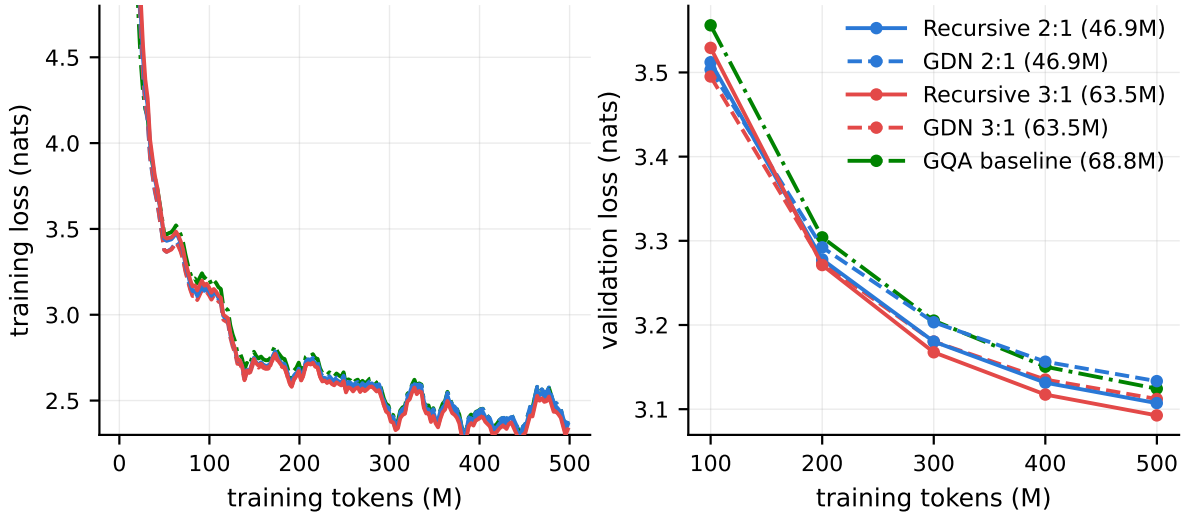


Figure 1: Training loss (left, smoothed, from training logs) and checkpoint validation loss (right, uniform protocol) versus training tokens for the five primary variants. Hue encodes the ablation pair (blue = 2:1, red = 3:1, green = baseline); solid lines are recursive models, dashed their non-recursive twins. All 21 runs in the study are stable, with zero loss spikes.

Pair	s42	s43	s44	mean	sd
2:1	+0.0261	+0.0236	+0.0261	**+0.0252**	0.0015
3:1	+0.0195	+0.0200	+0.0190	**+0.0195**	0.0005

Table 5: Paired recursion advantage (GDN minus Recursive) in nats, replicated over three independently seeded training runs. Per-model seed sd ranges from 0.0005 to 0.0020.

point of all thirteen 500M-token runs on the entire 17,390,678-token dev set, in windows contained within a single domain. The recursive model wins in **36 of 36** domain×seed×family cells; the domain-balanced advantage is +0.0270 (2:1) and +0.0196 (3:1), with per-domain values in Table 2 (Appendix A). Two things follow. First, the narrow slice was *conservative, not flattering*: the domain-balanced advantage (+0.0270, +0.0196) slightly exceeds the prefix figure (+0.0252, +0.0195), because the prefix is dominated by the two domains where recursion helps least. Second, the effect is domain-structured in the way a depth account predicts—roughly 3× larger on edited written text (Gutenberg, Simple Wiki) than on conversational speech (CHILDES, Switchboard).

On grammatical minimal pairs the single-seed picture appeared to agree: +3.31 [0.38, 6.85] points for the 2:1 pair and +4.21 [2.43, 6.11] for the 3:1, with the 3:1 effect directionally broad (46

Family	Recursive (500M, 3×)	Twin 1.5B (parity)	Twin 1B ($\frac{2}{3}$ parity)
2:1	3.1072	3.1081	**3.0955**
3:1	3.0926	3.1120	**3.0838**

Table 6: Validation loss under matched training FLOPs. The 1B-token arm uses two thirds of the recursive models' compute and still wins both families.

paradigms won, 20 lost, one tied; $p = .0019$) and the 2:1 mean driven by a positive tail (37 won, 30 lost, $p = .464$). Replicating both pairs across all three seeds dissolves most of this; §5.7 reports what is left.

5.3 Recursion at matched compute: the advantage reverses

Inverting the parameter-matched comparison, we hold compute fixed and let exposure vary (Table 6). At exact FLOP parity the 2:1 comparison is a tie (0.0009 nats) and the 3:1 favours recursion (0.0194). But that endpoint is misleading in the twin's disfavour: both 1.5B runs pass their minimum near 1.0B tokens (2:1 best 3.0922; 3:1 best 3.0733) and degrade thereafter as repetition accumulates past ∼6 epochs. Trained to 1B tokens with the cosine annealed to that horizon, the twins reach 3.0955 and 3.0838—beating the recursive models by **0.0117 [0.0097, 0.0137] and 0.0088 [0.0067, 0.0111] nats while consuming two thirds of the compute**. Recursion wins only 128/488 and 160/488 windows here. At 6–8× the seed sd, the reversal is not noise. BLiMP agrees: against the annealed twins recursion is level (-0.02 and -0.23) and loses on the supplement (-4.65 and -0.78).

Figure 2 shows both axes together. The two axes separate cleanly. Per word of exposure, recursion wins: it reaches 3.1072 on 295M words where the twin needs 589M to reach 3.0955. Per FLOP, it loses. Which conclusion matters depends on which budget binds—and on a fixed-corpus benchmark, the data budget is the one that does.

5.4 What tying costs

Table 7 reports the result: untying is worth 0.0467 and 0.0399 nats—roughly twice the recursion gain itself, for three times the weights. Tying is thus a real compression: it forfeits about 0.04 nats and returns a 3× parameter saving. The clearest evidence that this trade is favourable here is across architectures rather than within a pair: **Recursive 2:1, at 46.9M non-embedding parameters, reaches 3.1072 against the 68.8M-parameter GQA baseline's 3.1244**—better loss with 32% fewer learned weights (Figure 3). On BLiMP untying buys 2.61 points at 2:1 but nothing at 3:1 ($+0.02$), while both untied arms gain 4–6 points on the supplement.

Contrast	Δ loss	95% CI	Won
Untied vs Rec., 2:1	+0.0467	[.0447, .0488]	485/488
Untied vs Rec., 3:1	+0.0399	[.0379, .0419]	485/488

Table 7: The price of weight tying. Positive favours the untied model, which holds 3× the learned parameters at identical depth, FLOPs, and initialization scale.

Contrast	Δ loss	95% CI	Won
2:1 eff. vs uniq. init	+0.0034	[.0024, .0044]	305/488
3:1 eff. vs uniq. init	+0.0003	[−.0007, .0013]	257/488

Table 8: Effective-depth versus unique-depth residual scaling, holding tying fixed. Positive favours effective-depth scaling.

5.5 The initialization confound is small

In Table 8 the 3:1 contrast is a clean null: the interval spans zero and the sign test is at chance. The 2:1 contrast is small but detectable on evaluation-sample uncertainty—yet at roughly twice the seed sd of 0.0015, and with only 62.5% of windows won, it is not robustly identified either. BLiMP shows nothing in either family (-0.20, -0.06).

Effective-depth scaling therefore accounts for at most a seventh of the 2:1 recursion gain and none of the 3:1. The confound described in §3.2 exists but does not explain the result: **tying does the work**. We keep effective-depth scaling as the default on stability grounds rather than performance ones.

5.6 The crossover precedes the first epoch

Table 9 brackets the crossover. In the 2:1 pair it happens while both models are still inside their first pass over the corpus, so **repeated examples cannot be the mechanism there**. The 3:1 crossover sits near the boundary and remains ambiguous with respect to it. What varies systematically is effective depth: the 18-layer configuration crosses at ∼100M tokens and the 24-layer one at ∼175M, in both seeds. That ordering is what an early optimization cost predicts—deeper tied models pay more to get started—and is not what a repe-

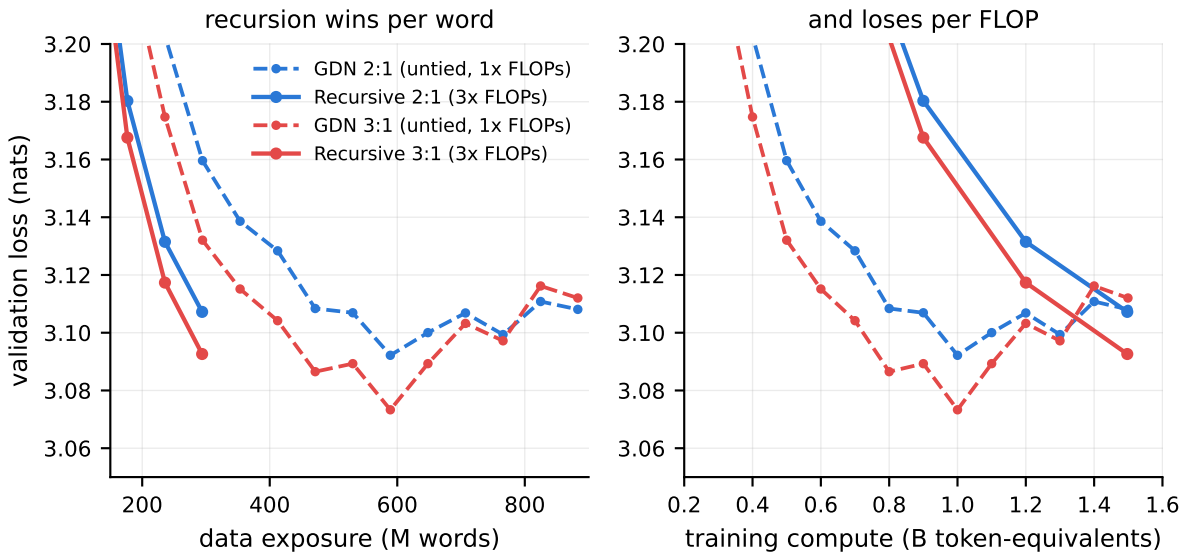


Figure 2: The same intervention against two budgets. **Left:** validation loss against data exposure. At equal exposure the recursive models (solid) sit below their untied twins (dashed); Recursive 2:1 reaches at 295M words what its twin needs ∼590M words to match. **Right:** the same runs against training compute in token-equivalents, where a recursive model costs 3× per token. Here the twins' curves run below the recursive endpoints, and both twins reach their minimum at ∼1.0B token-equivalents before overtraining degrades them. Hue encodes the family (blue = 2:1, red = 3:1).

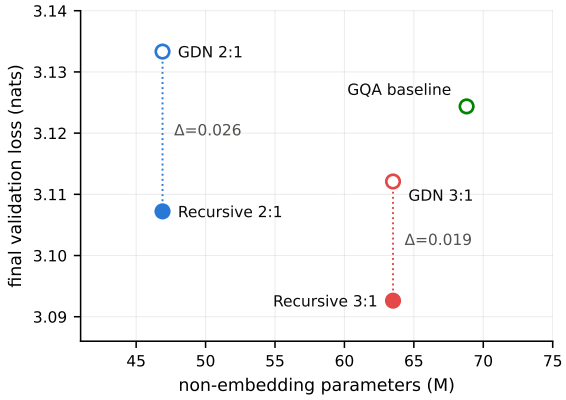


Figure 3: The parameter–quality trade-off for the primary grid. Vertical dotted lines connect the two parameter-matched pairs; filled markers are recursive models. Both recursive hybrids beat the larger pure-attention baseline on loss.

Family	seed 43	seed 44	vs. ep. 1
2:1 (D_{eff} 18)	75–100M	100–125M	**before**
3:1 (D_{eff} 24)	150–175M	175–200M	at / after

Table 9: Where the recursive model overtakes its twin, bracketed to 25M tokens. One corpus pass is 169.7M tokens.

Pair	Suite	s42	s43	s44	mean	sd
2:1	BLiMP	+3.31	−1.38	−0.18	**+0.58**	2.44
2:1	Suppl.	+1.01	−4.82	+2.54	**−0.42**	3.88
3:1	BLiMP	+4.21	+0.71	+0.59	**+1.84**	2.05
3:1	Suppl.	+1.24	+4.20	−0.25	**+1.73**	2.26

Table 10: Recursion advantage in macro-accuracy points, replicated over three seeds. Seed 42—the only seed evaluated in the original submission—is the most favourable of the three in both suites.

tition account predicts, since both models meet the epoch boundary at the same token count.

5.7 Grammatical accuracy does not replicate across seeds

Validation loss and BLiMP behave completely differently under seed replication, and the difference is large enough to change what this paper claims (Table 10).

The cause is not a defect of any one contrast but the noise floor of the benchmark at this scale. Retraining a *single* architecture under a different seed, changing nothing else, moves BLiMP macro-accuracy by 1.70 to 5.56 points (Table 1, Appendix A). A 5.56-point swing from the seed alone is larger than any architectural difference in our grid, and it swallows two cross-architecture readings we previously drew. The pure-GQA baseline leads BLiMP at 71.07 while sitting tenth of thirteen on loss, a 3.59-point margin; and GDN 2:1

appeared to gain 5.32 points from exposure alone (62.24 → 67.56, 295M to 884M words), which we had called the largest movement in the grid. Both are single-seed, and both lie inside the 5.56-point spread of one architecture, so we no longer claim either. For external scale the official BabyLM 2026 Strict GPT-2 reference reports 74.53 / 65.00, on a different architecture, recipe and exposure. The 2:1 recursion gain we previously reported at $+3.31$ is therefore withdrawn: it is one draw from a distribution whose spread exceeds it. The 3:1 gain keeps its sign in all three seeds and is the only accuracy claim we still make, at $+1.84 \pm 2.05$—directionally consistent, but with an interval that includes values near zero.

We state this as a finding rather than a limitation because the exposure is general: BabyLM-scale papers routinely compare architectures on single-seed BLiMP differences of one to three points, which our measurement says are not resolvable without replication.

The other two official zero-shot suites are also null. All thirteen 500M-token runs were also evaluated on the full official COMPS and entity-tracking sets. Recursion moves COMPS by -0.23 (sd 0.45) and $+0.21$ (sd 0.27), and entity tracking by -0.99 (sd 1.41) and -0.10 (sd 0.55); three of four contrasts change sign across seeds. More telling than the nulls is the absolute level: every model scores at or below the 20% chance level on entity tracking (16.79–19.11), far outside binomial noise over 6,780 items and not a length artifact—correct options and distractors both average 26.5 characters. The official GPT-2 reference reaches 23.58 there; COMPS sits at 53.5–54.7 against a 50% floor. Neither suite discriminates at this scale, so neither bears on the architectural question.

6 Discussion

Recursion is a data-efficiency mechanism. The two budget axes give opposite answers, and both are solid: $+0.0252/+0.0195$ nats per matched word, $-0.0117/-0.0088$ per matched FLOP. Tied re-application converts *additional computation* into quality at fixed parameter count, less efficiently than simply training the shallower model longer—but under a fixed corpus "longer" means repeating data, and repetition has a ceiling visible in our own runs. Recursion buys quality without spending exposure: where exposure is capped and compute is not, that is the useful trade; outside it, it is not.

Parameter efficiency survives on its own terms. Recursive 2:1 beats a 47% larger pure-attention baseline on loss, and untying costs $3\times$ the weights to buy 0.04 nats: under a weight-storage constraint—not a latency or FLOP one—recursion remains attractive, and §5.3 delimits rather than contradicts this. The 3:1 gap is the smaller of the two, most likely because its twin is less depth-starved at 8 unique layers and because the 24-deep tied model crosses over ∼75M tokens later (§5.6), leaving less budget to amortize its early deficit.

What the initialization result means. We adopted effective-depth scaling on stability grounds and it delivered stability—21 runs, zero loss spikes, smooth residual-RMS growth across passes. It did not deliver much accuracy. Practitioners should treat depth-aware residual scaling as a stability measure, not a performance lever, and should not attribute reported recursion gains to it.

Metric-dependent ranking, and which metric to trust. Validation loss rewards depth and parameters; BLiMP rewards the all-attention baseline. BLiMP morphology shows the baseline's largest margin (85.60 vs 78.80 for Recursive 3:1). We had read this divergence as an argument against selecting small models on dev loss alone. Seed replication inverts that advice: at this scale dev loss resolves architectures to ~ 0.001 nats while BLiMP moves by several points on the seed alone (§5.7). The divergence is real, but the sharper reading is that one of the two metrics is measuring the architecture and the other is largely measuring the draw.

7 Conclusion

Across 21 controlled runs, weight-tied depth recursion improves parameter-matched validation loss by 0.0252 and 0.0195 nats—replicated over three seeds, positive in all six dev domains. Under matched compute the advantage reverses. Tying itself produces the effect: untying recovers 0.0467 and 0.0399 nats at $3\times$ the parameters, and residual scaling accounts for at most 0.0034. No accuracy benchmark we ran resolves the contrast, so the claim rests on loss. Depth recursion pays where words are scarce and FLOPs are not—the regime BabyLM defines, and a narrower claim than "recursion helps."

Limitations

Grammatical accuracy is unresolved, not merely unreplicated. The paired models are now evaluated at three seeds (§5.7), and the result is that BLiMP's seed noise at this scale (up to 5.56 points range for one architecture) exceeds the effects we set out to measure. We therefore make no 2:1 accuracy claim and only a weak 3:1 one. Every other model in Table 4—the untied, unique-init and compute-matched arms and the baseline—is still evaluated at seed 42 alone, so all cross-architecture BLiMP comparisons here inherit that floor and should be read as provisional.

The checkpoint *series* is still on the narrow slice. Final-checkpoint numbers are now domain-balanced over the full 17.4M-token dev set (§5.2), but the 255-checkpoint trajectories behind Figure 1, the crossover brackets of §5.6, and the compute-matched curves of §5.3 are still computed on the 2M-token prefix, to bound evaluation cost. Since the per-domain deltas are same-signed and similar in magnitude, we do not expect the trajectories to reorder, but we have not verified that at every checkpoint.

Fine-tuning is not run. We report the four official zero-shot suites (BLiMP, BLiMP Supplement, COMPS, entity tracking) but none of the fine-tuning tasks, which would require a Hugging Face wrapper around our native checkpoints and an `attention_mask` argument our GQA implementation does not accept—a change to the one shared primitive all 21 runs depend on.

Compute matching is by token budget. We match FLOPs analytically and verify against measured throughput (Appendix A), but did not tune the learning rate per budget; all runs share the 6×10^{-4} literature prior. A tuned longer run might favour the twins further.

Narrow architectural slice. One width (768), two sizes, two hybrid ratios, $R = 3$ only. We do not vary R, and the untied arms exist at one depth each.

Inference cost is untouched. Naively unrolled decoding needs state for each application, increasing KV-cache and GDN recurrent-state storage. Parameter matching isolates quality per learned weight under a fixed weight-storage budget; it implies nothing about latency or inference-state memory.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. ArXiv:2305.13245.

BabyLM Organizing Committee. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. ArXiv:2602.20092.

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. 2024. Relaxed recursive transformers: Effective parameter sharing with layer-wise LoRA. ArXiv:2410.20672.

Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron Courville, and Se-Young Yun. 2025. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. ArXiv:2507.10524.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*. ArXiv:1807.03819.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. ArXiv:2502.05171.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*. ArXiv:2412.15978.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*. ArXiv:2511.05560.

Alex Henry, Prudhvi Raj Dachapally, Shubham Pawar, and Yuxuan Chen. 2020. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. ArXiv:2010.04245.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*. ArXiv:1909.11942.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong,

Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, Liangzhen Lai, and Vikas Chandra. 2024. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. In *International Conference on Machine Learning*. ArXiv:2402.14905.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*. ArXiv:1711.05101.

Qwen Team. 2025. Qwen3-next-80b-a3b: Towards ultimate training and inference efficiency. Model release and blog post, Alibaba Cloud.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. 2024. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. ArXiv:2407.08608.

Noam Shazeer. 2020. GLU variants improve transformer. ArXiv:2002.05202.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. RoFormer: Enhanced transformer with rotary position embedding. ArXiv:2104.09864.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, et al. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025. Gated delta networks: Improving Mamba2 with delta rule. In *International Conference on Learning Representations*. ArXiv:2412.06464.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. 2024. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*. ArXiv:2406.06484.

Songlin Yang and Yu Zhang. 2024. FLA: A triton-based library for hardware-efficient implementations of linear attention mechanisms. https://github.com/fla-org/flash-linear-attention.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*. ArXiv:1910.07467.

A Reproducibility details

Recipe. Sequence length 4,096; batch 32 sequences = 131,072 tokens per optimizer step. AdamW (Loshchilov and Hutter, 2019) with $\beta = (0.9, 0.95)$, weight decay 0.1 on matrices only, gradient clipping 1.0; peak learning rate 6×10^{-4} with cosine decay to 10% of peak; bf16 autocast with fp32 master weights. The primary budget is 3,814 steps = 499,908,608 tokens $\approx$ 2.9 epochs $\approx$ 295M words, with 250 warmup steps.

Hardware. Every run trains on a single H100-80GB (Modal); the full grid is $\sim$12 GPU-hours. All 21 runs completed with zero loss spikes (Figure 1).

Code and artifacts. Source code, model and variant definitions, the training and evaluation pipelines, and the evaluation artifacts behind every table and figure are available at `https://github.com/adityasasidhar/recursive-babylm`.

Table 11 reports per-run cost.

Throughput confirms the FLOP match. The untied deep arms are depth- and FLOP-matched to the recursive models by construction, and measured throughput agrees independently: 172,852 vs 180,660 tok/s at 2:1 and 133,451 vs 134,334 at 3:1. The compute-parity claim in §5.3 therefore does not rest on the analytic $3\times$ alone.

The evaluation-weighting pitfall, and its fix. The original training-time validation logger split the 488 evaluation windows into batches of `micro_batch_size` and returned the mean of per-batch means. For micro-batch 16, $488 = 30\times16+8$, so the trailing 8 windows were weighted double; for micro-batch 8, $488 = 61\times8$ exactly. The slice follows sorted-file order, and the last eight windows fall in an easier CHILDES region (1.60 nats against 3.15 for the preceding 480). Calibrating the five original runs against checkpoint evaluation recovers the bias exactly: $+0.0246$, $+0.0248$, $+0.0245$, $+0.0245$ nats for the four micro-batch-16 runs and -0.0000 for the micro-batch-8 run. The logger now weights by rows, and the new runs' logged values match checkpoint evaluation to four decimals. All paper numbers use uniform per-token weighting from checkpoints regardless of logger version. The lesson generalizes: *validation reductions must not depend on memory-layout settings*—and a correction calibrated on one code version must not be carried across a fix.

Model	Tokens	Wall clock	Throughput (tok/s)	Peak mem (GiB)	Micro-batch	Spikes
GQA baseline	500M	27.6 min	302,861	42.7	16	0
GDN 2:1	500M	18.8 min	443,084	33.6	16	0
GDN 3:1	500M	23.6 min	353,301	40.2	16	0
Recursive 2:1	500M	46.2 min	180,660	70.0	16	0
Recursive 3:1	500M	62.1 min	134,334	45.4	8	0
Untied 2:1 deep	500M	48.3 min	172,852	37.1	8	0
Untied 3:1 deep	500M	62.6 min	133,451	47.5	8	0
Recursive 2:1 (uniq. init)	500M	46.5 min	179,427	70.0	16	0
Recursive 3:1 (uniq. init)	500M	61.9 min	134,771	45.4	8	0
GDN 2:1 (1B, annealed)	1B	36.1 min	462,782	33.6	16	0
GDN 3:1 (1B, annealed)	1B	44.8 min	372,349	40.2	16	0
GDN 2:1 (1.5B)	1.5B	53.3 min	469,173	33.6	16	0
GDN 3:1 (1.5B)	1.5B	66.5 min	376,515	40.2	16	0

Table 11: Per-run cost. All runs: 1× NVIDIA H100 80GB HBM3 (Modal), CUDA 13.0, Python 3.11, `flash-linear-attention` 0.5.1, FlashAttention-3 for GQA layers, bf16 autocast, fp32 master weights, fused AdamW. Seed replicates (seeds 43, 44 of the four paired models) are omitted for space and match their seed-42 counterparts within measurement noise.

Paired uncertainty. `paper/figures/paired_uncertainty.py` uses 100,000 paired nonparametric resamples with seed 20260714. Validation resamples the 488 aligned windows; BLiMP resamples aligned UIDs. Every contrast is signed so that positive favours the treatment arm. Bootstrap intervals measure only finite evaluation-sample uncertainty and assume exchangeable units despite domain/order correlations; the separate `seed_replication` block reports across-seed spread, the only estimate of training variance here.

BLiMP protocol. `src/common/blimp_eval.py` reproduces the official BabyLM 2026 causal scorer directly against native checkpoints: summed log-probability over every non-BOS sentence token, temperature 1, macro-averaged by UID (Warstadt et al., 2020). Evaluator commit `3d57ddc8` and data revision `8d52da94` are pinned and identical across all 13 evaluated models. Seven of 59,875 BLiMP pairs tied under every model and were resolved with a fixed seed.

Extra logging for the recursive models. Per-super-block gradient norms and per-recursion-pass activation RMS were logged throughout; activation RMS grows smoothly and roughly linearly across the three passes (Recursive 2:1: $1.06 \to 1.60 \to 2.12$ in super-block 0 and $2.79 \to 3.82 \to 5.01$ in super-block 1) with no sign of divergence, consistent with the $1/\sqrt{2D_{\text{eff}}}$ residual initialization.

Parameter Golf: What Really Works?

Prashanna Mani Paudel and Shivanand Venkanna Sheshappanavar
Geometric Intelligence Research Lab.,
Department of Electrical Engineering and Computer Science
University of Wyoming
{ppaudel,ssheshap}@uwyo.edu

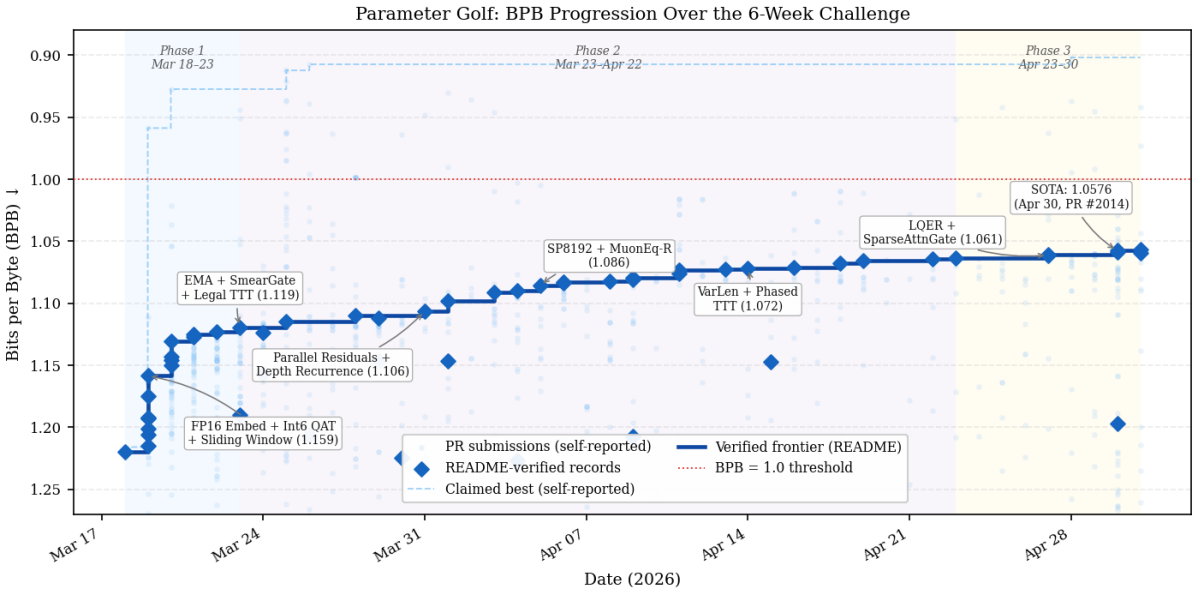


Figure 1: **Parameter Golf: 43 days, 1,810 submissions, BPB improved from 1.2244 to 1.058 under a fixed artifact-size budget.** Shaded bands (three improvement phases). Labeled points (README-verified milestones).

Abstract

How far can a language model improve under a strict artifact budget? *Parameter Golf* posed this question as an open community challenge in which participants trained the best language model, with the complete artifact (training code + compressed weights) required to fit within 16 MB and to be trained in under ten minutes on 8×H100 SXM GPUs. Quality was measured in bits-per-byte (BPB), the average number of bits required to encode each byte of unseen text. We analyze 2,037 pull requests and 1,430 clean-scored submissions from the contest, build a taxonomy of 84 optimization techniques, and measure each technique's contribution to BPB. The verified leaderboard score dropped from 1.2244 to 1.058 BPB across three phases—a 13.6% reduction, despite individual techniques rarely improving BPB by more than 1%. We show that most techniques' gains shrink when re-measured among competitive submissions, isolating the few methods that help regardless of the surrounding stack. Code and data are available at `github.com/PMP56/pmgolf-analysis`.

1 Introduction

Language model research largely follows a scaling paradigm—more parameters, data, and compute yield better performance. Yet this approach is impractical in settings with strict memory and latency constraints, including on-device inference, privacy-preserving local LMs, and real-time embedded systems operating on kilobyte-to-megabyte budgets. This work explores what is achievable within these extreme limits and how closely current methods approach that frontier. This question is central to BabyLM (Warstadt et al., 2023), which likewise asks how much language-model capability a fixed, severe resource constraint allows; *Parameter Golf* constrains artifact size and training time where BabyLM constrains the training data.

Parameter Golf is an open community challenge posted by OpenAI in March 2026 (OpenAI, 2026)

Proceedings of the Second BabyLM Workshop, pages 134–147
October 29, 2026 ©2026 Association for Computational Linguistics

that provides a setting to study this problem. Participants compete to minimize bits-per-byte (BPB) on the FineWeb (Penedo et al., 2024) validation dataset, subject to two hard constraints: (i) the complete submission artifact (training code + compressed model weights) must not exceed 16 MB, and (ii) training must complete in under ten minutes on $8\times$H100 SXM GPUs. The metric BPB is tokenizer-agnostic, allowing a fair comparison of submissions with different vocabularies. The challenge drew over 2,000 submissions from the global ML community in six weeks, creating an unprecedented public record of incremental discovery.

Why is this scientifically valuable? Traditional empirical ML research often takes months or years to reach publication, with papers reporting only final outcomes. Under tight resource constraints, *Parameter Golf* compresses this cycle to days: each submission becomes a natural ablation against the current frontier. The resulting history forms a structured dataset for *retrospective meta-analysis*, enabling the estimation of technique-level BPB effects across thousands of submissions.

2 Related Work

2.1 Constrained LM Challenges

Parameter Golf belongs to a family of community challenges built around the scaling triad $L(N, D, T)$, where N is the model size, D is the dataset size, and T is the training compute. Each challenge fixes one of these axes and asks participants to optimize a model within that constraint. NanoGPT Speedrun (Jordan, 2024) fixes a target validation loss and minimizes training time; NanoGPT Slowrun (Q Labs, 2025) fixes the dataset and minimizes the achievable loss. BabyLM (Warstadt et al., 2023) fixes a 10M- or 100M-word corpus and asks participants to train the best model on it. TinyStories (Eldan and Li, 2023) pushes the parameter axis to the extreme low end, showing coherent English emerges at sizes far below the frontier. Together, they shift attention from "how big can we scale?" to "how much can we achieve within a fixed constraint?"

2.2 Transformer Ablation Studies

Architectural improvements are typically evaluated in isolation, leaving open the question of whether they remain effective when combined. Narang et al. (2021) is the most direct precedent: they tested dozens of published modifications under matched conditions and found that almost none robustly outperformed a vanilla Transformer. Our within-frontier analysis (Section 6) reaches a similar conclusion from a complementary angle: rather than rerunning controlled experiments, we ask which techniques distinguish the strongest submissions from one another. Both studies found that most architectural changes look promising in isolation but not in an already competitive stack. Primer (So et al., 2021) uses neural architecture search to find the small set of changes that do survive. The scaling-law literature (Kaplan et al., 2020; Hoffmann et al., 2022) supports this view: within a reasonable architectural family, the dominant levers are model size, data, and compute. Parameter Golf is consistent with that: when one axis is locked, what separates the frontier is not architectural novelty but careful allocation of the remaining budget.

2.3 Hybrid and Non-Parametric LMs

The within-frontier analysis finds that classical n-gram blending is the only technique to retain a clearly positive gain across already-competitive submissions. This result has a long pedigree. Katz backoff (Katz, 1987) and PPM (Cleary and Witten, 1984) predate neural LMs by decades and remain strong at byte-level prediction. Modern work has revisited the same idea under different framings: kNN-LM (Khandelwal et al., 2020) adds a nearest-neighbor cache over training-set continuations to improve perplexity; RETRO (Borgeaud et al., 2022) scales the same idea to trillion-token retrieval; and Infini-gram (Liu et al., 2024) shows a large n-gram model remains useful even alongside frontier LMs. These results suggest that the neural model and the n-gram lookup hold complementary information and that combining them yields performance gains that neither could achieve alone.

3 Dataset and Methodology

3.1 Data Collection

We collected all 2,037 pull requests from the `openai/parameter-golf` repository (March 18–April 30, 2026) via the GitHub REST API. We downloaded each PR's `README.md` and `submission.json`, which carry reported scores and architecture metadata. De-duplicating to one row per PR leaves 1,810 rows. For each, we extract a score from five sources in priority order, stopping at the first match (Appendix Figure 6): the official leaderboard, the `val_bpb` field

Table 1: Score-source tiers for the 1,810 submissions.

Tier	Count	Source
Verified leaderboard	61	OpenAI-verified
Submission-JSON	1,268	`val_bpb` field
README-extracted	104	README headline
PR-body	31	self-reported
LLM-reviewed	14	LLM-recovered
Unscored	332	No score found
Total	**1,810**	

of `submission.json`, a README score, PR body text, and finally an LLM reviewer, which adjudicates the 481 PRs the first two sources miss (Section 3.4). In total, 1,478 submissions carry a valid score; excluding 48 with BPB below 0.9 leaves **1,430 clean-scored submissions** used in all subsequent analyses.

3.2 Technique Classification

For each PR, we detect techniques using deterministic keyword matching across three text sources: the PR title, the README body, and the `submission.json` metadata, which encodes architectural hyperparameters. README mining identified 84 binary technique flags across five categories: Quantization, Architecture, Tokenization, Training, and Evaluation, listed in full in Appendix A.4.

3.3 Technique Impact Metric

For each technique k, we compute:

$$\Delta_k = \overline{\text{BPB}}_{k=0} - \overline{\text{BPB}}_{k=1} \tag{1}$$

where $\overline{\text{BPB}}_{k=0}$ and $\overline{\text{BPB}}_{k=1}$ are the mean BPB across clean submissions not using and using technique k, respectively. A positive Δ_k means users of technique k have a lower (better) BPB on average. We compute Δ_k over the 1,430 clean-scored submissions, excluding the 48 below 0.9 BPB, which are suspected of validation-data leakage and whose abnormally low scores would inflate the apparent gain of any technique they use.

This metric is observational, not causal, and two systematic biases affect it in opposite directions. Techniques introduced late appear only in submissions that already incorporate all prior improvements, inflating their apparent Δ_k. Near-universal techniques such as sliding-window evaluation, Quantization-Aware Training (QAT), and Exponential Moving Average (EMA) yield near-zero Δ_k due to the lack of a competitive "without" group. Δ_k should therefore be read as a correlation with strong submissions, not a controlled ablation. To separate the two, Section 6 recomputes Δ_k within the competitive frontier alone, and we discuss the confounding explicitly per technique.

3.4 Accuracy of BPB Score and Technique Detection

Score validation: We validated the extracted BPB score in two passes. A sample of 70 PRs, stratified across sources, was manually reviewed against the parser output. One miss was found, yielding 98.6% accuracy for *score extraction*; this was later fixed. This accuracy refers only to score parsing; the 84 technique flags are validated separately, as detailed below. The 481 PRs not covered by leaderboards or `submission.json` are more error-prone, so a language-model reviewer double-checked them. Given the PR's README and `submission.json` candidates, it assessed whether each candidate's score was genuine. Of the 481, 389 carried a regex-extracted candidate from the README or PR body text: the reviewer confirmed 135, corrected 14 parser errors, and rejected 240 for lacking a valid score. The remaining 92 carried no recoverable score. The 149 retained scores (135 confirmed plus 14 corrected) match the tiers extracted from the README, PR body, and LLM review, as shown in Table 1.

Technique validation: For technique presence, all 61 verified leaderboard entries were checked by hand, and the keyword detectors were tightened after auditing their matches on all 1,810 PRs to remove substring false positives. Three residual error sources remain: code comments that name a technique without implementing it, non-standard terminology, and multi-contribution PRs with an ambiguous primary technique. Since we did not hand-label every flag, some noise remains, so we treat each Δ_k as approximate, relying on the *ranking* of techniques and the within-frontier test (Section 6) rather than small differences in values.

4 The Parameter Golf Challenge

4.1 Evaluation Metric: Bits-per-Byte

BPB measures how many bits a language model needs to encode each byte of a corpus under an arithmetic coding scheme derived from the model's predicted probability distributions. Formally, for a model assigning token probability p_t at position t:

$$\text{BPB} = \frac{-\sum_t \log_2 p_t}{B_{\text{bytes}}} = \frac{\text{NLL}_{\text{nats}}}{\ln 2 \cdot \text{BPT}} \qquad (2)$$

where B_{bytes} is the corpus byte count, $\text{NLL}_{\text{nats}} = \frac{-\sum_t \ln p_t}{T_{\text{tokens}}}$ is the mean per-token negative log-likelihood in nats (the natural-logarithm analog of bits), and BPT is bytes per token. The BPT factor makes the BPB tokenizer-agnostic. A 1,024-token vocabulary (BPT $\approx$ 2.4) and an 8,192-token vocabulary (BPT $\approx$ 3.7) are compared on the same scale, since BPB accounts for how efficiently each tokenizer encodes the data, not just how well the model predicts the next token.

4.2 Baseline Architecture

The challenge started with a baseline submission model implementing a compact transformer: 9 transformer layers, a SentencePiece tokenizer with a vocabulary size of 1024, int8 post-training quantization, and zlib compression of the weight file. This achieved 1.2244 BPB, defining the starting point from which all improvements are measured.

4.3 Score Distribution

Figure 2 shows how BPB scores are distributed across the 1,430 clean-scored submissions. While Figure 1 traces the record-setting frontier, this distribution reveals where the broader community clustered. Most submissions landed in the 1.05–1.20 range, with the competitive frontier (1.05–1.10) containing most verified records. Only a few self-reported submissions crossed the BPB = 1.0 threshold, and none are README-verified.

5 A 3-Phase Analysis of Parameter Golf

We identified three qualitatively distinct improvement phases, each initiated by one or two novel techniques that the community rapidly adopted and built subsequent versions on. We characterize the phases in Table 2, show how each phase's verified improvement splits across technique categories in Figure 3, and describe them below; per-phase technique-impact rankings (Δ_k) are given in Appendix A.2 (Figure 8).

Appendix Figure 9 tracks technique adoption across 1,810 submissions: quantization reached nearly 90% by Phase 1, evaluation methods emerged at the Phase 1/2 boundary, and tokenization exceeded 70% by Phase 3, aligning each category's rise with its defining phase.

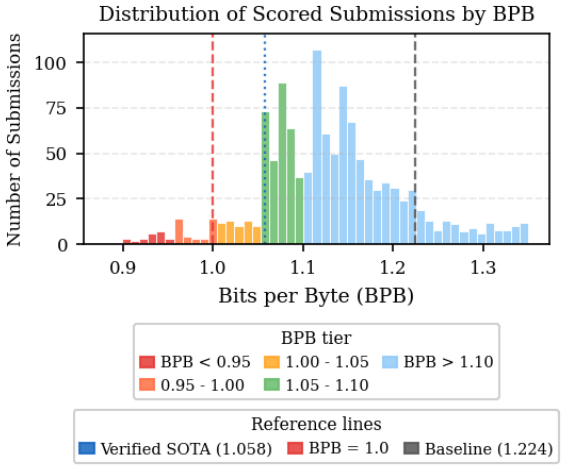


Figure 2: Distribution of BPB scores across the 1,430 clean-scored submissions; 48 leakage-suspected entries below 0.9 BPB are excluded.

5.1 Phase 1: Initial Stack Adoption

The challenge quickly showed that the evaluation strategy can matter as much as model quality. Two major gains in this phase required no architectural or training changes:

5.1.1 Evaluation strategies

Sliding window evaluation: Standard chunked evaluation scores non-overlapping 1024-token blocks independently, so early tokens get near-zero context, and the rest average only 512 tokens of context. This inflates BPB by misaligning evaluation with training. A stride-64 sliding window instead gives each scored token 960 tokens of left context without modifying the model. It decreased BPB by 0.032, the largest improvement with no architectural changes. The cost is 16$\times$ more forward passes, making evaluation rival training directly.

FP16 tied embeddings: The int8 baseline lost precision in the output projection, where small logit errors can substantially increase BPB. Retaining FP16 (16-bit floating point) embeddings preserves accuracy with negligible size overhead. The dataset-wide $\Delta_k = +0.036$ across 131 submissions is small, as FP16 embeddings were largely superseded once later stacks moved to int6 GPTQ (Frantar et al., 2023), a post-training quantizer that rounds each layer's weights to minimize its output error.

5.1.2 Compression and Quantization

The first phase of the challenge focused on implementing various quantization strategies to free up

Table 2: Three-phase improvement trajectory. BPB decreases steadily across phases (Best result is 1.058 in Phase 3).

Phase	Dates	BPB	Dominant techniques
1. Initial Stack Adoption	3/18–3/23	1.224 → 1.12	Sliding window, FP16 embed, Int6 QAT-STE, SmearGate, BigramHash, XSA
2. Architectural Diversification	3/23–4/22	1.12 → 1.064	SP8192, depth recurrence, parallel residuals, MuonEq-R, VarLen attention, Legal TTT, Gated DeltaNet, Brotli
3. Hybrid Emergence	4/23–4/30	1.064 → 1.058	N-gram backoff, PPM byte mixture, TTT-SLOT, TTT (Global/Phased), LQER, sparse attention gate, BOS-boundary fix

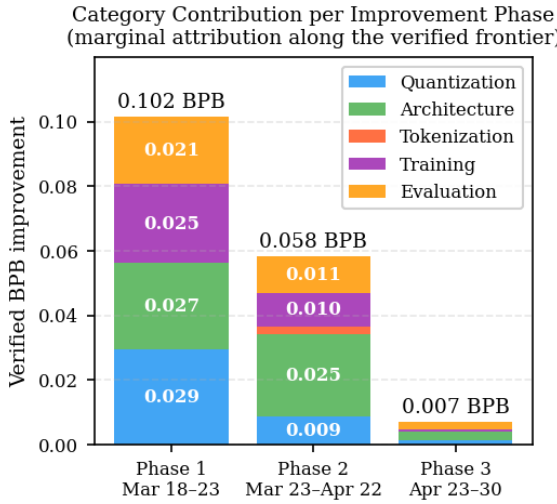


Figure 3: Category contribution per phase. Each record's BPB drop over the previous record is split across its newly added techniques in proportion to their all-data Δ_k (negatives clamped to zero; drops with no positive-Δ_k addition split equally). Bar height is the phase's total verified improvement; segments are the per-category shares. The weighting inherits Δ_k's adoption-timing bias, so the split is indicative rather than exact.

space and add more layers to the model, thereby improving its response quality. The implementations that proved to be the most valuable are:

Int6 Quantization-Aware Training with Straight-Through Estimator: Switching from int8 to int6 reduces storage to 0.75 bytes per weight, freeing roughly 4 MB that funded MLP expansion from 2× → 3× and two extra transformer layers (9→11). Since naive int6 post-training degrades accuracy, Quantization-Aware Training (QAT) (Jacob et al., 2018) simulates quantization in the forward pass, while the Straight-Through Estimator (STE) (Bengio et al., 2013) passes gradients through the rounding step unchanged. Int6 reached 968 submissions at $\Delta_k = +0.130$, with QAT-STE in 575, carrying the verified leaderboard to approximately 1.146 BPB.

SmearGate: In capacity-constrained models, high-variance activations in the residual stream propagate through the MLP block and waste computation on uncertain directions. SmearGate adds a learnable soft gate that down-weights these high-variance directions before they reach the MLP, reducing misallocated computation without removing any model components. It reached 548 submissions at a dataset-wide $\Delta_k = +0.105$.

BigramHash Embeddings: Standard embeddings give the model no signal about local token co-occurrence. BigramHash embeddings augment each token embedding with a learned bigram hash table, adding a cheap inductive bias for short-range dependencies without extra transformer layers. BigramHash reached 635 submissions at $\Delta_k = +0.081$, with negligible compute overhead since the hash table is a simple embedding lookup.

5.1.3 Attention and Averaging

Following the compression-driven gains of the first phase, three independently mergeable contributions arrived in rapid succession between March 20 and 23, completing the canonical Phase 1 stack:

XSA (Exclusive Self-Attention): XSA replaces full attention in the deepest transformer blocks with a parameter-reduced exclusive-attention variant, freeing bytes that can be reallocated to MLP width or additional layers. This makes XSA a capacity reallocation technique rather than a pure architectural improvement. XSA reached 571 submissions with a dataset-wide $\Delta_k = +0.119$, among the highest of any single technique in this phase.

EMA Weight Averaging: Stochastic gradient descent produces a noisy parameter trajectory whose final checkpoint is rarely the best generalizing point. Exponential moving average (EMA) weight averaging keeps a running average of checkpoints, producing a smoother estimate that generalizes better, in the spirit of stochastic weight averaging (Izmailov et al., 2018). It is applied post-hoc and requires no retraining. EMA reached 767 submissions at $\Delta_k = +0.123$, reflecting consistent im-

provement and easy integration into existing stacks.

Legal Score-First Test-Time Training: Test-time training (TTT) (Sun et al., 2020) updates model weights directly on the evaluation corpus before scoring, adapting the model to the target distribution at inference. This variant constrained updates to techniques permitted under the challenge rules, making it the first validated instance of TTT in the competition. Gains were modest initially, but adoption accumulated through Phase 2 to 413 submissions at $\Delta_k = +0.143$, foreshadowing the unrestricted adaptation that would dominate Phase 3.

Phase 1 closed at approximately 1.1194 BPB (March 23) with the community converging on a canonical stack: Int6 QAT-STE, sliding-window eval, XSA, EMA, and SmearGate. Nearly all later submissions treated this stack as the new baseline.

5.2 Phase 2: Architectural Diversification

Phase 2 spanned 30 days and improved BPB by approximately 0.06, against 0.10 in the five days of Phase 1. The slowdown is understandable: Phase 1 gains came from fixing evaluation errors and byte inefficiencies, both of which yield substantial improvements at a low implementation cost. Phase 2 required real architectural and training innovations that are harder to find and implement for smaller marginal gains against an already strong base.

5.2.1 Vocabulary Expansion and Training

SP8192 Vocabulary: The baseline tokenizer used a vocabulary of 1,024 tokens, covering approximately 2.4 bytes per token on the evaluation corpus. Expanding to an 8,192-token SentencePiece (Kudo and Richardson, 2018) vocabulary raises this to approximately 3.7 bytes per token, a 53% increase. Since BPB scales inversely with bytes per token at a fixed per-token NLL, the longer tokens would lower BPB by 35% if the model predicted them as well as before. In practice each token carries more information, so per-token NLL rises and the realized gain is smaller, and the eightfold larger embedding table costs bytes within the 16 MB budget. SP8192 reached 351 submissions at $\Delta_k = +0.115$, among the largest Phase 2 gains needing no architectural change.

Depth Recurrence: Depth recurrence loops over selected existing layers multiple times, creating virtual depth from a fixed parameter budget at zero byte cost. The tradeoff is one full forward pass per extra loop. Depth Recurrence reached 405 submissions at $\Delta_k = +0.079$. The relatively modest gain reflects that recurrence introduces optimization difficulties alongside the capacity benefit.

Parallel Residuals (GPT-J Style): In standard transformer blocks, the attention sublayer modifies the residual stream before the MLP sees it. Parallel residuals run both sublayers on the same unmodified input, a variant popularized at scale by PaLM (Chowdhery et al., 2023), improving gradient flow at the same parameter count. Parallel residuals reached 237 submissions with a dataset-wide $\Delta_k = +0.136$, suggesting that sequential coupling between attention and the MLP is a meaningful bottleneck at this scale.

MuonEq-R: MuonEq-R is an orthogonalized variant of the Muon optimizer (Jordan et al., 2024) with equalized update norms, which enforces more uniform parameter utilization. MuonEq-R reached 146 submissions at $\Delta_k = +0.127$, suggesting that the choice of optimizer is a larger lever than commonly assumed at this scale.

5.2.2 Non-Transformer Architectures and Compression Implementations

Variable-Length Attention (VarLen): VarLen processes variable-length document boundaries within a single forward pass, eliminating the padding waste that fixed-length attention incurs when documents are shorter than the context window. VarLen reached 34 submissions with a dataset-wide $\Delta_k = +0.150$. The low submission count relative to the high Δ_k suggests that VarLen was difficult to implement correctly, limiting adoption despite a strong signal.

Gated DeltaNet (GDN): Gated DeltaNet (Yang et al., 2025) replaces attention entirely with a gated delta-rule recurrence that maintains a fixed-size hidden state, so the full artifact budget goes to model weights rather than a growing key-value cache. GDN reached 50 submissions at a dataset-wide $\Delta_k = +0.076$, with a best reported (unverified) score near 0.998, showing that non-transformer architectures can be competitive at this scale.

Brotli Weight Compression: Switching from zlib or zstd-22 compression to Brotli (Alakuijala et al., 2018) achieves better compression on neural weight distributions, freeing extra bytes within the 16 MB constraint with no model changes. Brotli reached 310 submissions with a dataset-wide $\Delta_k = +0.149$. Its high adoption reflects that the switch is a single-line change in the artifact pipeline.

CaseOps/Casefold Tokenizer: BPE tokenizers treat case variants as distinct tokens. Applying

Unicode case-folding before BPE segmentation merges these otherwise-separate tokens, reducing vocabulary sparsity without increasing vocabulary size. CaseOps reached 135 submissions with a dataset-wide $\Delta_k = +0.135$, among the highest in the dataset. This is unusually large for a preprocessing step, suggesting the baseline tokenizer handled the corpus's case-variant sparsity poorly.

By late April, Phase 2 reached approximately 1.064 BPB. Further gains now require combining multiple complementary techniques rather than any single fix, setting the stage for the hybrid, carefully coordinated stacks of Phase 3.

5.3 Phase 3: Hybrid Emergence

Phase 3 was qualitatively different from Phase 2. Rather than further refining the neural stack, the top submissions increasingly combined the neural language model with classical n-gram sequence prediction and specialized attention mechanisms that directly target the evaluation setting. In eight days, Phase 3 produced a 0.007 BPB improvement in the verified frontier, concentrated in a small number of highly-engineered record-setting stacks rather than broad community adoption.

N-gram Blending: Phase 3's defining move was to mix neural LM predictions with a classical n-gram model at the byte level via a learned interpolation coefficient: $P_{\text{mix}} = \lambda P_{\text{neural}} + (1-\lambda) P_{\text{ngram}}$. Two forms of n-gram blending appeared: an n-gram *cache* over recently-seen contexts and an n-gram *backoff* (Katz, 1987) that falls back to shorter contexts, in the manner of PPM (Cleary and Witten, 1984). Both add zero learnable parameters and store compactly as frequency tables. N-gram backoff reached 28 submissions at $\Delta_k = +0.180$, the highest among all techniques used. The cache form reached 218 submissions at $\Delta_k = +0.147$, and the PPM byte-mixture reached 57 at $\Delta_k = +0.111$.

Sparse Attention Gate: The Sparse Attention Gate applies a learned sparsity mask to the attention score matrix before softmax, zeroing low-magnitude entries and concentrating capacity on high-salience pairs. It reached 71 submissions with a dataset-wide $\Delta_k = +0.159$, the second-highest in the dataset. Its late introduction means the value reflects a correlation with a strong late-stage base rather than an isolated contribution. All four lowest verified BPB scores include this technique.

TTT-SLOT (Sequence-Local Online Training): Where standard TTT adapts weights to the whole evaluation corpus before scoring, TTT-SLOT applies local weight updates within each scored sequence chunk, adapting to each document before predicting. It reached 242 submissions with a dataset-wide $\Delta_k = +0.139$, the highest adoption of any Phase 3 technique.

TTT (Global/Phased): Global/Phased TTT adapts the model to the whole evaluation corpus before scoring, rather than per chunk. The phased variant uses a staged schedule: an initial short-context pass, then passes that extend context and reduce the learning rate. It reached 101 submissions at dataset-wide $\Delta_k = +0.146$. The final SOTA scores use phased TTT with sequences up to 2,816 tokens, driving the verified frontier from 1.061 to 1.058.

LQER (Learned Quantization Error Reduction): LQER (Zhang et al., 2024) applies a learned post-training correction to the quantization error left by GPTQ (Frantar et al., 2023), recovering precision loss without retraining. It is composable with any quantized model, including the Int6 QAT-STE stacks from Phases 1 and 2. LQER reached 96 submissions with a dataset-wide $\Delta_k = +0.142$.

BOS-Boundary Fix: An off-by-one error in how beginning-of-sequence (BOS) boundaries were handled during evaluation introduced a systematic scoring bias for documents starting near chunk boundaries. The fix required no model changes, only a correction to the evaluation loop, showing that measurement errors were still silently inflating BPB even in the final phase.

The verified frontier reached 1.058 BPB, the lowest within-deadline score; a later 1.057 entry was submitted on May 1 and accepted by OpenAI.

6 Technique Impact Summary

First, the largest gains come from techniques that step outside the neural model. N-gram backoff leads the dataset at $+0.180$, with the cache form at $+0.147$, and these classical sequence-prediction methods sit alongside the Sparse Attention Gate and the evaluation-time TTT variants at the top of Table 3. Classical n-gram mixing and direct adaptation to the evaluation distribution outrank any single architectural change.

Second, several widely adopted techniques still post high ΔBPB. GPTQ and Brotli both compress the weight file to free byte budget, and CaseOps tokenization reaches $+0.135$ across 135 submissions. High values at high adoption reflect that these became part of the canonical stack, so the small "without" group is dominated by weaker

Table 3: Selected techniques from the top and bottom of the all-data Δ_k ranking across the 1,430 clean-scored submissions; the full top 22 appears in Appendix Figure 7. A positive ΔBPB = lower BPB (better). Techniques not described in Section 5 are defined in Appendix A.4.

Technique	Cat.	Count	ΔBPB
N-gram Backoff	Eval	28	+0.180
Sparse Attention Gate	Arch	71	+0.159
GPTQ	Quant	664	+0.155
AsymLogit	Arch	28	+0.155
Attention Output Gate	Arch	38	+0.151
Variable-Length Attention	Arch	34	+0.150
Brotli	Quant	310	+0.149
N-gram Cache	Eval	218	+0.147
TTT (Global/Phased)	Eval	101	+0.146
LQER	Quant	96	+0.142
Ternary Quantization	Quant	188	−0.013
Universal Transformer	Arch	38	−0.098
State-Space Model	Arch	60	−0.156
Int8-only Quantization	Quant	258	−0.200

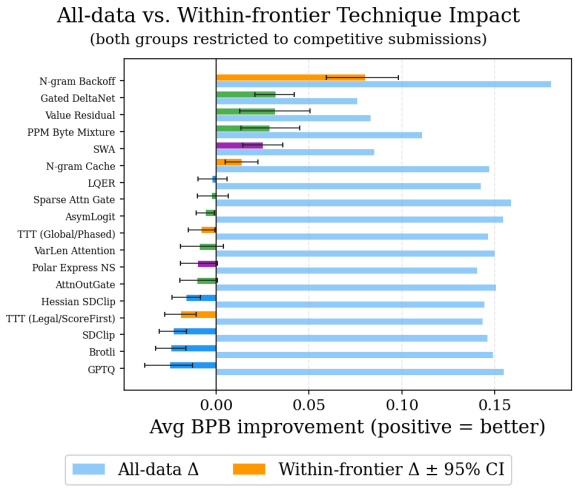


Figure 4: All-data versus within-frontier Δ_k, ordered by within-frontier value. The within-frontier bars carry 95% bootstrap confidence intervals.

early submissions.

Third, there is a clear precision sweet spot at int6. Int8-only quantization carries the most negative ΔBPB in the dataset, largely from early submissions, but mechanically it spends byte budget that int6 would turn into capacity. Low-bit formats hurt at the other end: Binary (1-bit) at -0.071 and Ternary at -0.013. Pushing precision either way gives up more than the artifact budget can recover.

Within-Frontier Impact: The Δ_k values above carry the adoption-timing bias of Section 3: a technique that became standard in strong stacks is measured against a "without" group made largely of weak submissions. To separate timing from contribution, we recompute Δ_k within the competitive frontier alone, restricting both groups to the 429 submissions scoring below 1.10 BPB. A positive value means the technique separates the best submissions from one another rather than the best era from the worst. Because submission counts span 28 to 968, we attach to each within-frontier Δ_k a 95% bootstrap confidence interval (CI) and a permutation p-value (10,000 resamples), with Benjamini–Hochberg correction (Benjamini and Hochberg, 1995) across the 70 techniques having at least five frontier submissions.

Figure 4 contrasts the two measurements. The effect is striking: almost every technique's apparent gain collapses toward zero or turns slightly negative within the frontier. GPTQ, Brotli, depth recurrence, and the two most common TTT variants, which are all near-universal among strong submissions, fall to small but significantly negative values, confirming that their large all-data Δ_k reflects era correlation rather than an independent contribution. N-gram backoff is the clear exception, retaining $\Delta_k = +0.080$ (95% CI $[0.06, 0.10]$, $p < 0.0001$) on an interval lying entirely above every other technique's. This indicates that classical n-gram blending improved even submissions that were already competitive. A second tier of techniques (PPM byte mixture, Gated DeltaNet, value residuals, and SWA) remains significantly positive with Δ_k ranging from $+0.025$ to $+0.032$. These techniques genuinely separate record-setting stacks from the merely strong field.

Technique Bundling in the Frontier: Figure 5 maps the twelve most prevalent techniques across the 429 frontier submissions by lift, the rate at which two techniques co-occur relative to that expected under independent adoption. Two groups emerge. The Phase 1 core (GPTQ, int6, EMA, and sliding window; nodes 1–3 and 7) is near-universal, each present in 54–81% of frontier stacks and all of GPTQ, int6, and EMA in 51%, yet it forms no bundle: GPTQ, the most prevalent technique, clears the 1.2 edge threshold with nothing; its closest pairing, with Brotli, falls just short. Its members co-occur simply because everyone uses them. The Phase 2–3 additions (nodes 4–6, 8, and 10–12) are the genuine bundle, adopted together well above chance, led by depth recurrence with parallel residuals (lift 1.80) and Brotli with parallel residuals (1.60). Even so, there is no fixed recipe: these twelve techniques

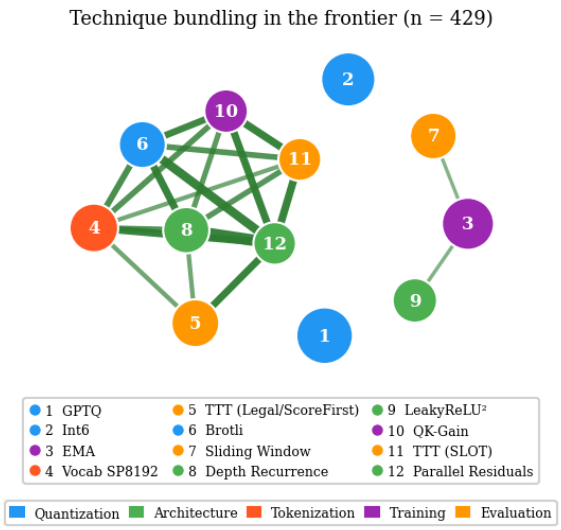


Figure 5: Technique bundling among the 429 frontier submissions (BPB < 1.10). The twelve most prevalent techniques are placed so that pairs with high lift sit close together; circle area is frontier prevalence, color is category, and the number is prevalence rank (1 = most prevalent). An edge is drawn wherever lift, the co-occurrence rate divided by the rate expected under independent adoption, is at least 1.2, with width growing with lift. Exact pairwise values are given in Appendix Figure 11.

form 222 distinct stacks across the 429 submissions, the commonest shared by only 6%, and only 22% use all six most-prevalent techniques. This variation gives the within-frontier test its power. Attribution is hardest for the universal core, whose members are too prevalent to leave a comparable "without" group, and most tractable for the bundle and periphery, which vary across stacks; the strongest independent contributor, n-gram backoff, appears in just 5% of the frontier.

7 Conclusion

We analyzed six weeks of *Parameter Golf* across 2,037 pull requests and 1,430 clean-scored submissions. The community lowered the best verified BPB from the naive baseline of 1.2244 to 1.058, a 13.6% improvement, across three phases: evaluation fixes and quantization, architectural and vocabulary exploration, and finally n-gram blending and test-time training. Much of the early improvement came from correcting a deliberately naive baseline by using sliding-window evaluation, FP16 embeddings, and improved tokenization. The headline gain, therefore, reflects cumulative community engineering rather than a single breakthrough.

The throughline is that the frontier moved less through new architectures than by stepping outside the neural model, via n-gram blending and test-time adaptation. Int6 quantization made the best use of the byte budget, since it neither wastes bits like int8 nor sacrifices accuracy as lower-bit formats do. And most reported gains shrink once remeasured among competitive submissions, leaving only a few techniques with an independent effect. More broadly, open, continuously scored competitions can function as living ablation studies, revealing which innovations truly advance the frontier and which merely benefit from timing.

Limitations

Our impact metric Δ_k is observational, not causal. It measures how strongly a technique co-occurs with low BPB and is biased by adoption timing, which is why we report the within-frontier recomputation alongside it. Even within the frontier, Δ_k is a marginal association: submissions change several techniques at once and inherit earlier stacks, so it cannot isolate a technique from the bundle it travels with. The bundling analysis in Section 6 shows where this bites: the Phase 1 core is so prevalent that its members lack a comparable "without" group, so their individual effects are only weakly identified here, whereas the periphery varies across 222 distinct stacks, which is what gives the within-frontier comparison its power. Only a controlled ablation, re-running submissions with one technique toggled, would yield causal effects. Technique detection relies on keyword matching, so it can miss or mislabel techniques that are named only in linked papers, described in non-standard terminology, or mentioned in code comments without implementation. The 61 verified entries were checked by hand, and the detectors were audited on all 1,810 PRs (Section 3.4). Scores below the verified-leaderboard tier are author-reported and only partially rerun, although those recovered from the README or PR text were additionally checked by a language-model reviewer. We also exclude submissions below 0.9 BPB as suspected validation-data leakage. These caveats limited the precision of individual Δ_k values but did not affect the qualitative ordering of the strongest and weakest techniques. Finally, all results come from a single challenge setting, so the optimal configurations are tied to its budget and would shift under tighter or looser constraints.

References

Jyrki Alakuijala, Andrea Farruggia, Paolo Ferragina, Eugene Kliuchnikov, Robert Obryk, Zoltan Szabadka, and Lode Vandevenne. 2018. Brotli: A general-purpose data compressor. *ACM Transactions on Information Systems*, 37(1).

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*. ArXiv:2112.04426.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240). ArXiv:2204.02311.

John G. Cleary and Ian H. Witten. 1984. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32(4):396–402.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*. ArXiv:1807.03819.

Ronen Eldan and Yuanzhi Li. 2023. TinyStories: How small can language models be and still speak coherent English? *arXiv preprint arXiv:2305.07759*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*. ArXiv:2210.17323.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*. ArXiv:2203.15556.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*. ArXiv:1803.05407.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *IEEE Conference on Computer Vision and Pattern Recognition*. ArXiv:1712.05877.

Keller Jordan. 2024. modded-nanogpt: NanoGPT speedrunning. `https://github.com/KellerJordan/modded-nanogpt`.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks. `https://kellerjordan.github.io/posts/muon/`.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*. ArXiv:1911.00172.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71. ArXiv:1808.06226.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*. ArXiv:2306.00978.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. In *Conference on Language Modeling (COLM)*. ArXiv:2401.17377.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit LLMs: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*.

Sharan Narang, Hyung Won Chung, Yi Tay, William Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. 2021. Do transformer modifications transfer across implementations and applications? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ArXiv:2102.11972.

OpenAI. 2026. Parameter golf. `https://github.com/openai/parameter-golf`.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*. ArXiv:2406.17557.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. YaRN: Efficient context window extension of large language models. In *International Conference on Learning Representations*. ArXiv:2309.00071.

Q Labs. 2025. NanoGPT slowrun: Language modeling with limited data, infinite compute. `https://github.com/qlabs-eng/slowrun`.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. 2024. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. In *Advances in Neural Information Processing Systems*. ArXiv:2407.08608.

Noam Shazeer. 2020. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*.

David R. So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V. Le. 2021. Primer: Searching for efficient transformers for language modeling. In *Advances in Neural Information Processing Systems*. ArXiv:2109.08668.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568. ArXiv:2104.09864.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning*. ArXiv:1909.13231.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the Conference on Computational Natural Language Learning*.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025. Gated delta networks: Improving Mamba2 with delta rule. *International Conference on Learning Representations*. ArXiv:2412.06464.

Cheng Zhang, Jianyi Cheng, George A. Constantinides, and Yiren Zhao. 2024. LQER: Low-rank quantization error reconstruction for LLMs. In *International Conference on Machine Learning*. ArXiv:2402.02446.

Zhanchao Zhou, Tianyi Wu, Zhiyun Jiang, and Zhenzhong Lan. 2024. Value residual learning for alleviating attention concentration in transformers. *arXiv preprint arXiv:2410.17897*.

A Appendix

A.1 Data Collection Pipeline

Figure 6 details the score-extraction pipeline summarized in Section 3: each of the 1,810 rows is scored by trying five sources in priority order, after which the leakage filter yields the 1,430 clean scored submissions used throughout.

A.2 Technique Impact Rankings

Figure 7 shows the full top-22 all-data ranking that Table 3 summarizes, and Figure 8 breaks the ranking down by phase, with submission counts shown in white on each bar. They expand the per-technique discussion in Section 5. Like all dataset-wide Δ_k values, these rankings carry the adoption-timing bias discussed in Section 3; the within-frontier recomputation in Section 6 isolates the genuine contributors.

A.3 Adoption Dynamics and Category Deep Dives

Figure 9 (discussed in Section 5) traces how quickly each category's signature techniques propagated through the community. Figure 10 drills into quantization, and Figures 12 and 13 into tokenization, the two categories with the most internal variation in Δ_k. Figure 11 gives the exact pairwise co-occurrence and lift values summarized by the bundling map in Figure 5.

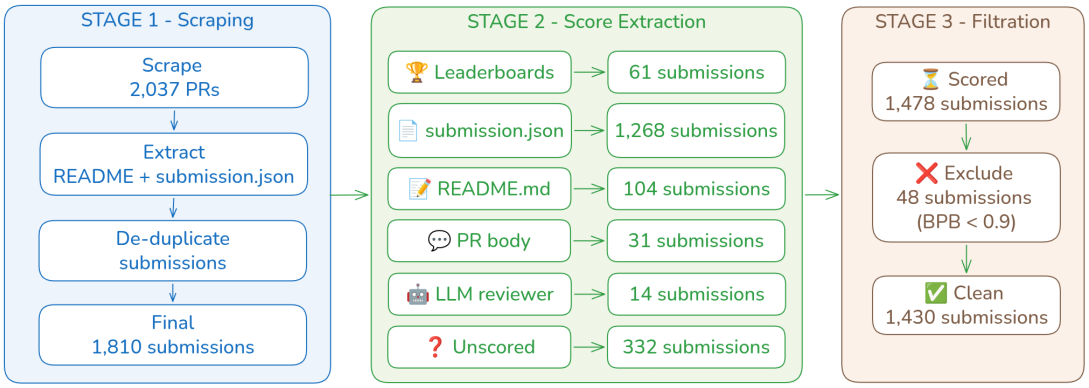


Figure 6: Data collection and score-extraction pipeline. Scores are assigned by trying five sources in order of priority: leaderboard, `submission.json`, README, PR body, and then an LLM reviewer as a last resort. The leakage filter removes 48 implausible entries (BPB < 0.9), leaving **1,430 clean scored** submissions.

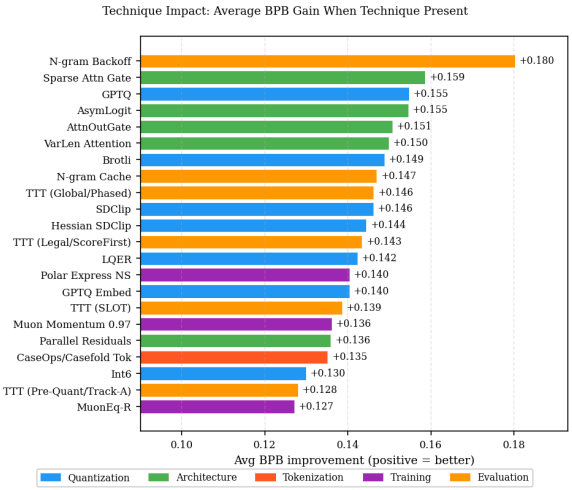


Figure 7: Top-22 techniques ranked by all-data Δ_k across the 1,430 clean scored submissions. Δ_k is observational; see Section 3.

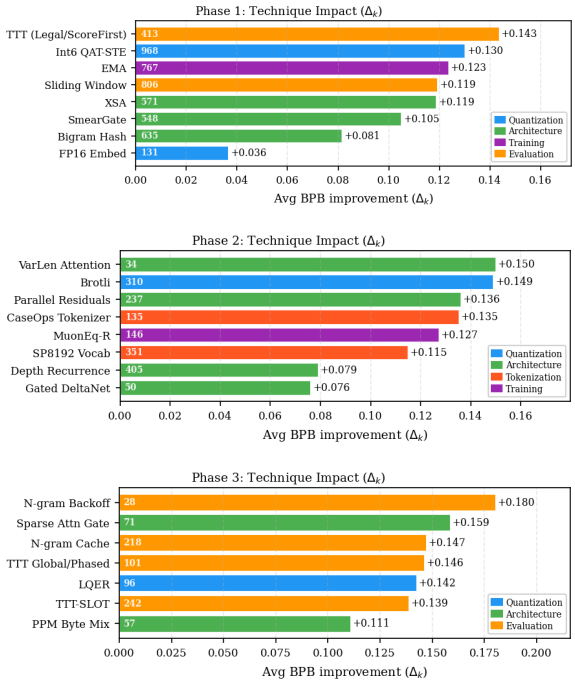


Figure 8: Per-phase technique-impact rankings (Δ_k) across all scored submissions, broken out by phase (top to bottom: Phases 1–3). White bar counts show the number of submissions using each technique.

A.4 Full Technique Flag List

The 84 binary technique flags are organized into five categories.

1. **Quantization** (17 flags): the precision of weight representation and the compression format of the weight file. Covers int8, int6, int5, FP16 embeddings, QAT-STE, GPTQ (Frantar et al., 2023), GPTQ Embeddings, Ternary, Binary, Mixed-precision, zstd, Brotli, SDClip, Hessian SDClip, LQER, AWQ (Lin et al., 2024), and Crown-Q.

2. **Architecture** (37 flags): structural modifications to the transformer or non-transformer model. Covers depth recurrence, parallel residuals, XSA, SmearGate, Bigram Hash, U-Net skip, Partial RoPE (Su et al., 2024), YaRN RoPE (Peng et al., 2024), NTK-Aware RoPE, LeakyReLU^2, SwiGLU (Shazeer, 2020), Gated DeltaNet (Yang et al., 2025), State-Space Model (Gu and Dao, 2023), Universal Transformer, VarLen Attention, Flash Attention 3 (Shah et al., 2024), Fused MLP, Fused QKV Projection, Cross-Layer Attention, KV Sharing, AttnOutGate, Paged KV Cache, Gated Attention, Sparse Attention Gate, Logit Softcap, Polynomial Softcap, AsymLogit, Mixture of Softmax, MoE MLP,

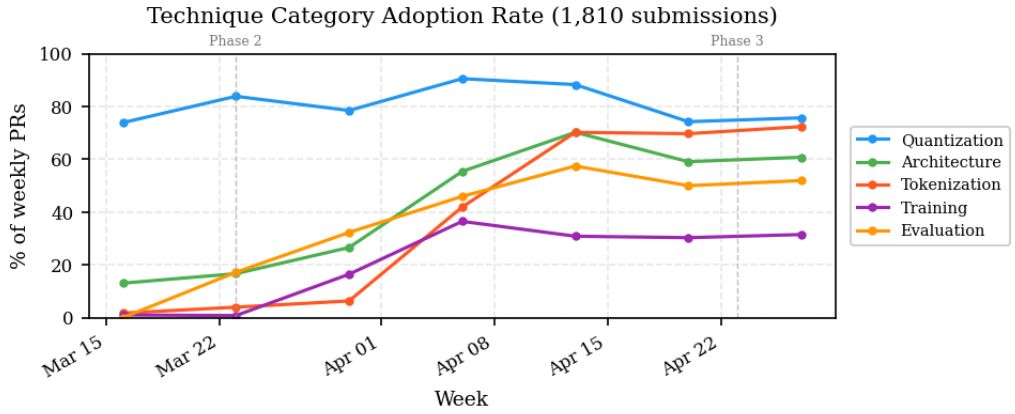


Figure 9: Weekly adoption rate per technique category. Each point is the fraction of that week's PRs using at least one signature technique from the category (counting all 84 flags saturates, since nearly every PR uses one technique per category). Dashed lines mark phase transitions.

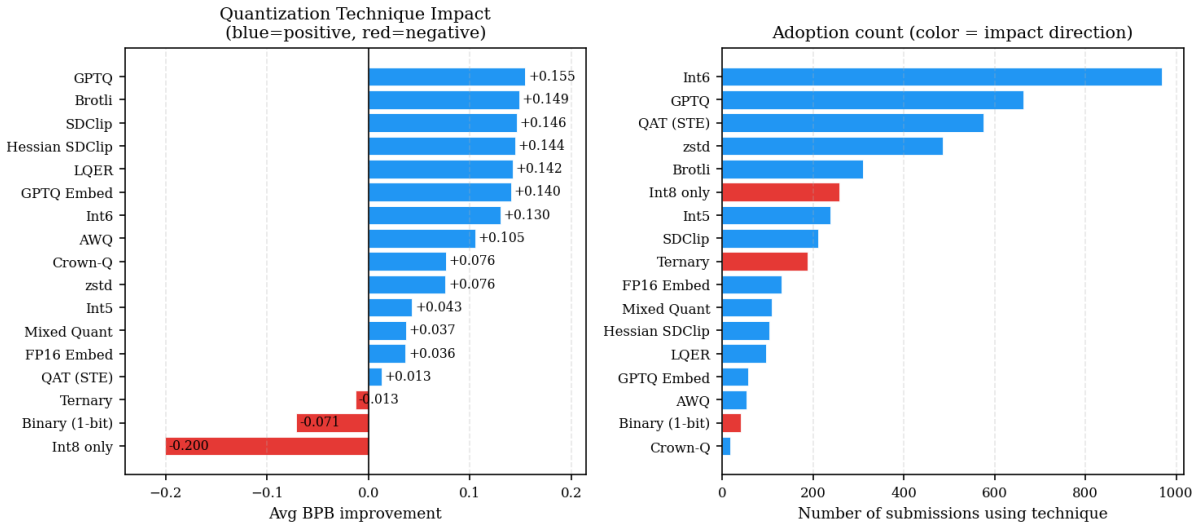


Figure 10: Quantization technique deep dive. *Left*: average BPB improvement Δ_k for all 17 quantization flags; blue bars are beneficial, red are harmful. *Right*: adoption counts colored by impact direction. Int8-only and Binary (1-bit) correlate with worse BPB despite wide adoption, while GPTQ, Brotli, and SDClip show the strongest gains.

Stochastic Depth, Selective Pruning, Skip Gates, Tied Embeddings, Factored Tied Embedding, Value Embeddings, Value Residual, and PPM Byte Mixture.

3. **Tokenization** (4 flags): vocabulary design and preprocessing choices. Covers SP1024, SP4096, SP8192, and CaseOps/Casefold.

4. **Training** (16 flags): optimizer and regularization choices. Covers Muon WD (Jordan et al., 2024), SWA, EMA, MuonEq-R, Parallel Muon, NeoMuon, OrthoInit, Overtone Init, QK-Gain, Lower LR, Banked Muon, Muon Momentum 0.97, Polar Express NS, Warmdown, Z-Loss, and Complementary Training.

5. **Evaluation** (10 flags): strategies applied at inference or evaluation time inside the 16 MB artifact. Covers Sliding Window, the six TTT variants (LoRA, Legal/ScoreFirst, Pre-Quant/Track-A, Global/Phased, SLOT, and Other), N-gram Backoff, N-gram Cache, and Neural Cache.

Techniques referenced in the main text but not described in Section 5.

- **Attention Output Gate** and **AsymLogit** are challenge-community techniques: the former applies a learned gate to the attention sublayer's output before it enters the residual stream; the latter is an asymmetric variant of

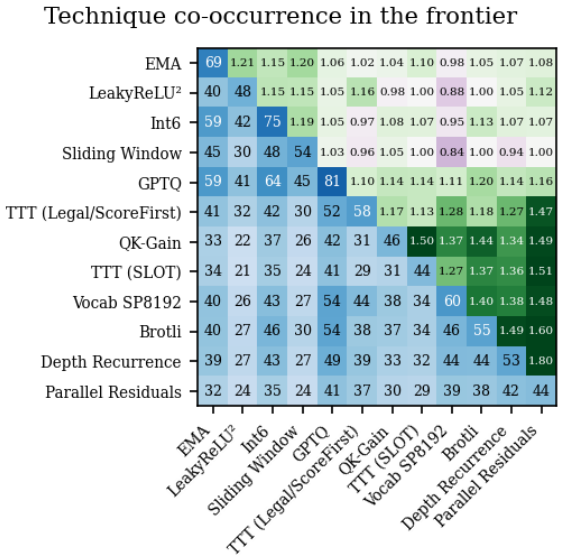


Figure 11: Pairwise co-occurrence of the twelve most prevalent frontier techniques, cluster-ordered so that bundled techniques are adjacent. Lower triangle: percentage of the 429 frontier submissions using both techniques; diagonal: frontier prevalence. Upper triangle: lift (green, > 1: adopted together more than chance; purple, < 1: less). This is the exact-number companion to Figure 5.

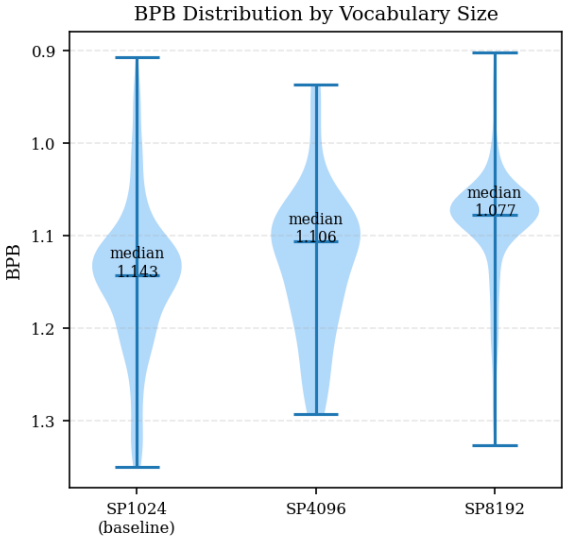


Figure 12: BPB distributions for the three SentencePiece vocabulary sizes; larger vocabularies shift the median downward (SP8192 median 1.077 vs. SP1024 baseline 1.143).

output-logit soft-capping that bounds positive and negative logits differently.

- **Ternary and Binary quantization** restrict weights to $\{-1, 0, +1\}$ or $\{-1, +1\}$, as in BitNet (Ma et al., 2024).
- **Universal Transformer** (Dehghani et al., 2019) applies one shared layer recurrently over depth in place of a stack of distinct layers.
- **State-Space Model** replaces attention with a selective linear state-space recurrence, as in Mamba (Gu and Dao, 2023).
- **Value residuals** (Zhou et al., 2024) add the first layer's value vectors into every later layer's values, preserving early-layer information.
- **SWA** (stochastic weight averaging) (Izmailov et al., 2018) averages weights sampled along the training trajectory.

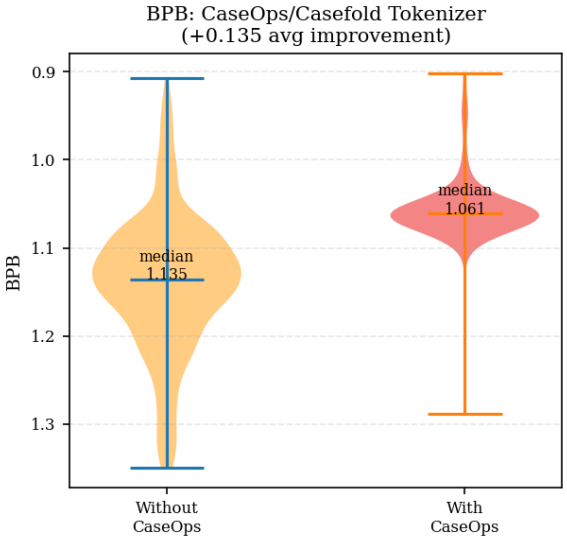


Figure 13: CaseOps/Casefold tokenizer impact: it tightens the upper tail and shifts the median from 1.135 to 1.061, an average $\Delta_k = +0.135$.

Prosodic Structure as a Critical Component in Phonemic Language Modelling at BabyLM Scale

Bill K. S. Ku, Janet B. Pierrehumbert
Department of Engineering Science
University of Oxford

Abstract

Human children learn language from continuous, prosodically structured speech long before they can read, yet almost all language models train on orthographic text. Building on previous phonemic BabyLM training (Goriely et al., 2024), we ask whether supplying an estimate of the prosody infants exploit improves phonemic language modelling at BabyLM scale. We compare an orthographic baseline to three conditions forming a ladder of phonological richness. All conditions are trained as BERT MLMs at 10M and 100M words. Addressing the ecological validity of existing metrics in the small-data regime, we build a lexical decision task and two semantic organisation tasks.[1] Our new lexical decision task offers improved insights into the distinction between wordlikeness, token familiarity, and lexicality as factors. Triphone phonotactic control and pseudo-log-likelihood scoring shows that these scores reflect phonotactic familiarity rather than wordhood: every model sits near or below chance on an adversarial inverted condition, revealing prior results of this kind as partly artefactual. On our new semantic tasks, a prosodically annotated phoneme stream gives the strongest taxonomic organisation at both the 10M and 100M word scales. Phonemic models with boundary structure are competitive with the orthographic baseline. These findings suggest that prosodic structure is a key component of phonemic language modelling in the small-data regime most analogous to infant learning.

1 Introduction

Human children reach near-adult linguistic competence from a few tens of millions of words of input, whereas state-of-the-art language models require several orders of magnitude more data (Warstadt et al., 2023; Fridman and LeCun, 2024). The BabyLM Challenge (Warstadt et al., 2023) reframes this gap as a question of sample efficiency, capping training at developmentally plausible budgets. A second and equally basic difference concerns the *modality* of the signal. Children understand and produce language years before they learn to read: they acquire language from a continuous stream of speech in which word boundaries are not marked and must be inferred. Almost all language models, and most BabyLM submissions, instead train on orthographic text, in which word boundaries are marked with whitespace.

Working in *phonemic* space is a natural response (Goriely et al., 2024). However, a plain phoneme stream discards prosody, a source of information that infants are known to use to bootstrap word segmentation and syntactic structure from the earliest stages of language acquisition. Here, we ask whether providing an estimate of the prosodic structure can enhance word learning from phonemic input at BabyLM scale. We compare an orthographic baseline (ORTHO) to three conditions of increasing phonological richness: a plain continuous-phoneme reconstruction of Goriely et al. (2024) (P-GORIELY); a condition that inserts intonation break tokens at major syntactic boundaries, as estimated from a dependency parse (P-BREAKS); and one that further adds sentence-type boundary tones from a neural annotator trained on prosodically labelled speech (PROSODY-NN). Each condition is trained as a BERT masked language model (Devlin et al., 2019) at two corpus scales, 10M and 100M words, yielding eight models.

Standard benchmarks use orthographic input. While conventions about the use of punctuation and whitespace in orthography reflect some prosodic features, such as the ends of utterances, they also reflect norms about which sequences are meaningful words. Some norms, such as the spacing around function words and elements of compounds, are

[1] The lexical decision and semantic evaluation sets will be distributed through https://github.com/billkuai/prosodic_language_modelling_babylm after the BabyLM Workshop at EMNLP 2026, in late Oct 2026.

Proceedings of the Second BabyLM Workshop, pages 148–164
October 29, 2026 ©2026 Association for Computational Linguistics

arbitrary and language-specific – clearly going beyond what human infants learn on a 10M budget, and in some respects beyond what children have learned after 100M words of language exposure. A focus on learning from phonemic input thus aligns with the goals of the BabyLM project, in building models that draw inspiration from the efficiency of language learning by children.

Focusing on learning of the mental lexicon, we propose and evaluate three new tasks over child-relevant vocabulary. A phonotactically controlled lexical decision task scored by pseudo-log-likelihood is similar to the BabySLM task in Lavechin et al. (2022). However, we redesign this task to make a stronger distinction between wordlikeness (having the phonological form of a word) and lexicality (having a meaning associated with the form). Our semantic tasks, concept boundary discrimination and unsupervised category induction, are grounded in research on language development.

Our contributions are threefold. In our redesign of the lexical decision task, we aim to support strong comparisons amongst wordlikeness, token familiarity, and lexicality. Pseudoword candidates are generated with a trigram Markov chain and scored in the phonemic domain. A filter removes all candidates that are real words in either orthographic or phonemic form. Then, pseudowords are individually selected to best match the phonotactic scores of the real words in the test. We also introduce an adversarial *Inverted* test condition, in which each pseudoword has a higher phonotactic score than its real word competitor. The results show that pseudo-log-likelihood scoring probes token familiarity more than lexical knowledge, and that previously reported phonemic-model results on the lexical decision task are partly artefactual. Second, we show that providing estimated prosodic markers makes phonemic models competitive with, or better than, the orthographic baseline on age-relevant semantic organisation. Taken together, these results indicate that, since the infant's input is speech rather than text, careful phonemic representation enriched with prosody is a key component in developmentally plausible language modelling.

2 Background and Related Work

Acquisition from continuous speech. Children reach near-adult linguistic competence from roughly 100 million words of input by early adolescence (Warstadt et al., 2023). Before they learn to read, their input is speech: a continuous acoustic stream with no whitespace, no punctuation, and no explicit word boundaries. To build their mental lexicon, they segment words from fluent speech using distributional regularities (Saffran et al., 1996), and exploit prosodic structure to do so. The prosodic bootstrapping hypothesis (Christophe et al., 2008; Jusczyk, 1997) holds that phrase boundaries and boundary tones provide a partial parse of the signal before any grammar is in place. In addition, they carry information about some aspects of meaning. English yes/no questions typically carry a high (rising) boundary tone while declaratives and wh-questions carry a low (falling) tone, a contrast that child-directed speech exaggerates (Soderstrom et al., 2003). Diphone statistics in combination with utterance boundaries are already rich enough to strongly support segmentation (Daland and Pierrehumbert, 2011). These considerations motivate modelling language from a phoneme stream rather than from text.

Phonemic language modelling and BabyLM. Most BabyLM submissions train on orthographic text. Goriely et al. (2024) is a notable exception. They convert the BabyLM corpus to IPA with eSpeak-NG (Duddington and Dunn, 2025), remove word separators to form a continuous stream of phonemes, and train a GPT-2 model, finding phonemic models to be fairly competitive with orthographic baselines. We extend this framework in three ways: we use the masked language model BERT throughout (Devlin et al., 2019). Its bidirectional objective is more data-efficient for the short utterances that are typical in the training set. We make prosodic structure a primary experimental variable rather than discarding it; and we redesign the lexical decision task to remove a confound in prior evaluations.

Evaluating the mental lexicon. A word is an associative pair of a phonological form and a meaning (Needle et al., 2022). So, a lexical decision task should test whether a form is meaningful, not merely whether it is phonotactically word-like. Standard benchmarks serve this end imperfectly. The speech-oriented BabySLM (Lavechin et al., 2022) uses the pipeline in Keuleers and Brysbaert (2010) and Nguyen et al. (2020) to design pseudowords for the forced-choice lexical decision task reported in Goriely et al. (2024). This pipeline does not control for triphone phonotactics, which are known to play an important role in wordlike-

ness judgments (Bailey and Hahn, 2001; Hay et al., 2004; Needle et al., 2022).

We control for this factor, which intrinsically encompasses some aspects of syllable structure (though not all the aspects of syllable structure that BabySLM controls for). Overall, real words on average are expected to be phonotactically better than nonwords, because the encoding, memory, and production of lexical items is biased towards high phonotactic likelihood (Hay et al., 2004). Through repeated learning and use, forms that include less likely sequences tend to be regularized towards more typical forms. A matched-foil design that does not control for triphone likelihood thus retains a systematic asymmetry. A post-hoc analysis of the BabySLM test set shows that the real word has better triphone phonotactics than the matched pseudoword in 70.3% of the test pairs (see Appendix C). Thus, a model could score above chance simply by preferring the phonotactically better-formed item, without storing form-meaning associations. The key distinction between *wordlikeness* and *lexicality* is probed in our Inverted category (Section 4.1).

The GLUE (Wang et al., 2018) and BLiMP (Warstadt et al., 2020) tasks are used in BabyLM papers to probe the learnability of syntax and semantics. After remapping the materials to a phonemic representation, Goriely et al. (2024) reports accuracies in the range of 64% to 78% on BLiMP and GLUE for their various conditions and LMs, trained on 100M words. It is difficult to disentangle how much the errors result from weaknesses in the LMs versus intrinsic limitations of the training set size. Some patterns probed in these datasets are obviously not learned by humans from 10M words. For example, the word *any* is learned only at age 4 or 5 (Tieu and Lidz, 2016), reflecting some 40M to 50M words of exposure. For a training budget of 64M tokens Warstadt et al. (2020) report accuracies of about 50% for words like *any*, and about 40% for syntactic island constraints. Given this situation, we judged that the development of new simple semantic tasks with high face validity would best further our aim of evaluating how well prosodic information can support learning of lexicality.

3 Experimental Conditions

The orthographic text condition is included as a baseline. The phonological conditions form a ladder of increasing phonological richness, from the continuous phoneme stream as defined in Goriely et al. (2024) to a phoneme stream annotated with phrase boundaries and sentence-type boundary tones (Table 1). All conditions are produced by a single preprocessing pipeline parameterised per condition, so only the input representation differs across them. Every condition is trained identically at both the 10M and 100M scales.

Condition	Added structure
ORTHO	orthographic text (baseline)
P-GORIELY	continuous IPA; `[UTT]` at utterance ends
P-BREAKS	+ `[B2]`, `[B4]` phrase boundaries
PROSODY-NN	+ `[H%]`, `[L%]` boundary tones

Table 1: The four conditions, organised as a baseline plus three phonemic conditions of increasing richness. The phonemic conditions carry no whitespace.

3.1 Corpus

All conditions are trained on the BabyLM 2025 corpus (Charpentier et al., 2025), which draws on six English sources (CHILDES, Project Gutenberg, OpenSubtitles, Simple Wikipedia, the spoken BNC, and Switchboard). Roughly half of the tokens are transcripts of conversational or child-directed speech, but the prosodic features were not transcribed, so our study estimates them (see Section 3.3). The full composition is given in Appendix B.

3.2 The Four Conditions

ORTHO is lowercased orthographic text and serves as the baseline. P-GORIELY reconstructs the approach of Goriely et al. (2024): the corpus is transcribed to a continuous IPA stream with word separators and punctuation removed, leaving only phonemes with an utterance-boundary token `[UTT]`. P-BREAKS adds two levels of the ToBI break-index hierarchy (Beckman and Ayers Elam, 1997), an intermediate-phrase token `[B2]` and an intonation-phrase token `[B4]`, derived from a syntactic parse rather than punctuation; it is the first condition in which a structural token can appear in a sentence with no surface punctuation. PROSODY-NN keeps these break tokens and additionally marks sentence-final intonation with a high (rising) boundary tone `[H%]` or a low (falling) tone `[L%]`, assigned by a neural annotator. Table 2 shows all four conditions on three example sentences chosen to expose the two thresholds: S3 triggers a `[B2]` at a subordinate-clause onset with no comma, and S2 receives `[H%]` as a yes/no question.

Condition	Rendered output
S1: "The dog sat on the mat."	
Ortho	the dog sat on the mat .
P-Goriely	ðədɒgsætɒnðəmæt [UTT]
P-breaks	ðədɒgsætɒnðəmæt [B4]
Prosody-NN	ðədɒgsætɒnðəmæt [B4] [L%]
S2: "Did she eat?"	
Ortho	did she eat ?
P-Goriely	dɪdʃiːiːt [UTT]
P-breaks	dɪdʃiːiːt [B4]
Prosody-NN	dɪdʃiːiːt [B4] [H%]
S3: "He left because it was late."	
Ortho	he left because it was late .
P-Goriely	hiːlɛftbɪkɒzɪtwɒzleɪt [UTT]
P-breaks	hiːlɛft [B2] bɪkɒzɪtwɒzleɪt [B4]
Prosody-NN	hiːlɛft [B2] bɪkɒzɪtwɒzleɪt [B4] [L%]

Table 2: Three example sentences rendered under each condition. S3 shows a [B2] inserted at a subordinate-clause onset with no comma; S2 shows the [H%]/[L%] sentence-type distinction.

3.3 Phonemic Transcription and Prosodic Annotation

The phonemic conditions are transcribed with eSpeak-NG (Duddington and Dunn, 2025) through the `phonemizer` library (Bernard and Titeux, 2021), with the word separator set to empty so the output is a continuous IPA stream with no explicit word boundaries. Prosodic tokens are inserted by a two-pass scheme that phonemises each word separately and interleaves boundary tokens at the recorded positions. In P-breaks, [B2] and [B4] are derived from a spaCy dependency parse (Honnibal et al., 2020); in Prosody-NN, the boundaries and sentence-final tones ([H%]/[L%]) are predicted by a `bert-base` sequence labeller trained on the ToBI-annotated Boston University Radio News Corpus (Ostendorf et al., 1995), with the tone head gated on held-out F_1 and a rule-based fallback. (For the annotator and gating thresholds, see Appendix B.)

3.4 Tokenisation

A Byte-Pair Encoding tokeniser (Sennrich et al., 2016) is trained from scratch for each condition, on its preprocessed corpus; the character inventory and merge statistics of a prosodically annotated IPA stream differ substantially from orthographic English. Vocabulary size is 4,000 at 10M and 8,000 at 100M. For the roughly 50-symbol IPA alphabet, BPE yields primarily diphone and triphone units. All structural tokens ([B2], [B4], [H%], [L%], [UTT], and the reserved BERT tokens) are filtered from the corpus before BPE training and inserted at fixed vocabulary ids afterwards. This prevents the algorithm from learning spurious merges inside token strings. Annotating the stream lengthens it: The annotated conditions carry roughly 10% more tokens than P-Goriely, though every phonemic condition still consumes fewer tokens than Ortho. Appendix G reports the token counts and Appendix F the segmentation behaviour.

3.5 Model Architecture and Training

A BERT-family masked language model (Devlin et al., 2019) is trained from scratch for each condition. BERT was selected due to its data-efficiency at this scale. The model size is matched to corpus scale: a BERT-small configuration (≈7.7M parameters) at 10M and a BERT-base configuration (≈92M parameters) at 100M. All conditions share the same training recipe; full architecture and training hyperparameters are given in Appendix A.

4 Evaluation

We evaluate all eight models on three purpose-built tasks, each defined in the same representational space as training and restricted to infant-relevant vocabulary: a phonotactically controlled lexical decision task (Section 4.1) and two semantic tasks: concept boundary discrimination (Section 4.2.1) and unsupervised category induction (Section 4.2.2). All tasks read from the frozen trained models without any fine-tuning.

4.1 Lexical Decision

The lexical decision task asks whether a model prefers a real word to a phonotactically matched pseudoword (except for *Inverted* category pairs). Because the two items in a pair are matched on phonotactic plausibility (or, by deliberate design, favours the pseudoword for the *Inverted* category) under the model's own training distribution, above-chance accuracy is evidence of stored lexical knowledge rather than phonotactic filtering alone.

Phonotactic scoring. Each condition has its own trigram (n=3) phonotactic model, estimated from that model's training dictionary. Following Needle et al. (2022), counts are type-based: each word type contributes once regardless of corpus frequency, so the model captures phonotactic *licitness* rather than lexical *frequency*. The phonotactic score of a word w with phone sequence $\phi_1 \dots \phi_L$ is the log-probability of its padded form

$\Phi(w) = \langle s\rangle\,\langle s\rangle\,\phi_1 \ldots \phi_L\,\langle /s\rangle$ under a Witten-Bell smoothed model (Witten and Bell, 1991):

$$S(w) = \sum_{i=3}^{L+3} \log P_{\text{WB}}(\Phi_i \mid \Phi_{i-2}, \Phi_{i-1})\,. \quad (1)$$

Witten-Bell smoothing interpolates each context with its lower-order backoff so that unseen trigrams in the test set still receive probability mass. Writing h for a context, $c(h,\phi)$ for the count of ϕ following h, $c(h) = \sum_{\phi'} c(h,\phi')$, and $t(h) = |\{\phi' : c(h,\phi') > 0\}|$ for the number of distinct continuations of h,

$$P_{\text{WB}}(\phi \mid h) = \begin{cases} \frac{c(h,\phi)}{c(h)+t(h)} & c(h,\phi) > 0, \\ \lambda(h)\, P_{\text{WB}}(\phi \mid h') & c(h,\phi) = 0, \end{cases} \quad (2)$$

where h' drops the leftmost symbol of h and the recursion bottoms out at a uniform unigram floor. The backoff weight $\lambda(h) = t(h)/\big(c(h) + t(h)\big)$ grows with the diversity of observed continuations $t(h)$, allocating more probability to novel phones in contexts that have shown lexical variety. For Ortho the scorer operates on characters.

Pseudoword generation. For practical reasons, pseudowords are first generated in the orthographic space. Then they are mapped into the phonemic space using eSpeak-NG, and scored using the triphone Markov model described above. Pairing is carried out once, in the phonemic space of P-Goriely, to ensure every condition is tested on identical items. This means that the pseudowords are created to match the real-word phonotactic distribution by construction and lets us control their score directly, which the Inverted category requires. Given a context ctx $= (\phi_{i-2}, \phi_{i-1})$, the next character is sampled with a temperature T:

$$P_T(\phi_i \mid \text{ctx}) \;\propto\; c(\text{ctx}, \phi_i)^{1/T}, \quad (3)$$

generating autoregressively from $\langle s\rangle\langle s\rangle$ until $\langle /s\rangle$ is emitted. Each candidate is then phonemised with eSpeak-NG and scored with the model above, and it is that phonemic score, not the proposal, which decides what enters the test set. Lowering the temperature to T=0.3 concentrates proposals on frequent transitions, yielding a pool dense enough in high-scoring forms to supply the Inverted category. The full procedure is given in Appendix C.

Items are pairs of a real word and a pseudoword. Two filters guarantee that the items are valid. *Filter A* admits a real word only if it is a genuine English word, that is, attested in a reference word list after lemmatisation; this removes corpus artefacts, such as *https* or repeated-character strings that survive a purely alphabetic filter and would otherwise be labelled real words. *Filter B* admits a pseudoword only if neither its orthographic form nor its eSpeak IPA form (raw, and with stress and length marks stripped) occurs in the training set, so that no pseudoword is in fact a word the model has seen, whether spelled differently from a real homophone or transcribed identically to one. Rejected candidates are resampled so category sizes are held fixed. This phonemic screen is essential: without it, a large share of candidate pseudowords coincide with real words in IPA space and are wrongly counted as model errors. Contamination statistics are reported in Appendix D. For the Ortho condition, the real word is paired with the original orthographic form of the pseudoword, from which the phonemic form was generated. (Using the same test items across all conditions means that the orthotactic likelihood is not fully controlled).

Pair categories. Real words fall into three categories. *LexicalMemory* words are a single token in the condition's BPE vocabulary, so the model can score the whole word in one step. *Composition* words span multiple tokens, requiring the model to compose a representation from sub-word units. *Inverted* pairs are adversarial: the real word is drawn from the bottom of the phonotactic-score distribution and the pseudoword from the top, so that $S(w_{\text{fake}}) > S(w_{\text{real}})$ by construction. A model relying only on phonotactics will therefore prefer the pseudoword and score at or below chance on Inverted; only stored lexical knowledge that overrides phonotactic expectation can raise accuracy above chance. Standard, non-inverted pairs are matched to within 2.0 log-probability units. The evaluation set is built once on orthographic words and derived for the phonemic conditions by phonemising the same pairs, so every model is tested on identical word content; it contains 4,129 pairs, of which 2,000 are Inverted, with the split between LexicalMemory and Composition depending on each condition's vocabulary.

Metric. Each model is scored by pseudo-log-likelihood (PLL; Salazar et al., 2020), the natural scoring rule for a masked language model. For a

word tokenised as $\mathbf{w} = (w_1, \ldots, w_N)$,

$$\mathrm{PLL}(\mathbf{w}) = \sum_{i=1}^{N} \log P_\theta\big(w_i \mid \mathbf{w}_{\setminus i}\big)\,, \qquad (4)$$

the sum of masked-token log-probabilities across the word. A pair is judged correct when $\mathrm{PLL}(w_{\text{real}}) > \mathrm{PLL}(w_{\text{fake}})$. Accuracy is reported separately for each category; chance is 0.50.

4.2 Semantic Tasks

Both semantic tasks operate on contextualised embeddings of 68 early-vocabulary words drawn from the MacArthur-Bates Communicative Development Inventories (Fenson et al., 1993), spanning six categories (animals, food, body parts, clothing, vehicles, and household objects); the full list is given in Appendix E. For each target word we collect up to 30 short excerpts where it occurs from the CHILDES set (MacWhinney, 2000). render each excerpt into the surface form for the condition, with the frozen model. We use the *target-anchored* representation: the per-excerpt vector is the mean of the final-layer hidden states at the subword positions of the target word, and the word vector is the mean over all excerpts. Writing T_e for the target's subword positions in excerpt $e \in E_w$ with hidden states $\mathbf{h}_t^{(e)}$,

$$\mathbf{e}_w = \frac{1}{|E_w|} \sum_{e \in E_w} \frac{1}{|T_e|} \sum_{t \in T_e} \mathbf{h}_t^{(e)}. \qquad (5)$$

Anchoring on the target word, rather than mean-pooling the whole excerpt, isolates the word's representation from the sentence topic and is what makes the measure word-specific.

4.2.1 Concept boundary discrimination (CBD)

Concept boundary discrimination (CBD) is a 3-way ranking task. For each target word, a probe presents three alternatives: a *same-category* word, a *thematic foil* from a different category but linked to the target by a non-taxonomic relation, and an *unrelated foil*. Two words are defined as thematically related when they stand in a script or event association, a production or source relation, a containment or part-whole relation, a spatial co-occurrence, or a functional relation, and not through shared category membership (Lin and Murphy, 2001; Imai et al., 1994). Each of the 68 targets is assigned a single thematic foil by a hand-curated lookup; because the foil must itself be a CDI word to have CHILDES excerpts, roughly 30 targets yield a valid foil, producing 300 probes at ten draws per word. Table 3 gives representative foil pairs across the five relation families.

Target	Thematic foil	Relation	CDI
Script / event			
cat	milk	typical food	✓
fish	water	habitat	✓
dog	bone	typical food	†
Production / source			
cow	milk	product	✓
egg	bird	source	✓
Containment / part-whole			
milk	bottle	container	✓
shoe	foot	worn on	✓
pocket	hand	contains	✓
Spatial / locational			
boat	water	location	✓
car	road	location	†
Functional / instrumental			
eye	look	function	†
cheese	mouse	typical eater	✓

Table 3: Representative thematic foil pairs by relation family. Each foil is a different-category word tied to the target by a non-taxonomic relation and, to have CHILDES excerpts, must itself be a CDI target. Entries marked † have no CDI-resident foil and contribute no probes.

The model output is the cosine similarity of each of the three alternatives to the probe word. The measure of interest is the ranking of these three alternatives. The cosine similarity is defined by the following equation (with maximum similarity having the value 1.0).

$$s_{j \in \{\text{same, them, unrel}\}} = \cos\big(\mathbf{e}_{\text{tgt}}, \mathbf{e}_j\big)\,, \qquad (6)$$

We report three accuracies: three-way (the same-category word preferred over both foils, chance $\frac{1}{3}$), taxonomic-versus-thematic, and taxonomic-versus-unrelated (the same-category word preferred over the thematic or unrelated foil respectively, chance $\frac{1}{2}$). Taxonomic-versus-thematic is the primary metric: it isolates taxonomic organisation from thematic association, a distinction distributional models often fail to make. A category neighbour (e.g. *cat* for *dog*) should be preferred to a merely thematic associate (e.g. *bone*). Writing s_{same}, s_{them} and s_{unrel} for the cosine similarity between $\mathbf{e}_{\text{tgt}}$ and the same-category word, the thematic foil and the unrelated foil respectively, a probe is scored correct

when:

$$\begin{aligned} \text{3-way:} \quad & s_{\text{same}} > \max(s_{\text{them}},\, s_{\text{unrel}})\,, \\ \text{Tv-them:} \quad & s_{\text{same}} > s_{\text{them}}, \\ \text{Tv-unrel:} \quad & s_{\text{same}} > s_{\text{unrel}}. \end{aligned} \tag{7}$$

We also report the mean taxonomic advantage: the average margin by which the same-category similarity exceeds the thematic-foil similarity,

$$\text{adv} = \frac{1}{N}\sum_{n=1}^{N}\Big[\cos\big(\mathbf{e}_{\text{tgt}}, \mathbf{e}_{\text{same}}\big) - \cos\big(\mathbf{e}_{\text{tgt}}, \mathbf{e}_{\text{them}}\big)\Big]. \tag{8}$$

4.2.2 Semantic Category Induction (SCI)

Semantic category induction (SCI) tests whether word vectors, clustered without supervision, recover the six ground-truth categories. The aggregated vectors $\mathbf{e}_w$ of all 68 words are clustered into $K = 6$ groups with both K-means (ten random initialisations) and agglomerative clustering under Ward's linkage (Ward, 1963). Cluster quality is measured against the category labels by the Adjusted Rand Index (ARI; Hubert and Arabie, 1985) and Normalised Mutual Information (NMI), with K-means ARI as the primary metric. The ARI corrects the Rand Index (RI) for chance agreement,

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{\max(\text{RI}) - \mathbb{E}[\text{RI}]}, \tag{9}$$

so that ARI $= 0$ corresponds to random labelling and ARI $= 1$ to perfect categorisation. SCI is evaluated on the words observed in training. It tests only how the model has organised the vocabulary it has actually seen.

5 Results

We report results for the three tasks, at both the 10M and 100M scales. All figures are read from the frozen models.

5.1 Lexical Decision

Table 4 reports PLL accuracy by pair category. On LexicalMemory, every condition is near ceiling (0.930 to 0.991). For Composition, the results range from 0.768 to 0.819; the phonemic conditions perform similarly to Ortho, with P-Goriely highest at 10M and Ortho highest at 100M. On the Inverted category, performance for every condition is poor. Values sit close to the 0.50 chance level at 10M (0.479 to 0.506); at 100M Ortho remains at chance (0.496) while the three phonemic conditions fall below it, to between 0.396 and 0.444.

Condition	LexMem	Comp	Inv.
10M			
Ortho	**0.991**	0.768	0.499
P-Goriely	0.942	**0.819**	**0.506**
P-breaks	0.948	0.799	0.489
Prosody-NN	0.930	0.800	0.479
100M			
Ortho	0.980	**0.813**	**0.496**
P-Goriely	**0.988**	0.801	0.396
P-breaks	0.982	0.808	0.444
Prosody-NN	0.979	0.807	0.421

Table 4: PLL accuracy on the lexical decision task ($n = 4{,}129$ pairs) by pair category. Chance is 0.50. Bold marks the best condition in each column within a scale.

Condition	3-way	Tv-them	Tv-unrel	Adv
10M				
Ortho	0.463	0.537	**0.693**	+0.019
P-Goriely	0.420	0.560	0.613	+0.028
P-breaks	0.400	0.490	0.600	+0.009
Prosody-NN	**0.483**	**0.607**	0.670	**+0.040**
100M				
Ortho	**0.533**	0.570	**0.860**	+0.019
P-Goriely	0.477	0.583	0.737	+0.022
P-breaks	0.443	0.540	0.703	+0.013
Prosody-NN	**0.533**	**0.610**	0.747	**+0.027**

Table 5: CBD accuracy ($n = 300$ probes; SE ≈ 0.028 on Tv-them). *Tv-them*: taxonomic vs. thematic (primary); *Tv-unrel*: taxonomic vs. unrelated; *Adv*: mean taxonomic advantage. Chance is $\frac{1}{3}$ for 3-way and 0.50 for the pairwise columns. Bold marks the best condition in each column within a scale.

5.2 Semantic Tasks

Table 5 reports Concept Boundary Discrimination accuracy over 300 probes. On the primary taxonomic-versus-thematic metric, Prosody-NN is highest at both scales. and no other condition exceeds 0.6; Prosody-NN also has the largest mean taxonomic advantage at both scales Ortho leads the easier taxonomic-versus-unrelated comparison at both scales P-breaks is lowest on taxonomic-versus-thematic at 10M, at 0.490. Figure 1 plots the taxonomic-versus-thematic accuracy across scales.

For Semantic Category Induction, Table 6 reports cluster recovery over the 68 words. At 10M, Ortho leads on all three measures (K-ARI 0.483), with the phonemic conditions between 0.279 and 0.303. At 100M the ordering changes: P-breaks (K-ARI 0.637) and Prosody-NN (0.617) exceed Ortho (0.576) on both k-means measures, while Ortho leads on agglomerative ARI (0.717). Figure 2 shows the 10M-to-100M trajectories, in which P-breaks and Prosody-NN cross above Ortho.

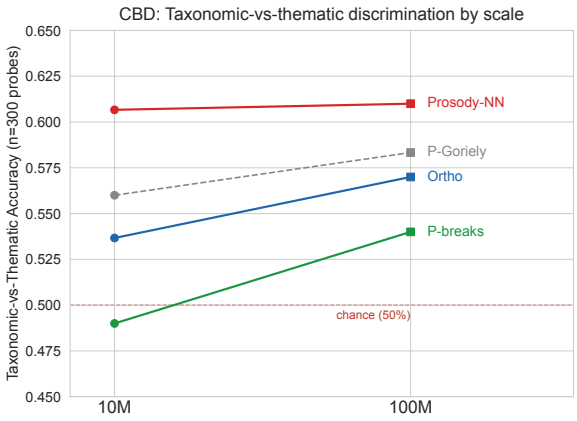


Figure 1: Taxonomic-versus-thematic accuracy at 10M and 100M for the four conditions. The dashed line marks chance (0.50).

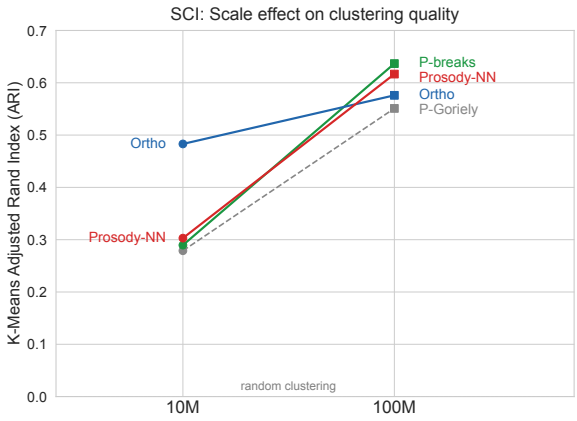


Figure 2: K-means ARI at 10M and 100M for the four conditions. P-BREAKS and PROSODY-NN cross above ORTHO at 100M; the dashed line marks ARI = 0.

Condition	K-ARI	K-NMI	A-ARI
10M			
ORTHO	**0.483**	**0.601**	**0.337**
P-GORIELY	0.279	0.439	0.257
P-BREAKS	0.290	0.417	0.260
PROSODY-NN	0.303	0.454	0.259
100M			
ORTHO	0.576	0.681	**0.717**
P-GORIELY	0.551	0.666	0.503
P-BREAKS	**0.637**	**0.750**	0.530
PROSODY-NN	0.617	0.732	0.610

Table 6: SCI cluster-recovery scores ($n = 68$ words, $k = 6$). *K-ARI*, *K-NMI*: k-means ARI and NMI (K-ARI primary); *A-ARI*: agglomerative ARI. Bold marks the best condition in each column within a scale.

The detailed embedding geometry could be found in Appendix H.

6 Discussion

6.1 Lexical Decision

Pseudo-log-likelihood is strongly influenced by phonotactic familiarity. The near-ceiling accuracy for LexicalMemory reflects only the fact that a single-token real word receives strong probability under virtually any model. The accuracy for Composition is lower, but still above chance, because the real word's multi-token sequence is partially learned while the pseudoword's multi-token sequence is unseen. The Inverted category is decisive: every condition sits near or below chance. This means that the task does not incisively probe lexicality; high wordlikeness can cancel out the partial ability to learn meaningfulness, that was seen in the Composition category. At 100M the phonemic conditions fall below chance while ORTHO stays at chance: the larger model has learned its input's phonotactics more sharply and scale deepens PLL's phonotactic dependence rather than relieving it. These findings suggest that the high lexical accuracy in Lavechin et al. (2022) for the PLL lexical decision task is partially an artefact of the BabySLM stimulus design.

6.2 Semantic Organisation

For the CBD task, PROSODY-NN beats the orthographic and plain-phonemic baselines on taxonomic-versus-thematic discrimination at both scales (Figure 1). For the SCI task, it rises above ORTHO at 100M (Figure 2). PROSODY-NN's performance indicates that its boundary and tone tokens reshape the embedding space toward categorical rather than merely thematic similarity. ORTHO keeps the lead only on the easier taxonomic-versus-unrelated comparison, where co-occurrence alone separates a category neighbour from an unrelated word.

The behaviour of P-BREAKS is striking. It is the weakest condition on taxonomic-versus-thematic, yet on cluster quality it matches or exceeds PROSODY-NN and rises above ORTHO at 100M, where it posts the highest k-means ARI. Break tokens therefore appear to supply a segmentation scaffold that yields well-separated clusters, but on some basis other than taxonomy. The categorical sharpening that CBD rewards appears to come from the sentence-type tone annotation that distinguishes PROSODY-NN from P-BREAKS, not from the break structure alone.

A second conspicuous pattern is the slope of the scale effect. Every condition improves from

10M to 100M. However, the phonemic conditions climb far more steeply than ORTHO (Figure 2), converting a ten-fold increase in data into categorical organisation more efficiently than ORTHO does. We read this as a difference in what each representation still has to gain. Whitespace gives ORTHO its word units for free, so its category structure rests on the co-occurrence with other words. By 10M, it has obtained almost as much advantage from the whitespace as is possible. In contrast, the phonemic conditions must induce the word boundaries, and prosodic tokens mark the phrase edges that shape which co-occurrences count. By 100M words, it appears that the phonemically-induced word boundaries may actually be better than those marked by whitespace in ORTHO, overcoming some of the idiosyncracies of English spelling and punctuation. The margins on any single probe set are small (the binomial standard error at $n = 300$ is about 0.028), but the pattern is consistent across both tasks, both scales, and the more robust taxonomic-advantage measure; the tighter, isotropic clusters of the boundary-annotated conditions are visible in Appendix H.

6.3 Conclusion

Goriely et al. (2024) established that training on continuous phonemic streams is a strong competitor to orthographic text at BabyLM scale. Our work extends that framework with explicit prosodic structure, a phonotactically controlled evaluation, and a comparison across two scales over four conditions. Two findings stand out. First, our redesigned lexical decision task shows that pseudo-log-likelihood scoring is strongly influenced by phonotactic likelihood rather than probing lexical knowledge alone: every model sits around or below chance on the adversarial Inverted category, and accuracy moves with the phonotactic balance rather than with lexicality alone. Second, on the semantic tasks, prosodic annotation makes phonemic models competitive with, or in some cases better than, the orthographic baseline.

Together these results indicate that, in the data-limited regime most analogous to infant learning, the structure and naturalness of the input representation matters. Orthographic text is not a neutral default but a cognitive artefact whose statistics a model can exploit. Because the child's input is speech and not text, a carefully constructed phonemic representation enriched with prosody is a key component of developmentally plausible language models, and one that future BabyLM work can build on.

Limitations

An important limitation of this study is that the prosodic annotation is not observed from speech. Instead, it is estimated from text. The boundary and tone tokens are estimated using a dependency parser and a BERT sequence labeller, which was trained on the Boston University Radio News Corpus. Further, the read broadcast-news register of this corpus differs from the conversational and child-directed speech that dominates the BabyLM corpus. Our annotations are therefore an approximation of the prosody an infant hears, and the gains we report are plausibly a lower bound on what genuine speech-based training would yield. Phonemisation adds a further approximation: eSpeak-NG applies a single American-English accent and rule-based grapheme-to-phoneme conversion, which introduces systematic transcription errors shared across all phonemic conditions.

The study used BPE encoding of the input, as this encoding is used in the baselines to which we make comparisons. When the white space marking word boundaries is removed, BPE encoding of an utterance can result in single tokens that bridge a (de facto) word boundary; see Appendix F for further details. We do not consider alternative character-based tokenisation schemes. These have provided promising results in some BabyLM studies (Goriely et al., 2024; Bunzeck and Zarrieß, 2025; Bunzeck et al., 2025). Exploration of character-based models for our experimental conditions and tasks is an important direction for future research.

The evaluations are rather narrow. The semantic tasks use 68 high-frequency CDI words in six broad categories, with hand-curated thematic foils yielding 300 probes, so the margins on any single metric are small and the categories may miss finer-grained structure; low-frequency and out-of-vocabulary generalisation is untested. All models are trained with a single random seed, so we report no variance estimates, and because the two scales differ in both corpus size and model capacity, scale effects conflate the two and support only within-scale condition rankings. Finally, our lexical decision analysis relies on pseudo-log-likelihood alone; a discriminative probe of the same representations, and a full account of why P-BREAKS forms high-quality clusters on a non-taxonomic basis, are left to future work, as is the natural step of replacing text-derived annotation with acoustic prosody from speech or self-supervised speech units.

Acknowledgments

We are grateful to Marvin Lavechin for his prompt and helpful responses to our questions about the construction of the BabySLM dataset. We also appreciate the helpful input from members of the Pierrehumbert Language Modelling Group over the course of the project.

References

Todd M Bailey and Ulrike Hahn. 2001. Determinants of wordlikeness: Phonotactics or lexical neighborhoods? *Journal of Memory and Language*, 44(4):568–591.

Mary E. Beckman and Gayle Ayers Elam. 1997. Guidelines for ToBI labelling, version 3. Technical report, Department of Linguistics, The Ohio State University.

Mathieu Bernard and Hadrien Titeux. 2021. Phonemizer: Text to phones transcription for multiple languages in python. *Journal of Open Source Software*, 6(68):3958.

Bastian Bunzeck, Daniel Duran, Leonie Schade, and Sina Zarrieß. 2025. Small language models also work with small vocabularies: Probing the linguistic abilities of grapheme- and phoneme-based baby llamas. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6039–6048, Abu Dhabi, UAE. Association for Computational Linguistics.

Bastian Bunzeck and Sina Zarrieß. 2025. Subword models struggle with word learning, but surprisal hides it. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 286–300, Vienna, Austria. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025. BabyLM turns 3: Call for papers for the 2025 BabyLM workshop. *Preprint*, arXiv:2502.10645.

Anne Christophe, Sara Millotte, Séverine Bernal, and Jeffrey Lidz. 2008. Bootstrapping lexical and syntactic acquisition using prosodic and phonological constraints. *Language and Speech*, 51(1–2):61–75.

Robert Daland and Janet B. Pierrehumbert. 2011. Learning diphone-based segmentation. *Cognitive Science*, 35(1):119–155.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long and Short Papers)*, pages

4171–4186. Association for Computational Linguistics.

Jonathan Duddington and Reece H. Dunn. 2025. espeak ng: Open source speech synthesizer. https://github.com/espeak-ng/espeak-ng.

Larry Fenson, Philip S. Dale, J. Steven Reznick, Donna Thal, Elizabeth Bates, John P. Hartung, Stephen Pethick, and Judy S. Reilly. 1993. *MacArthur Communicative Development Inventories: User's Guide and Technical Manual*. Singular Publishing Group, San Diego, CA.

Lex Fridman and Yann LeCun. 2024. Yann LeCun: Meta AI, open source, limits of LLMs, AGI & the future of AI. Lex Fridman Podcast #416 transcript, accessed 2026-05-20.

Zébulon Goriely, Richard Diehl Martinez, Andrew Caines, Lisa Beinborn, and Paula Buttery. 2024. From babble to words: Pre-training language models on continuous streams of phonemes. *Preprint*, arXiv:2410.22906.

Jennifer Hay, Janet B. Pierrehumbert, and Mary E. Beckman. 2004. Speech perception, well-formedness and the statistics of the lexicon. In John Local, Richard Ogden, and Rosalind Temple, editors, *Papers in Laboratory Phonology VI*. Cambridge University Press, Cambridge.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2(1):193–218.

Mutsumi Imai, Dedre Gentner, and Noburo Uchida. 1994. Children's theories of word meaning: The role of shape similarity in early acquisition. *Cognitive Development*, 9(1):45–75.

Peter W. Jusczyk. 1997. *The Discovery of Spoken Language*. MIT Press, Cambridge, MA.

Emmanuel Keuleers and Marc Brysbaert. 2010. Wuggy: A multilingual pseudoword generator. *Behavior Research Methods*, 42(3):627–633.

Marvin Lavechin, Maureen de Seyssel, Jules Lévêque, Emmanuel Dupoux, and Alejandrina Cristia. 2022. BabySLM: Language-acquisition-friendly benchmark of self-supervised spoken language models. In *Proceedings of Interspeech 2022*, pages 1641–1645. ISCA.

Emilie L. Lin and Gregory L. Murphy. 2001. Thematic relations in adults' concepts. *Journal of Experimental Psychology: General*, 130(1):3–28.

Brian MacWhinney. 2000. The childes project: Tools for analyzing talk (third edition): Volume I: Transcription format and programs, volume II: The database. *Computational Linguistics*, 26(4):657–657.

Jeremy M. Needle, Janet B. Pierrehumbert, and Jennifer B. Hay. 2022. *Phonotactic and Morphological Effects in the Acceptability of Pseudowords*, chapter 4. Cambridge University Press.

Tu Anh Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Evgeny Kharitonov, Alexei Baevski, Ewan Dunbar, and Emmanuel Dupoux. 2020. The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling. *arXiv preprint arXiv:2011.11588*.

Mari Ostendorf, Patti J. Price, and Stefanie Shattuck-Hufnagel. 1995. The Boston University radio news corpus. LDC Catalog No. LDC95S5.

Jenny R. Saffran, Richard N. Aslin, and Elissa L. Newport. 1996. Statistical learning by 8-month-old infants. *Science*, 274(5294):1926–1928.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Kim Silverman, Mary Beckman, John Pitrelli, Mari Ostendorf, Colin Wightman, Patti Price, Janet Pierrehumbert, and Julia Hirschberg. 1992. ToBI: A standard for labeling English prosody. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, volume 2, pages 867–870.

Melanie Soderstrom, Amanda Seidl, Dana Gerber Kemler Nelson, and Peter W. Jusczyk. 2003. The prosodic bootstrapping of phrases: Evidence from prelinguistic infants. *Journal of Memory and Language*, 49(2):249–267.

Lyn Tieu and Jeffrey Lidz. 2016. Npi licensing and beyond: Children's knowledge of the semantics of any. *Language Acquisition*, 23(4):311–332.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355. Association for Computational Linguistics.

Joe H. Ward. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the

BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094.

Appendix A Model Architecture and Training

Hyperparameter	10M	100M
Architecture		
Variant	BERT-small	BERT-base
Hidden size	256	768
Layers	8	12
Attention heads	8	12
FFN size	1,024	3,072
Vocabulary	4,000	8,000
Parameters	≈7.7M	≈92M
Training		
Epochs	10	
Max sequence length	256	
Batch size / device	1,024	128
Gradient accumulation	1	4
Peak learning rate	5×10^{-4}	1×10^{-4}
LR schedule	cosine, 10% warmup	
Optimizer	AdamW	
Weight decay	0.01	
Gradient clipping	1.0	
Precision	bfloat16	
MLM masking	15%	
Random seed	42	

Table 7: Model architecture and training hyperparameters at the two scales. Architecture dimensions differ by scale; training settings are shared except batch size, gradient accumulation, and peak learning rate.

Appendix B Training Corpus and Prosodic Annotation

Corpus composition. All conditions are trained on the BabyLM 2025 corpus (Charpentier et al., 2025); Table 8 gives the composition, sampled at the same proportional weights at both scales.

ToBI inventory. The prosodic tokens are drawn from the Tone and Break Indices framework (Silverman et al., 1992; Beckman and Ayers Elam, 1997). We use two levels of the break-index hierarchy: `[B2]` for an intermediate-phrase boundary (ToBI break index 2) and `[B4]` for an intonation-phrase boundary (index 4); indices 0, 1, and 3 are not used. We use two boundary tones: `[H%]`, a high (rising) tone associated with yes/no questions, and `[L%]`, a low (falling) tone associated with declaratives, imperatives, and wh-questions. Any annotation method must be text-only, deterministic, fully automatic, and transparent, since no paired audio exists for the BabyLM sources.

Source	10M	100M	Share
CHILDES (child-directed)	2.8M	28.9M	29%
Project Gutenberg (prose)	2.5M	26.4M	26%
OpenSubtitles (dialogue)	2.0M	20.0M	20%
Simple Wikipedia	1.5M	14.7M	15%
BNC Spoken (conversation)	0.9M	7.8M	8%
Switchboard (telephone)	0.1M	1.3M	1%
Total	9.95M	99.0M	100%

Table 8: BabyLM training corpus composition (approximate word counts).

Rule-based break annotation (P-BREAKS). Each sentence is parsed by spaCy's `en_core_web_sm` dependency model (Honnibal et al., 2020) (NER and lemmatisation disabled for throughput). `[B4]` is inserted after any token that is a sentence boundary or bears terminal punctuation (`.`, `?`, `!`) and takes precedence over `[B2]` at the same position. `[B2]` is inserted, when `[B4]` has not been assigned, under the five syntactic cases in Table 9. For sentences of three or fewer whitespace words, cases A–C and E are suppressed and only case D and the sentence-final `[B4]` apply, reflecting the lower parse reliability and negligible internal structure of short conversational turns. Because standard phonemisation concatenates the utterance, tokens are placed by a two-pass scheme: each word is phonemised separately with a temporary separator, then the boundary tokens are interleaved at the recorded positions.

Neural annotator (PROSODY-NN). PROSODY-NN replaces the rule-based break detector with a trained sequence labeller while keeping the same token vocabulary and two-pass pipeline. The annotator is trained on the Boston University Radio News Corpus (Ostendorf et al., 1995), the only large manually ToBI-annotated American English corpus, comprising roughly 3,600 sentences and 113,000

Rule	Trigger
A	Coordinating clause junction: a coordinating conjunction heads a clause-level `cc` dependency on a verbal or auxiliary root.
B	Subordinate or complement clause onset: a subordinating conjunction or complementiser marks an adverbial, complement, or embedded clause.
C	Relative clause onset: a relative pronoun or the `relcl` arc.
D	Minor surface punctuation: comma, semicolon, colon, dash, or ellipsis.
E	Appositive or parenthetical nominal: a token bearing the `appos` dependency and not preceded by punctuation.

Table 9: Syntactic cases that trigger a `[B2]` token in P-BREAKS.

word tokens of read radio news after excluding one speaker who lacks break annotation. Break indices are collapsed to {NONE, B2, B4} and tone events (mapped to the nearest following word boundary) to {NONE, L%, H%}; splits are speaker-disjoint. The model is a `bert-base-uncased` backbone (Devlin et al., 2019) with two linear heads over the first-subword hidden state of each word, trained to minimise a joint class-weighted cross-entropy $\mathcal{L} = \mathcal{L}^{\text{brk}} + \lambda\,\mathcal{L}^{\text{ton}}$ with weights $[1, 3, 3]$ (break) and $[1, 4, 8]$ (tone); the elevated `[H%]` weight compensates for the rarity of yes/no questions in read news. Training uses AdamW at learning rate 2×10^{-5} with early stopping on held-out macro-F_1.

Each head is used for corpus annotation only if it clears a held-out F_1 gate (Table 10); the break head otherwise falls back to the rule-based P-BREAKS pipeline and the tone head to the heuristic below. The final annotator clears every threshold, so both heads are shipped. Inference over the full 100M corpus takes about an hour on a single GPU.

Head	F_1	Ship threshold
`[B2]`	0.58	≥ 0.50
`[B4]`	0.99	≥ 0.80
`[L%]`	0.69	≥ 0.60
`[H%]`	0.56	≥ 0.30

Table 10: Held-out F_1 of the neural annotator against the shipping thresholds. All four clear their gate, so the full annotator is used.

Rule-based tone heuristic (fallback). When the tone head fails its gate, sentence-final tone is assigned by a three-gate rule, which is also the source of the boundary tones' definition. A sentence whose final non-bracket character is not `?` receives `[L%]`. Otherwise, the first substantive word is found by skipping up to five leading discourse markers, fillers, and response tokens: a wh-word receives `[L%]`, a fronted positive auxiliary or copula receives `[H%]` (contracted negations are excluded as they often head echo questions), and any remaining question receives `[L%]` conservatively. In our run the tone head cleared its gate, so this heuristic was not triggered.

Appendix C Pseudoword Generation and Phonotactic Control

Generation: a proposal stage and a selection stage. Pseudowords are produced in two stages. The proposal stage supplies candidate forms and the selection stage decides which candidates enter the test set. The proposal stage is a trigram Markov chain over orthographic characters, trained on the alphabetic types of the training vocabulary, sampled autoregressively from $\langle s\rangle\langle s\rangle$ until $\langle/s\rangle$ with a temperature T that reshapes the empirical transition counts as $P_T \propto c^{1/T}$. Candidates are constrained to between four and seven characters. Each candidate is then phonemised by the same eSpeak-NG pipeline that produced the training corpora, and every subsequent step operates on the resulting IPA form: the collision filters, the phonotactic score S, the length and score matching, and the construction of the Inverted category. The chain is therefore a proposal distribution only. None of the properties the test set relies on are inherited from it, and all of them are imposed in phonemic space. This is what separates the design from a Wuggy-style pipeline, which controls sub-syllabic substitutions in the orthographic domain and phonemises afterwards, leaving the phonemic properties of the pair uncontrolled.

Selection and pairing. Real words are drawn from the Filter-A-validated pool (Appendix D) by stratified sampling over ten score strata, restricted to items of four to seven IPA phones. Candidate pseudowords are generated at five times the required pool size for the standard categories and screened by Filter B, with rejections backfilled so category sizes are held fixed. Each real word is then matched to the pseudoword nearest it in phonotactic score *drawn only from candidates with the same number of IPA phones*, subject to $|S_{\text{real}} - S_{\text{fake}}| \leq 2.0$. Every one of the 4,129 pairs is exactly length-matched in phones as a result; the mean absolute score difference is 0.19 for LexicalMemory and 0.16 for

Composition. The set comprises 967 LexicalMemory, 1,162 Composition and 2,000 Inverted pairs.

Phonotactic control in the BabySLM benchmark: The same phonemic control is absent from the benchmark our task replaces. Wuggy controls the biphone statistics of its pseudowords, but not triphone statistics, which are known to carry word-likeness (Bailey and Hahn, 2001; Hay et al., 2004; Needle et al., 2022). Table 11 scores all 17,706 real/pseudoword pairs of the BabySLM lexical development set under both models. Under the biphone model Wuggy does control, the real word is favoured on 58.1% of trials, close to the residual asymmetry expected from real words being better formed on average. Under the uncontrolled triphone model it is favoured on 70.3%. A model that had learned nothing beyond triphone phonotactics would therefore score close to 70% here, despite having no lexical knowledge at all.

	Biphone (controlled)	**Triphone** (uncontrolled)
Real word favoured	58.1%	70.3%
Mean difference	+0.31	+1.21
Median difference	+0.21	+0.89
Skew	+0.56	+0.75
Mean − median	+0.10	+0.32
Cohen's d	0.13	0.28
KS D	0.070	0.128
Wilcoxon	p=1.5×10^{-157}	p<10^{-300}

Table 11: Phonotactic asymmetry between the real words and the Wuggy pseudowords of the BabySLM lexical development set (n = 17,706 pairs), under a biphone model, which Wuggy controls, and a triphone model, which it does not.

Appendix D Lexical Decision: Filtering and Examples

The two filters. *Filter A* (real-word validity) admits a candidate real word only if it is attested by `wordfreq` *and*, after WordNet lemmatisation, present in the NLTK `words` list. `wordfreq` alone admits web artefacts such as *https* and repeated-character strings, while the NLTK list alone lacks inflected forms; gating one on the other keeps inflected and loan words while removing non-words. *Filter B* (pseudoword screen) rejects a generated pseudoword if its orthographic form occurs in the training vocabulary, or if its eSpeak IPA form, either verbatim or with stress and length marks stripped, occurs in the real-word IPA lexicon (the set of IPA forms of all corpus words). Rejected candidates are resampled so category counts are held fixed.

Contamination. Filter A reduced the real-word pool from the 82,548 alphabetic corpus types to 43,947 validated English words, rejecting 38,601 (47%) as non-words. Filter B then removed a substantial number of pseudowords whose phonemic form coincided with a real word (Table 12); in the Inverted category the 3,244 IPA collisions exceed the 2,000 pairs finally retained, confirming that without the phonemic screen a large share of "pseudowords" would in fact be real words in IPA space.

Category	**Filter B IPA collisions removed**
LexicalMemory	959
Composition	1,053
Inverted	3,244

Table 12: Pseudowords rejected by the phonemic screen (Filter B) because their IPA form matched a real corpus word.

Example pairs. Table 13 shows representative pairs from each category. In LexicalMemory and Composition the phonotactic scores are matched within the 2.0 threshold; in Inverted the pseudoword's score is far higher than the real word's by construction.

Real	**Pseudoword**	**S$_{real}$**	**S$_{fake}$**
LexicalMemory			
pencil /pɛnsəl/	blaver /bleɪvɚ/	−18.7	−18.7
heart /hɑːɹt/	pred /pɹɛd/	−11.9	−12.0
Composition			
reissued /ɹiːɪʃuːd/	pophini /pəfiːni/	−29.1	−29.1
foppish /fɑːpɪʃ/	cenfse /sɛnfs/	−25.0	−25.0
Inverted			
gestalt /gəʃtɑːlt/	plation /pleɪʃən/	−39.9	−13.2
kinetic /kɪnɛrɪk/	noment /noʊmənt/	−37.5	−13.3

Table 13: Example real-word / pseudoword pairs with Witten-Bell triphone phonotactic scores S. Standard pairs are score-matched; Inverted pairs satisfy $S_{\text{fake}} > S_{\text{real}}$.

Appendix E Target Vocabulary

The semantic tasks use the 68 CDI words (Fenson et al., 1993) listed in Table 14, drawn from the earliest vocabulary of English-speaking children and high-frequency in the CHILDES development set of the BabyLM corpus. The six categories are concrete semantic fields whose categorical structure is well established developmentally, so the benchmark

tests whether the distributional structure the model has seen is reflected in its embedding geometry rather than generalisation to novel words.

Category	Words
Animal	dog, cat, bird, fish, bear, duck, horse, cow, pig, frog, mouse, sheep
Food	milk, juice, water, cookie, apple, banana, bread, cheese, cake, egg, soup, rice
Body part	eye, nose, mouth, ear, hand, foot, head, hair, finger, knee, arm, leg
Clothing	shoe, sock, hat, shirt, coat, dress, boot, sweater, diaper, button, pocket
Vehicle	car, bus, truck, bike, boat, plane, wagon, stroller, tractor
Household	cup, spoon, bowl, plate, bottle, chair, table, bed, blanket, door, phone, clock

Table 14: The 68 CDI target words by semantic category (Fenson et al., 1993).

Appendix F Tokenisation

Training. Every condition trains its own Byte-Pair Encoding tokeniser (Sennrich et al., 2016) from scratch on its own preprocessed corpus, using the HuggingFace `tokenizers` implementation. Vocabulary size is 4,000 at 10M and 8,000 at 100M, chosen to hold the ratio of vocabulary to model parameters roughly constant across the two scales; the minimum merge frequency is left at the library default of 2, and no merge-count or vocabulary sweep was run. The 10M orthographic tokeniser learns 3,708 merges and the 10M P-Goriely tokeniser 3,938, the balance of each vocabulary being the base character inventory. Out-of-vocabulary material falls back to `[UNK]`, and BPE dropout is not used.

Structural tokens (`[B2]`, `[B4]`, `[H%]`, `[L%]`, `[UTT]`, `[SPEAKER]`, `[HEADER]`, and the reserved BERT tokens) are filtered out of the corpus before BPE training and reserved at fixed vocabulary ids by the trainer. Without this the pre-tokeniser splits `[SPEAKER]` into `[`, `SPEAKER`, `]` and BPE learns merges inside the token name.

Segmentation. At application time BPE begins from the character sequence and repeatedly applies the highest-priority applicable merge from the learned merge list, in the order the merges were learned. It is not a longest-match scan: a longer entry in the vocabulary will not be preferred if the merges that would build it rank below the merges that build a different segmentation.

Pre-tokenisation, and the asymmetry between conditions. All conditions use the same whitespace pre-tokeniser, but it does very different work in each. Ortho retains ordinary inter-word whitespace, so the pre-tokeniser splits the stream at every word boundary and no merge can span one: word boundaries are preserved by construction, and the model receives them for free. The phonemic conditions have inter-word whitespace removed, which is the defining property of the Goriely et al. (2024) setting we build on, so the pre-tokeniser splits only around structural tokens and effectively presents BPE with the whole utterance as a single unit. Merges may therefore span word boundaries, and no word boundary is marked in the input at any point.

This asymmetry is deliberate rather than incidental: it is what modelling continuous speech without orthographic segmentation means. It does, however, mean that the orthographic baseline is handed a source of information the phonemic conditions must infer.

Appendix G Token Budget by Condition

Annotating a phoneme stream with structural tokens necessarily lengthens it, which raises the question of whether any advantage of the annotated conditions is a token-budget effect rather than a representational one. Each unified corpus was tokenised with its own trained tokeniser, with `add_special_tokens=False` so that no `[CLS]` or `[SEP]` padding is counted.

Composition of the structural share. At 10M, P-breaks spends 795,812 tokens on `[B2]` and 1,356,806 on `[B4]`, and Prosody-NN spends 883,401 on `[B2]`, 1,166,168 on `[B4]`, 17,723 on `[H%]` and 36,927 on `[L%]`. At 100M the corresponding counts are 7,852,083 and 13,443,590 for P-breaks, and 8,757,921, 11,465,427, 172,011 and 385,551 for Prosody-NN. P-Goriely carries only `[UTT]`, 1,010,527 at 10M and 9,875,227 at 100M. Neither annotated condition emits it: in both, the utterance-final position is carried by `[B4]`. The two annotators differ in how often they place `[B4]` utterance-internally, the rule-based annotator doing so on roughly 192,000 lines at 10M against roughly 1,700 for the neural annotator, which is why P-breaks carries slightly more structural tokens than Prosody-NN despite emitting no tones.

Condition	Tokens	Per utt.	Structural	% struct.	vs P-GORIELY	vs ORTHO
10M						
ORTHO	14,979,765	12.85	0	0.00%	+22.0%	n/a
P-GORIELY	12,276,352	10.55	1,010,527	8.23%	n/a	−18.0%
P-BREAKS	13,498,252	11.59	2,152,618	15.95%	+10.0%	−9.9%
PROSODY-NN	13,417,556	11.52	2,104,219	15.68%	+9.3%	−10.4%
100M						
ORTHO	140,210,835	12.24	0	0.00%	+27.8%	n/a
P-GORIELY	109,677,972	9.58	9,875,227	9.00%	n/a	−21.8%
P-BREAKS	122,184,461	10.67	21,295,673	17.43%	+11.4%	−12.9%
PROSODY-NN	121,277,565	10.59	20,780,910	17.13%	+10.6%	−13.5%

Table 15: BPE token counts per condition and scale, with the share contributed by structural tokens and the budget relative to P-GORIELY and to ORTHO.

Interpretation. Prosodic annotation does inflate the budget within the phonemic family, by 9.3% to 10.0% at 10M and 10.6% to 11.4% at 100M relative to P-GORIELY, so the confound is real for comparisons between the annotated conditions and P-GORIELY. It cannot, however, explain any phonemic advantage over orthography, because every phonemic condition still consumes markedly fewer tokens than ORTHO: the annotated conditions sit 9.9% to 10.4% below at 10M and 12.9% to 13.5% below at 100M, and P-GORIELY is 18.0% to 21.8% below. The orthographic baseline has the largest token budget of all eight models and is nonetheless overtaken at 100M on unsupervised category induction. The asymmetry runs against the annotated conditions, not in their favour.

Appendix H UMAP Projections - Semantic Category Structure

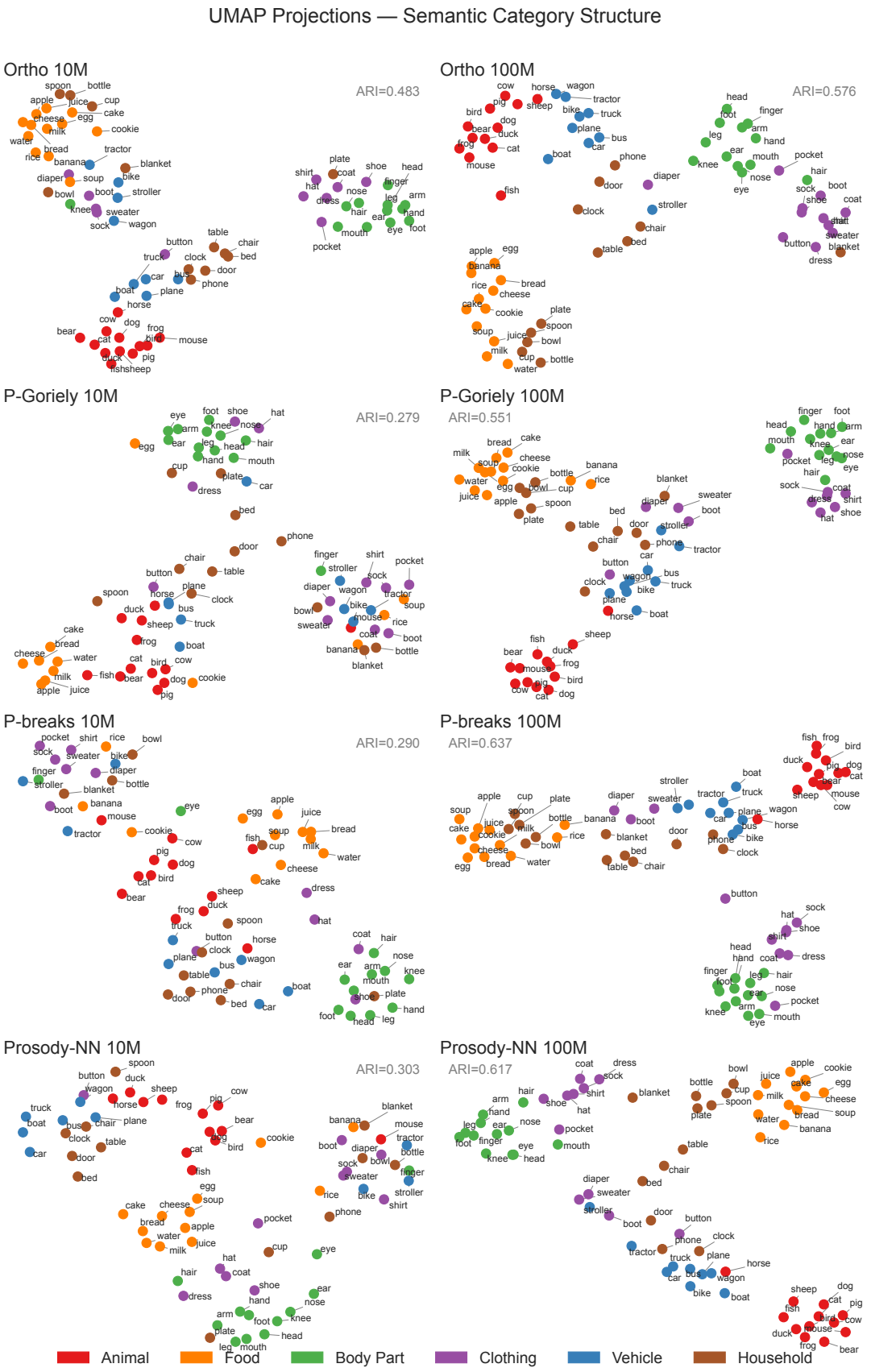


Figure 3: UMAP projections of the 68 CDI word embeddings, coloured by semantic category, for Ortho and Prosody-NN at both scales; k-means ARI is annotated in each panel.

Augustinian BabyLM: What Ostensive Definition Can and Cannot Teach a Small Language Model

Lisa Bylinina
Institute for Language Sciences
Utrecht University
e.g.bylinina@uu.nl

Abstract

A language model normally begins training with random word embeddings: whatever *banana* means must be learned from training corpora. I implement St. Augustine's picture of word learning, meaning by ostension, for a small masked language model (DeBERTa) trained on ≈10M words: before training, visually grounded tokens receive embeddings derived from the image regions they label; other tokens start random.

Visual initialization leaves a measurable imprint that lasts until the end of training. At the same time, the effect remains invisible under most BabyLM benchmarks, which probe abstract grammatical knowledge: visual initialization does not affect performance there. The only zero-shot exception is object-property knowledge (COMPS, Misra et al. 2023), where seeding helps in every configuration. To follow up on this result, I build a corpus-tailored version of the Visual-Property Swap benchmark (Lin et al., 2026), which tests color, material, size, and shape knowledge, with per-item training frequency and seeded status. Here, vision-seeded models have a persistent, seed-replicated advantage, confined to the seeded words. As a causal test, I show that synthetic grounding of previously unseeded words transfers the advantage to exactly those words.

Function words and abstract vocabulary also receive strong visual seeds and retain them throughout training, and the training objective draws on them: held-out mask-prediction loss falls for these words in every seed. However, no benchmark I run registers this. What evaluation would pick this up remains an open question.

1 Introduction

> When they (my elders) named some object . . . I grasped that the thing was called by the sound they uttered.
>
> *Confessions* I.8.13 (Augustine of Hippo, 1991)

Augustine's account of how he learned to speak is probably the oldest theory of lexical acquisition we have: elders point at things and say their names, and the child, seeing what is pointed at, binds sound to object. Wittgenstein (1953) opened the *Philosophical Investigations* by arguing against this view: ostension cannot be the basis of word learning, because pointing presupposes that the learner already parses the scene into objects and properties, and in that sense already has the space of lexical meanings ready. On the other hand, core-knowledge research (Spelke and Kinzler, 2007) demonstrates exactly this kind of prior in newborns, with cognitive systems for objects, number, agents, and space. So, the matter is not settled.

I look for effects of visual demonstration in language learning using computational methods, in particular, in language models. A typical text-only language model is not equipped for ostension: its word embeddings begin as random vectors, and their meanings must be discovered from distribution alone. For comparison, I supply the visual demonstration: before any text is seen, words that label regions in visual data are initialized from a vision encoder's features over those regions (Fig. 1); the model undergoes text-only training from that point on. The BabyLM strict-small budget, roughly a child's language input without the embodied part, is where I would expect this prior to matter (cf. Zhuang et al., 2024). I quantify its effect against a matched text-only baseline.[1]

I show that visual initialization has a measurable effect: its imprint on the embeddings lasts to the end of training, and the training objective itself becomes easier for the words that received visual grounding. This effect is, however, difficult to ob-

[1] Code, models, and my version of the VP-Swap benchmark are available at https://github.com/bylinina/augustinian_babylm. All trained models, the per-region visual features, and the per-token embedding tables are released under the augustinian-babylm organization on the Hugging Face Hub: https://huggingface.co/augustinian-babylm.

Proceedings of the Second BabyLM Workshop, pages 165–178
October 29, 2026 ©2026 Association for Computational Linguistics

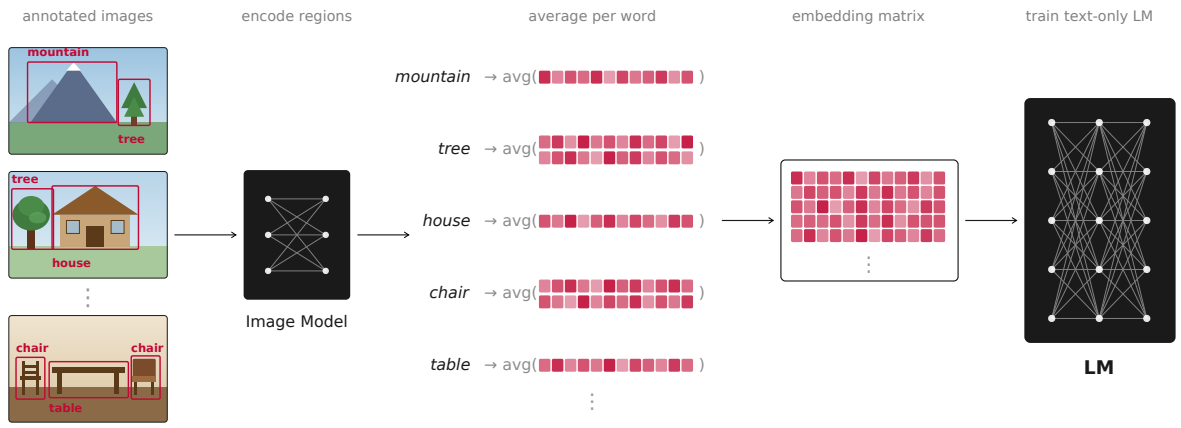


Figure 1: The vision-seeding pipeline. For each word that labels regions in the visual grounding data, I encode those regions with a frozen vision model, average the region features into a single vector, and write it (projected and scaled) into the corresponding row of the language model's input embedding matrix; words with no image support keep their random initialization. The language model is then trained on text only.

serve through the standard BabyLM benchmarks: barely any of them probe the aspects of meaning for which visual information is relevant. The exception is COMPS (Misra et al., 2023), which tests object-property knowledge and improves in all configurations.

Following this observation, I build my version of VP-Swap (Lin et al., 2026), a visual-property minimal-pair benchmark. There the advantage is clear: it's persistent, replicated across seeds, and specific to the words that were actually grounded; finally, it extends to new words when I ground them synthetically. Abstract and function words complicate the picture: their seeds are equally strong and equally well retained, and the objective draws on them, but nothing I measure rewards it.

Contributions. (1) An effect of vision-seeded token embedding initialization for a small language model under the 10M-word training budget (frequency-matched, word-specific, placebo-controlled, replicated under different configurations and seeds, corresponding to a consistent object-property gain according to the COMPS benchmark). (2) An adjusted version of VP-Swap: a corpus-tailored visual-property benchmark with per-item training-frequency and seeded-status metadata. (3) A causal test: synthetically grounding unseeded words transfers the benefit to exactly those words. (4) A mechanistic account of how the effect survives the embeddings' departure from their visual anchors. (5) A scope result: for function words the visual seeds are strong, retained, and relevant for the learning objective, yet the effect is not detected by existing benchmarks.

2 Related Work

Multimodal learning under the BabyLM budget. Whether visual input helps models learn language in a more human-like way has been a recurring BabyLM question in the (now abandoned) multimodal track (Hu et al., 2024). Results have been modest so far: submissions have mostly not beaten the baselines (Hu et al., 2024; Charpentier et al., 2025), and multimodal training has sometimes harmed language-only performance, under the catastrophic forgetting effect when a text model is fine-tuned on linguistically deficient caption data (Amariucai and Warstadt, 2023; Klerings et al., 2024). Even approaches that beat the baselines inherit the problem: Takmaz et al. (2025) report models that beat prior baselines but fall behind on grammar-focused text-only benchmarks, motivating inference-time model merging with a text-only model to recover the lost ability. My approach addresses the same issue in a different way: visual information is supplied only once, as the initialization of a subset of embedding rows, with no continued caption signal and no change to the architecture or training objective. The model is a regular text-only masked language model.

Grounded word representations. The idea that word meaning has a perceptual component with measurable consequences for representation has a long history, from multimodal distributional semantics (Bruni et al., 2014) to the finding that grounded and text-only models decode brain activity differ-

ently for concrete versus abstract nouns (Anderson et al., 2017). I reuse this insight in a different setup: features from a frozen vision encoder (Siméoni et al., 2025; Zhou et al., 2022; Kirillov et al., 2023) supply the starting embeddings for the words that name image regions. Unlike approaches that inject grounding through auxiliary objectives or cross-modal attention, my setting does not affect the model's parameters beyond the initial embedding values.

What targeted evaluations measure. The benchmarks that dominate small-model evaluation are mostly probes of abstract structural competence (Warstadt et al., 2020). Visual grounding is hardly expected to have a systematic effect here – instead, it should be relevant for aspects of lexical semantics: knowledge of what words denote and what their referents are like. This is also linguistically relevant knowledge, since it governs how words are used. So, an intervention that affects meaning will surface only on evaluations that address word meaning. Two such probes are important for me: COMPS (Misra et al., 2023), which tests knowledge of object properties, and VP-Swap, a visual-property minimal-pair benchmark I adapt from Lin et al. (2026). Their version tested a different intervention and found no effect. I rebuild the benchmark from my corpus so that each item records how often the model saw each word and whether that word was visually seeded. This lets me trace an effect to the specific words the intervention affected (§5).

3 The Augustinian Setup

My setup implements ostension as a single change to how training begins (Figure 1). A language model's input embedding matrix has one row per vocabulary item, and normally every row starts with random numbers. I replace this noise for some rows with a visual summary of the corresponding word: for each word that appears labeling regions of images in visual grounding data, I collect those regions, pass them through a frozen vision encoder, and average the resulting features into a single vector that becomes the word's initial embedding. Words that never label an image region keep their random start. Nothing else about the model or its training changes: the architecture, the objective, and the text data belong to the regular text-only masked language model training setup. In this way, any difference I observe between a visually seeded model and the baseline is attributable to this intervention. The rest of this section makes the two halves of the procedure precise: how a word's (more specifically, token's) visual embedding is computed (§3.1), and which words can receive one at all (§3.2).

3.1 Region Features and Seeding

For visual grounding I rely on a combination of four datasets that pair words with image regions: Flickr30k Entities (Plummer et al., 2015), RefCOCO+ (Kazemzadeh et al., 2014), RefCOCOg (Mao et al., 2016), and THINGS (Hebart et al., 2019), together 563k region annotations. For a target word, I take every region it labels, crop the region, and encode it with a frozen vision model; the region feature is the mean of the encoder's patch embeddings inside the bounding box, and the word's visual embedding is the average of these region features over all its occurrences (a word that tokenizes into several subwords contributes to each of the tokens). I report three encoders (DINOv3 Siméoni et al. 2025, iBOT ViT-B/16 Zhou et al. 2022, and SAM ViT-B Kirillov et al. 2023), which differ in training objective but are used identically here. Importantly, all three encoders are trained on images alone with no text or caption supervision, unlike vision–language encoders such as CLIP (Radford et al., 2021) whose features are shaped by language: DINOv3 and iBOT are trained by self-distillation on unlabeled images, SAM is trained on image-mask pairs. The visual seeds therefore carry a purely perceptual prior, uncontaminated by linguistic signal, which also means that the grounding adds no words to the model's language budget. All three encoders produce 768-dimensional features, matching DeBERTa-v3-base's embedding size, so no projection is needed; I only rescale each seeded vector (mean-centering, L_2-normalizing, and scaling to the standard deviation of the model's random initializer) so that seeded and unseeded rows are statistically comparable at the start of training. Tokens whose word never labels a region are left at their random initialization. Only input embeddings are seeded; all other parameters, and the tied output embeddings, start as usual.

3.2 Coverage: What Gets Seeded

Because only words that label image regions can be seeded, the intervention reaches a subset of the vocabulary. Of the $\sim$163k word types in the corpus, about 10% ever appear in the grounding data,

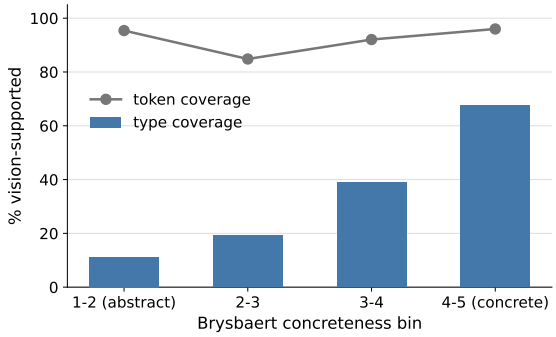


Figure 2: Seedability rises with concreteness. Word types binned by their Brysbaert concreteness rating; the height of each bar is the fraction of that bin appearing in the visual grounding data. Concrete words are far more likely to be seeded.

but these are common words, covering 85% of text. Subword tokenization raises type coverage by sharing subwords: the seedable fraction of the vocabulary is 37.7% at 50k merges, 29.1% at 75k, and 23.8% at 100k – larger vocabularies leave more rare whole-word rows un-seeded. Token coverage stays near 87% across vocabulary sizes. Seedable words skew strongly concrete (seededness correlates with concreteness norms at $r = 0.46$; Figure 2); predictably, verbs, discourse and abstract terms, and temporal expressions do not get strong visual support.

3.3 Models and Training

I train DeBERTa-v3-base masked language models on `bb24.train`, the ∼9.9M-word corpus of Edman et al. (2024), which mixes LLM-generated paraphrase data with portions of the official BabyLM corpus (not the official 2026 strict-small distribution). Crossing three BPE vocabulary sizes (50k/75k/100k) with the three image encoders, plus a random-initialized baseline per vocabulary, gives three baselines and nine vision-init models, each trained for 10 epochs under a fixed recipe (Appendix A) with checkpoints saved by word count. Runs are single-seed except the headline comparison (75k-SAM vs. 75k) and the synthetic-extension model 75k-SAM-ext, which I repeat with three seeds to address variance.

3.4 Released Artifacts

All twelve models are public. Baselines are named `deberta-base-`V and vision-init models `deberta-base-`V`-`E, for vocabulary $V \in \{50k, 75k, 100k\}$ and encoder $E \in \{sam, dinov3, ibot\}$; three-seed replicates add a `-s1`, `-s2` or `-s3` suffix. Intermediate checkpoints are stored as branches (`step0`, then `chck_1M` to `chck_100M`), so the training dynamics in Sections 5 and 7 can be reproduced without retraining. The region features and the per-token embedding tables that implement the seeding are released as datasets; the latter would be needed for another model in order to reuse the intervention.

4 Official BabyLM Evaluation

Measuring the effect. Every result below compares a vision-init model to its same-vocabulary baseline; a *delta* is vision-init minus baseline accuracy, in points, and all zero-shot tasks are scored as minimal pairs under masked-language-model pseudo-log-likelihood (mask each token in turn, sum the log-probability of the original). Because single runs give no variance estimate, I look at *sign-consistency*: a task whose delta is positive in all nine encoder×vocabulary configurations is unlikely under a no-effect null even when each delta is small. For the targeted analyses of Section 5 I add three tools: (a) a frequency-binned *difference-in-differences* (DiD) – the seeded minus the unseeded advantage *within* a corpus-frequency bin, which addresses any global effect that would help seeded and unseeded words alike; (b) a *placebo condition* – items whose relevant words were never seeded, where the delta should be zero if effects are attributable to seeding; and (c) *McNemar's test* (McNemar, 1947), which assesses whether the two paired models differ significantly on the items where they disagree. Following the diagnosis in Appendix H, I also check every minimal-pair task at initialization, where an unbiased probe should be at chance.

An effect where expected. Figure 3 gives the per-task deltas across all nine encoder×vocabulary configurations. Two tasks are consistently positive: object-property knowledge (COMPS, +1.30, 9/9) and GLUE (+1.08, 9/9). The rest are near zero and inconsistent: BLiMP and EWoK split roughly evenly across configurations, and the entity-tracking and supplement tasks are slightly negative. This is expected given the coverage analysis (§3.2): a vision prior should help with knowledge of what objects are like, and COMPS is the only standard zero-shot task that probes it directly.[2]

[2]The GLUE effect is harder to attribute: GLUE is evaluated by fine-tuning, and I decided not to investigate in detail how exactly my initialization gets exploited in the task-specific

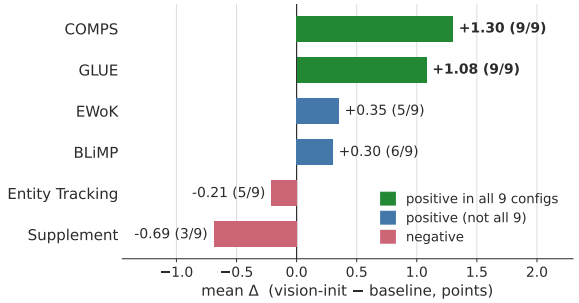


Figure 3: Per-task deltas on the official BabyLM 2026 evaluation, averaged over the nine encoder×vocabulary configurations; the count is how many of the nine are positive. Only object-property knowledge (COMPS) and GLUE are positive in every configuration.

The three vision encoders behave the same: COMPS and GLUE are positive in all nine configurations (three vocabularies × three encoders), so the effect does not depend on the choice of vision model. I observe the same encoder-agnostic behaviour in the artifact of Appendix H. For brevity I report the remaining analyses with a single representative encoder (SAM).

The COMPS gain is expected: COMPS pairs a concept with a property and asks the model to prefer the true sequence; its subconditions vary how the pair is presented. The gain is concentrated in the conditions that test property knowledge itself (the base condition $+1.37$, 9/9 and the harder novel-concept "wugs" condition $+3.65$, 9/9) and vanishes in the distractor conditions (≈ 0), where surface cues dominate. Vision-init helps the model know that, say, a banana is yellow.

The effect is stable. Because all configurations are single runs, I retrained the headline pair (75k-SAM vs. 75k) with three seeds and re-ran the full zero-shot suite. COMPS is the only zero-shot task whose sign is stable across all three seeds ($+1.46/+1.05/+0.77$); all other zero-shot tasks change signs depending on the random seed (BLiMP, EWoK, supplement, entity tracking).

Zooming in on EWoK subtasks buttresses the same conclusion: gains concentrate in visual and quantitative content (*number* $+6.53$, *quantitative-properties* $+3.08$, *material* $+2.15$) and disappear for social or material knowledge (Appendix B).

Leaderboard standing. I submitted my models to the official BabyLM 2026 leaderboard (Choshen et al., 2026), and they proved to be being competitive with the field despite the intervention touching only a fraction of the vocabulary. The result should be read as a progression across my three models, in which each step of the method results in an improvement (all figures as of 19 July 2026). The text-only baseline `deberta-base-75k` scores 37.91 overall; adding visual seeding (`deberta-base-75k-sam`) raises this to 39.63; and adding the synthetic-grounding extension (to be described in Section 6) (`deberta-base-75k-sam_ext-s1`) raises it again to 40.62, which ranks 5th of the strict-small submissions ordered by overall average and sits above the strongest official baselines (e.g. the Interaction baseline at 38.71). The same ordering holds, more sharply, on the Human-like Average: the baseline scores 0.61, the seeded model 7.74, and the extended model 12.19 – the highest of any entry on the board. I report the Human-like figures with a caveat, since they are driven largely by the age-of-acquisition component; but the monotonic improvement from baseline to seeded to extended, on both aggregate metrics, tracks exactly the design choices I introduce.

5 VP-Swap: A Visual-Property Probe

The official evaluation showed the effect only through a single zero-shot task (COMPS) and left its size and specificity unclear. To explore this effect further, I re-build VP-Swap (Lin et al., 2026), a minimal-pair probe of the knowledge a visual prior should bring in. In my version of the benchmark, each item is annotated with the training frequency and seeded status of its words, allowing me to trace which words carry the effect.

5.1 Benchmark Construction

Each VP-Swap item is a minimal pair in which a noun is swapped for one whose typical properties are incompatible with the property attributed to it, as in *A cucumber is green* vs. *A dune is green*. The model is scored as correct when it assigns the higher pseudo-log-likelihood to the compatible option. The benchmark covers four property types (color, material, relative size, shape) across four syntactic frames (copular, attributive, existential, relative). Each line contributes two items, since each of the two sentences is also scored against the other's noun, for 7,416 items in total. Every item is generated from and keyed to my training corpus `bb24.train` so that it records each word's

further training.

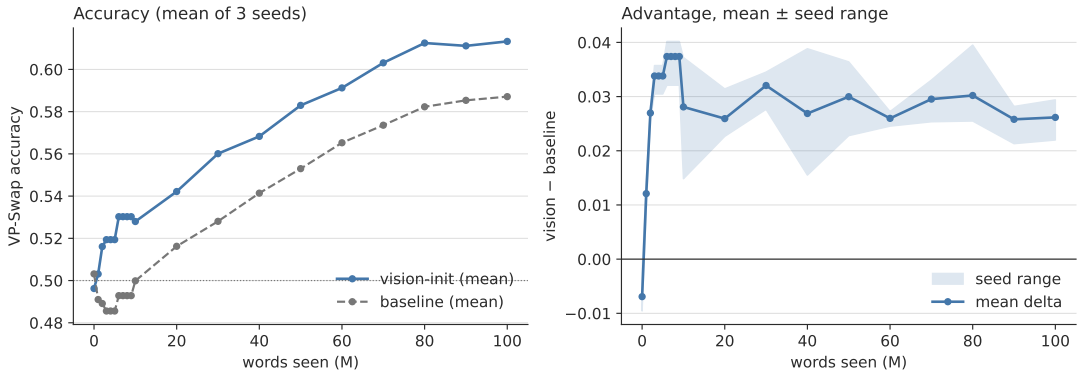


Figure 4: VP-Swap accuracy over training, SAM-seeded (75k) vs. its baseline, three random seeds. The visually seeded model leads from ∼1M words to the end of training.

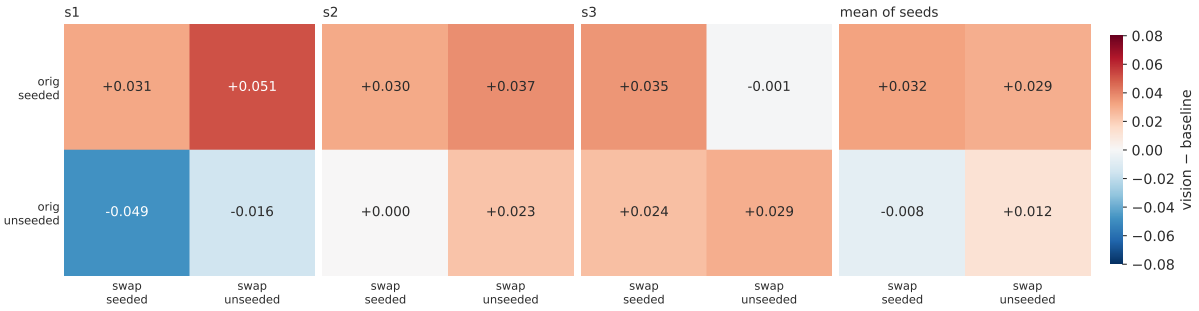


Figure 5: Seed-averaged VP-Swap delta by whether the original and swapped-in noun were seeded. The advantage is confined to items involving seeded words; the placebo cell (neither seeded) is ≈ 0.

corpus frequency and whether that word was visually seeded. Items are produced by an LLM generator and filtered by a separate LLM judge; frequency bins follow the logarithmic edges of LongTail-Swap, VP-Swap's sibling benchmark in the same suite, with all ten bins populated per property (construction details and the full funnel are in Appendix C). At initialization, untrained checkpoints score 0.49–0.51, confirming the probe is unbiased. I release my version of VP-Swap, along with all code and models.

5.2 Results

VP-Swap makes the initialization effect more visible and more explorable. Averaged over the benchmark, the SAM-seeded model leads from very early in training – by 1M words in all three seeds – and keeps the advantage of roughly two to three points to the end (final delta $+2.2/+2.7/+2.9$ across seeds; McNemar $z = 4.37/5.44/5.79$, all significant; Figure 4). This is the same effect COMPS detected, revealed by the dedicated probe.

The effect is specific to seeded words. The benchmark's metadata lets me attribute the gain to individual words. Splitting items by whether the original and swapped-in nouns were seeded (Figure 5) shows that the advantage falls entirely on items with seeded words: the seeded-noun delta is $+0.034/+0.031/+0.030$ across random seeds, while the *placebo* pairs, where neither noun was seeded, score near zero ($+0.012$, $n = 310$). A frequency-binned DiD confirms this is not a frequency artifact: the seeded-minus-unseeded advantage is positive in six of the seven bins with enough unseeded items to measure, and at -0.002 in the seventh. The effect is therefore a targeted improvement, located precisely on the words the intervention aimed at.

What is not robust. An apparent *penalty* on unseeded words in a single run did not survive replication ($-0.040/+0.006/+0.026$ across seeds), so seeding does not harm other words. The gain holds across syntactic frames (attributive $+0.023$, existential $+0.031$, relative $+0.027$) and is near zero in the short copular frame (-0.001), which offers the least context for a property to be evaluated against. It is largest for color ($+0.035$) and concentrated in mid- and high-frequency words; for the rarest

words both models sit near chance, consistent with the coverage picture. Full per-property, per-frame, and per-bin tables are in Appendix E.

6 Synthetic Grounding

The evidence for the role of ostension so far can be seen as correlational: words that happened to be visually seeded given my corpora show the effect. A stronger test is to *intervene* – for instance, to ground words that were not grounded before and check whether the effect follows the intervention. If it is caused by grounding, then for newly grounded words we should see VP-Swap improvement, while words that were already seeded or remain unseeded should be unaffected.

Extending grounding synthetically. For 1,986 concrete words with no image support, I generate scene descriptions with an LLM, render each as images with a text-to-image model, detect the target object with an open-vocabulary detector, and pool the same SAM features used for real grounding inside the detected boxes; words the detector cannot find are dropped. This grounds 1,155 previously unseeded words (adding 737 seeded tokens), and I retrain the model from this extended initialization with three random seeds (75k-SAM-ext; pipeline details in Appendix D). As a result, VP-Swap items fall into three groups: those whose noun was *really* seeded (6,242 items, a control that should not change), those whose noun is now *synthetically* seeded (835 items, the treated group), and those still unseeded (339 items, a second control).

The effect follows the grounding. The prediction holds on all three groups (Figure 6). On the synthetically grounded words, 75k-SAM-ext improves over the SAM model in all three seeds ($+0.012/+0.020/+0.010$; mean $+0.014$, in contrast with the mean delta of -0.002 for these words in the original SAM seeding). The really-seeded control is untouched (mean -0.003), as it should be: those words were already grounded, so adding others does not change them. The advantage on the treated group emerges over training rather than at initialization, and COMPS remains positive under all random seeds ($+1.57/+1.16/+0.40$), confirming the extension does not disturb the original effect. The per-word benefit of synthetic grounding is smaller than that of real grounding by roughly half, which is unsurprising given that generated images and automatic detection are noisier than human region annotations. But its direction is unambiguous and has exactly the grounded words as its scope.

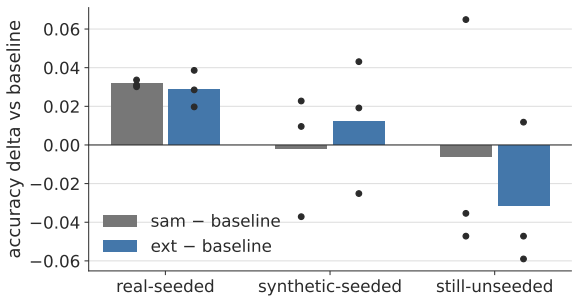


Figure 6: VP-Swap delta by grounding group. The synthetic extension (75k-SAM-ext) helps the *synthetically seeded* words – where the original SAM seeding had no effect – while leaving the really-seeded and still-unseeded groups essentially unchanged.

7 Mechanism: A Prior with a Lasting Imprint

How does an intervention applied only at initialization still shape a model after 100M words of training? I track each seeded embedding's cosine similarity to its initial value over training: it falls from 1.00 to about 0.15 by 100M words – the embeddings end up nearly orthogonal to where they began. This means that whatever is driving the effect of the visual initialization, it is not the survival of the original visual vectors. If the benefit came from staying close to the visual initialization, words that drifted less should benefit more. That is not the case: per-word drift is essentially uncorrelated with the change in that word's VP-Swap advantage ($r = -0.017$ over the 816 seeded words with at least three items).

The effect is relational. While the absolute positions of the seeded embeddings move almost entirely, the *relative* geometry among them is partly preserved. Measured by representational similarity analysis (RSA) (Kriegeskorte et al., 2008), the similarity structure of the seeded embeddings still correlates with that of their visual anchors at 100M words (RSA 0.31). The natural comparison is the baseline model: computed over the very same words, its embeddings show almost no such structure (RSA 0.10), confirming that a text-only model does not recover this visual geometry on its own – it is present only because those words were visually initialized. The correlation is stable over the second half of training. The visual prior thus

leaves a lasting imprint on how seeded words are arranged relative to one another, even if each word's individual position is overwritten by distributional learning.

8 The Limits of Ostension

So far the story has mostly been about concrete nouns, as suggested by the coverage analysis. But grounding data does not only contain concrete nouns. Captions also contain function words and abstract vocabulary – *the*, *every*, *three* – and these words, too, received visual initializations, from the average of the many scenes they appear in. Do those seeds matter? I show that ostension is important even for the meaning of these words, but this is an effect that existing benchmarks do not register.

Visual initialization for abstract vocabulary? One might expect a function word appearing in a lot of image contexts to receive a washed-out average vector. It does not appear to be the case: the centered magnitude of function-word seeds matches that of concrete words (curated function words 10.3 vs. concrete controls 10.3; both sets are listed in Appendix G), so their visual contexts are systematic enough to yield distinctive vectors. Moreover, the model retains the visual information to the end of training. The relational structure among function-word embeddings still reflects the visual anchor at 100M words (RSA 0.45; the baseline model, computed over the same words, shows none, RSA ≈ 0), and their absolute similarity to the anchor is if anything *higher* than for concrete controls (0.44 vs. 0.34).

Used by the objective, not rewarded by benchmarks. However, on the benchmarks, these retained seeds do nothing. Across three seeds, BLiMP phenomena whose minimal pairs differ in an *abstract* word average -0.8 points (vision $-$ baseline), while phenomena differing in a *concrete* word average $+0.8$; the correlation between a phenomenon's delta and the concreteness of its varying words is $r = +0.20$ over the 54 fast-subset phenomena. The structural competence these benchmarks test doesn't seem to be something a visual prior for abstract vocabulary can improve. But this is a fact about the benchmarks, not about whether the seeds are useful for language learning: measured directly on the pretraining objective, they are useful. Figure 7 reports held-out masked-LM loss by token class under identical masking: the vision-initialized model predicts masked function words better than the baseline in all three seeds (a mean cross-entropy reduction of 0.080 nats), with never-seeded tokens at ≈ 0. The retained visual structure is useful for the training objective.

There is one suggestive exception. The only abstract domain with consistent vision-related improvement is EWoK *number* ($+5.8$). Numerals are quite well visually seeded, and quantity is abstract content that nonetheless is visually manifest, so a visual prior plausibly carries information about it. The samples are small and I treat this as speculation.

Summing up, the scope of ostension in language learning is broader than one might think, and broader than what we are currently able to measure. The visual seeds are strong even for functional words that are usually thought of as lacking visual aspects of meaning; moreover, the model's training objective finds these visual properties useful. Existing evaluations mostly test abstract structural competence and are blind to these effects. What evaluation would register them is an open question.

9 Discussion

I showed that the effect of visual grounding is real, but barely visible in the aggregate BabyLM evaluation. I would like to draw the field's attention to this and potential other cases where intervention and evaluation are misaligned in scope.

I demonstrate the effect with a targeted, metadata-rich probe informed by training dynamics – items that are annotated with training frequency and seeding status, checked at chance at initialization. I suggest such probes, rather than averages alone, as an instrument for measuring interventions that touch a specific and predictable part of what a model knows. The null result from EgoBabyVLM's text-encoder ablation (Lin et al., 2026), from a different intervention, reinforces the point: whether grounding "helps" isn't enough without clarifying which parts of vocabulary it helps and under what measurements.

Mechanism analysis (§7) has consequences for deciding where to invest effort to magnify the effects of injection. Since the visual information survives as relational structure rather than as a retrievable anchor, interventions that periodically reinject visual embeddings during training are not likely to work; richer grounding at initialization

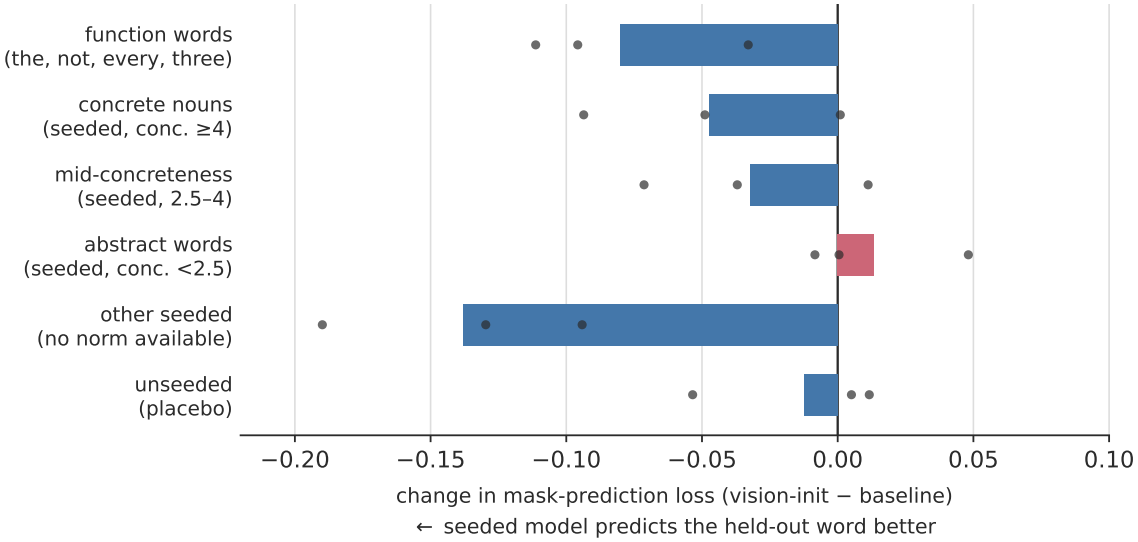


Figure 7: Change in held-out mask-prediction loss when words are vision-seeded (vision-init − baseline), by token class, under identical masking. Bars are the mean over three seeds, dots the individual seeds. Seeded classes become easier to predict, the never-seeded placebo class at ≈ 0.

is more promising. And since the visual seeds for function words are strong, retained, and used by the objective but not rewarded by any benchmark (§8), an obvious next step is evaluation designed to test what a visual prior could actually contribute to, see the suggestive EWoK *number* effect above.

Limitations

Most configurations are single runs; three model types were trained with three seeds each: the 75k baseline, its SAM-seeded counterpart, and the synthetic extension. My corpus is not the official 2026 strict-small distribution, which limits direct comparison to submissions trained on the standard data. VP-Swap is generated and filtered by LLMs and inherits their notion of typical object properties. Finally, several analyses have reduced scope: concreteness-band results cover only the ∼11k of 21k seeded tokens with Brysbaert norms, the curated function-word probe is 64 words, and the BLiMP concreteness gradient uses the fast-evaluation subset. All results are for a single masked-LM architecture.

Ethics Statement

This work carries low ethical risk. The models are trained from scratch on a public developmentally motivated corpus, and are not intended for deployment. Two components of my method rely on generative models. My version of the VP-Swap benchmark is constructed with an LLM generator and an LLM judge; the resulting items reflect those models' notions of typical object properties, which may encode cultural or distributional biases, and the released benchmark should be read as a diagnostic probe rather than a source of ground-truth world knowledge. My synthetic-grounding extension (§6) generates images with a text-to-image model and localizes objects with an open-vocabulary detector; these components can hallucinate or mis-detect, and the pipeline inherits whatever biases they carry. Because both pipelines feed only model initialization and a minimal-pair evaluation – never user-facing output – I judge these risks as contained, but I flag them so that downstream users of the released artifacts can account for them. All datasets and models used are publicly available and used consistently with their intended research purpose.

Acknowledgments

I thank Gabriele Sarti for the original idea behind the image-derived embeddings and for pointing me to the VP-Swap benchmark; Rick Nouwen for connecting visual initialization to Augustine's account of word learning; and Ece Takmaz for suggesting concreteness bins as a lens on the benchmarks, and for much other discussion. I am grateful to Lukas Edman and Jakub Dotlačil for helpful discussions, and to the anonymous reviewers for their comments. This work used the Dutch national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-18223.

References

Theodor Amariucai and Alex Warstadt. 2023. Acquiring linguistic knowledge from multimodal input. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 128–141, Singapore. Association for Computational Linguistics.

Andrew J. Anderson, Douwe Kiela, Stephen Clark, and Massimo Poesio. 2017. Visually grounded and textual semantic models differentially decode brain activity associated with concrete and abstract nouns. *Transactions of the Association for Computational Linguistics*, 5:17–30.

Augustine of Hippo. 1991. *Confessions*. Oxford University Press, Oxford. Translated by Henry Chadwick. Book I, section 8. Original work composed ca. 397–400 CE.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Lucas Georges Gabriel Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025. BabyLM turns 3: Call for papers for the 2025 BabyLM workshop. In *arXiv preprint arXiv:2502.10645*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092. ArXiv:2602.20092.

Lukas Edman, Lisa Bylinina, Faeze Ghorbanpour, and Alexander Fraser. 2024. Are BabyLMs second language learners? In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 166–173, Miami, FL, USA. Association for Computational Linguistics.

Martin N. Hebart, Adam H. Dickter, Alexis Kidder, Wan Y. Kwok, Anna Corriveau, Caitlin Van Wicklin, and Chris I. Baker. 2019. THINGS: A database of 1,854 object concepts and 26,107 naturalistic object images. *PLOS ONE*, 14(10):e0223792.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026.

Alina Klerings, Christian Bartelt, and Aaron Mueller. 2024. Developmentally plausible multimodal language models are highly modular. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 118–139, Miami, FL, USA. Association for Computational Linguistics.

Nikolaus Kriegeskorte, Marieke Mur, and Peter A. Bandettini. 2008. Representational similarity analysis – connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, 2:4.

Dongyan Lin, Phillip Rust, Angel Villar Corrales, Alvin W. M. Tan, Mahi Luthra, Charles Éric Saint-James, Rashel Moritz, Sheila Krogh-Jespersen, Vanessa Stark, Surya Parimi, Jiayi Shen, Youssef Benchekroun, Yosuke Higuchi, Martin Gleize, Tom Fizycki, Nicolas Hamilakis, Manel Khentout, Sho Tsuji, Balázs Kégl, and 3 others. 2026. EgoBabyVLM: Benchmarking cross-modal learning from naturalistic egocentric video data. *Preprint*, arXiv:2605.19130.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11–20.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. 2023. Scaling open-vocabulary object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 2928–2949.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana

Lazebnik. 2015. Flickr30k Entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2641–2649.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8748–8763.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. 2024. Adversarial diffusion distillation. In *European Conference on Computer Vision (ECCV)*.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. 2025. DINOv3. *arXiv preprint arXiv:2508.10104*.

Elizabeth S. Spelke and Katherine D. Kinzler. 2007. Core knowledge. *Developmental Science*, 10(1):89–96.

Ece Takmaz, Lisa Bylinina, and Jakub Dotlačil. 2025. Model merging to maintain language-only performance in developmentally plausible multimodal models. In *Proceedings of the First BabyLM Workshop*, pages 66–75. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Ludwig Wittgenstein. 1953. *Philosophical Investigations*. Blackwell, Oxford. Translated by G. E. M. Anscombe.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. 2022. iBOT: Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations (ICLR)*.

Chengxu Zhuang, Evelina Fedorenko, and Jacob Andreas. 2024. Visual grounding helps learn word meanings in low-data regimes. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1311–1329.

A Training Hyperparameters

All models use the same recipe (Table 1), following the configuration of Edman et al. (2024). Only the input-embedding initialization differs between a vision-init model and its baseline; every other hyperparameter, the data order, and the random seed are held fixed, so the two model pairs match as close as possible.

Hyperparameter	Value
Architecture	DeBERTa-v3-base (MLM)
Hidden size / layers / heads	768 / 12 / 12
Intermediate size	3072
Vocabulary (BPE)	50k / 75k / 100k
Learning rate	2×10^{-4}
LR schedule	cosine
Warmup	4000 steps
Weight decay	0.01
Optimizer	AdamW (β_1=0.9, β_2=0.95)
Effective batch size	256 (accum. 4)
Epochs	10
Context-length warmup	64→128 (from ep. 5)
Max sequence length	512

Table 1: Training hyperparameters, shared across all baselines and vision-init models.

B EWoK Subtask Deltas

Table 2 gives the per-subtask EWoK deltas (vision-init minus baseline, averaged over the nine encoder×vocabulary configurations, `best` revision). Subtasks whose content is visual or quantitative (*number*, *quantitative-properties*) show the largest and most consistent gains, while socially or spatially framed subtasks are flat or negative.

C VP-Swap Construction Details

VP-Swap contains 7,416 minimal-pair items across four properties (color, material, relative size, shape), built from `bb24.train` following the Visual-Property Swap protocol of Lin et al. (2026) with the deviations noted below. Each pair-file line encodes two sentences and the character offsets of the swapped nouns; metadata records each noun's corpus frequency, seeded status, property, and syntactic frame.

Generation funnel. Starting from corpus nouns with Brysbaert norms, items pass a sequence of automatic gates (Table 3). Sentences are generated by `claude-sonnet-4-6` (temperature 0.7); a concreteness/inanimacy gate and a 4× A/B plausibility judge use `claude-haiku-4-5` (temperature 0), and only items the judge accepts on all four

Subtask	Mean Δ	Positive
number	+6.53	6/9
active-passive	+4.81	7/9
quantitative-properties	+3.08	8/9
material-dynamics	+2.51	6/9
material	+2.15	6/9
direct	+1.02	8/9
negation	+0.76	5/9
concept swap	+0.59	6/9
physical-dynamics	+0.47	4/9
physical-relations	+0.45	4/9
social-relations	+0.32	5/9
antonym	+0.26	7/9
agent-properties	+0.14	5/9
physical-interactions	+0.10	5/9
indirect	+0.01	5/9
material-properties	+0.00	4/9
other	−0.16	3/9
social-interactions	−0.76	3/9
spatial-relations	−1.02	1/9
social-properties	−1.42	2/9
game	−4.44	0/9

Table 2: EWoK subtask deltas (vision-init − baseline), mean over 9 configurations. "Positive" counts configurations with a positive delta.

trials survive. Frequency bins follow the logarithmic edges $[1, 2, 4, \ldots, 512]$ used by LongTail-Swap (Lin et al., 2026), the frequency-controlled sibling of VP-Swap in the same benchmark suite; all ten bins are populated for every property.

Stage	Count
Corpus nouns w/ Brysbaert norms	11,921
Concreteness gate (≥ 4.0, noun)	5,006
Inanimacy gate (LLM)	1,669
seeded / unseeded	1,306 / 363
Sampled pairs per property	2,000
Well-formed generations	1,739–1,864
Attribution gate	1,160–1,259
4× A/B judge (all-correct)	878–992
Final (per property)	865/964/944/935

Table 3: VP-Swap construction funnel. Final counts are per property (color / material / relative-size / shape); each pair yields two minimal-pair items, for 7,416 total.

Syntactic frames. Sentences rotate over four frames – copular (*A femur is white*), attributive (*the white femur*), existential, and relative-clause – so the effect can be checked for robustness to sentence form; the judge rejects copular items at a somewhat higher rate.

Deviations from the upstream protocol. I follow the Visual-Property Swap task of Lin et al. (2026) (the `visual_property_swap` benchmark) with several deliberate changes. All items are generated from and frequency-binned against my own training corpus `bb24.train` rather than a fixed external word list, so that difficulty bins reflect the frequencies this model actually saw. The generator and judge models are my own (`claude-sonnet-4-6` and `claude-haiku-4-5`) with the four-trial A/B acceptance gate described above. Item selection is stricter: upstream uses an LLM yes/no physical-object judgment, whereas I gate on Brysbaert concreteness and part of speech, then add the inanimacy and attribution gates of the funnel above. Sentences rotate over fixed syntactic frames rather than being generated freely, and swap indices are validated at word boundaries, so that a substring match cannot corrupt a swap (*tub* inside *bathtub*). Most important for my analysis, every item is annotated with the seeded status of both of its nouns, which is what makes the difference-in-differences and placebo analyses of Section 5 possible. Finally, when a noun is substituted I adjust a preceding *a* or *an* to agree with the incoming noun, so that the distractor never differs from the original in grammaticality. Without this, swapping a vowel-initial noun under *a* yields an ungrammatical distractor that a model can reject on agreement alone. The correct article is taken from the generated sentences themselves where available, since they are a better authority than a spelling rule (*a unicycle*, *a uterus*), falling back to the vowel rule otherwise.

D Synthetic Grounding Pipeline Details

To isolate whether it's grounding that causes the effect (§6), I extend coverage to previously unseeded concrete words. From 1,986 concrete zero-support words I generate short scene descriptions with `claude-sonnet-4-6` (temperature 0.8), prompting it to place as many target words as fit naturally into a single scene. Each description is rendered into three images with SDXL-Turbo (Sauer et al., 2024) (`stabilityai/sdxl-turbo`, 2 inference steps, guidance scale 0). I then localize each target word with open-vocabulary detection (Minderer et al. 2023, `owlv2-base-patch16-ensemble`, score threshold 0.25). Words that the detector never finds in any of the images are dropped, so a word enters the extended seed set only if it can actually be pictured and found. SAM features are pooled inside the detected boxes through the same Stage-1 extraction code used for real grounding. This yields 1,155 newly grounded words (+737 seeded

tokens, 21,134 → 21,871), from which I train 75k-SAM-ext with three seeds.

Because a noun's "unseeded" status in VP-Swap is defined against the original seed set, extension induces a three-way item split: real seeded / synthetically seeded / still-unseeded (see Section 6). The still-unseeded group (339 items) serves as a within-benchmark control, and the real-seeded group (6,242 items) should be untouched by extension.

E Full Results Tables

This appendix collects the breakdowns referenced from the main text. The per-task official deltas and the EWoK subtask deltas appear in Figure 3 and Appendix B; the coverage figures are in the main text. Table 4 gives the COMPS subtask breakdown, Table 5 the VP-Swap results by property and by syntactic frame, and Table 6 the three-way synthetic-extension split.

COMPS subtask	Mean Δ	Positive
base (property knowledge)	+1.37	9/9
wugs (novel concept)	+3.65	9/9
wugs-dist-before	+0.15	4/9
wugs-dist-in-between	+0.04	4/9

Table 4: COMPS subtask deltas (vision-init − baseline), mean over 9 configurations. The gain is in the property-knowledge conditions and vanishes when distractors are added.

Group	n	vision	Δ
By property			
color	1730	0.639	+0.035
material	1928	0.646	+0.015
relative size	1888	0.575	+0.015
shape	1870	0.594	+0.024
By syntactic frame			
attributive	1890	0.658	+0.023
copular	1244	0.469	−0.001
existential	1680	0.652	+0.031
relative	2602	0.624	+0.027

Table 5: VP-Swap accuracy (SAM-seeded 75k) and delta vs. baseline, by property and by syntactic frame, seed `s1`. The effect is largest for color and holds in every frame but the short copular one, where both models sit slightly below chance.

F Grounding Scope: Additional Analyses

Supporting detail for Section 8. Table 7 reports representational-similarity (Kriegeskorte et al., 2008) between seeded embeddings and their visual anchors at the end of training (100M words), by concreteness band: the relational structure of the visual seeds survives training across all bands, and is retained most strongly for function words, even though, as the main text shows, no benchmark rewards it. The full function-word and concrete-control sets are listed in Appendix G.

Group	ext−base	sam−base	ext−sam
real seeded	+0.029	+0.032	−0.003
synthetic	+0.012	−0.002	+0.014
still unseeded	−0.031	−0.006	−0.026

Table 6: Synthetic-extension three-way split, VP-Swap deltas at 100M words (mean over three seeds). Synthetic grounding lifts exactly the synthetically grounded group (`ext−sam` column), where the original SAM seeding had no effect (`sam−base` = −0.002), while leaving the really-seeded group unchanged. The still-unseeded group is noisiest, being the smallest at 339 items.

Group	n	vision-init	baseline
band 1–2 (abstract)	731	0.188	−0.011
band 2–3	2186	0.189	0.001
band 3–4	3123	0.131	−0.004
band 4–5 (concrete)	5006	0.180	0.034
function words	64	0.446	−0.055
concrete controls	30	0.741	0.250

Table 7: RSA between seeded embeddings and their visual anchors at 100M words, by concreteness band, with the text-only baseline floor. Visual structure is retained across all bands (baseline ≈ 0), and among the abstract vocabulary is strongest for function words.

Table 8 gives the held-out MLM loss by token class (vision-init − baseline, three seeds and their mean). The objective is easier for seeded classes – most of all for function words (−0.080 nats, 3/3 seeds) – while the never-seeded class sits at the ≈ 0 placebo.

Class	n	s1	s2	s3	mean
function	1385	−.096	−.033	−.111	−.080
abstract (< 2.5)	1445	−.008	+.048	+.001	+.013
mid (2.5–4)	1327	+.011	−.037	−.071	−.032
concrete (≥ 4)	589	+.001	−.094	−.049	−.047
seeded-other	2446	−.130	−.094	−.190	−.138
unseeded (placebo)	1140	+.005	+.012	−.053	−.012

Table 8: Held-out MLM loss by token class (vision-init − baseline, in nats; negative = vision-init predicts the masked token better). Identical masks across models. Seeded classes improve; the unseeded class is ≈ 0.

The BLiMP concreteness gradient (correlation

of a phenomenon's vision-init delta with the concreteness of its varying words) is $r = +0.20$ over the 54 fast-subset phenomena.

G Word Lists for the Scope Analysis

The scope analysis of Section 8 and Appendix F uses two curated word sets. The *function-word* set (64 items) collects negation, quantifiers, determiners, connectives, numerals, and a few high-frequency deictics, words standardly treated as lacking perceptual meaning:

```
not, no, never, nothing, nobody, none,
neither, nor, every, each, all, some,
any, few, many, most, several, both,
either, or, and, but, if, unless,
because, although, whether, than, as,
only, even, also, too, very, quite, must,
might, may, could, should, would, will,
can, one, two, three, four, five, six,
seven, eight, nine, ten, first, second,
third, the, a, an, this, that, these,
those, there, it
```

The *concrete-control* set (30 items) is a set of high-concreteness, picturable common nouns used as the comparison group:

```
dog, cat, table, chair, apple, banana,
car, tree, house, ball, cup, door, book,
shoe, water, bread, bird, fish, horse,
bed, spoon, window, bottle, hat, box,
stone, flower, truck, boat, chicken
```

H The Entity-Tracking Scoring Artifact

During checkpoint-dynamics evaluation I initially observed a vision-init effect on the BabyLM entity-tracking task: a $+17$-point spike at $\sim$step 1000 (50k vocabulary, all three encoders). Closer analysis showed this to be a scoring artifact rather than entity-tracking competence, and I exclude entity tracking from interpretation throughout.

- **Untrained MLMs score $\sim$42%** on this task – far above the nominal chance level – because the pseudo-log-likelihood scoring interacts with the answer options' length and frequency priors.
- As training proceeds this inflated baseline *collapses*; vision-init models merely delay the collapse by $\sim$1000 steps rather than building competence.
- **Inverted subtask difficulty** confirms the diagnosis: 0-operation items score *lowest* and 5-operation items *highest* at every checkpoint, including the final one, which is the opposite of the true difficulty ordering.
- The spike is 50k-only; at 75k and 100k the corresponding peaks are just $+1.0$ and $+1.7$.

I conclude that entity-tracking scores of MLM-scored models should be treated as an artifact. An untrained-model check at step 0 is a cheap, effective diagnostic that would have caught this immediately; I now run it for every minimal-pair-style evaluation (e.g. VP-Swap sits at 0.49–0.51 at initialization, which is chance level).

How Far Is the Training Corpus from Real Infant Input? Auditing BabyLM Against Day-Long Home Recordings, and What an Age-of-Acquisition Curriculum Does Not Buy

Imrul Huda
University of Chicago
imrulhuda@uchicago.edu

Abstract

The BabyLM Challenge asks models to learn from developmentally plausible data, but how closely the standard corpus resembles the language infants actually hear has not been measured. We compare the Strict-Small corpus against SEEDLingS, a corpus of day-long home recordings from 44 infants with 358,300 hand-annotated noun tokens, over the 299 words the two share. Corpus word frequencies match real infant input only moderately (Spearman's $\rho = 0.42$; all correlations reported here are Spearman rank correlations), and real input frequency predicts human age of acquisition (AoA) far better ($\rho = -0.79$) than the training corpus's own frequencies do ($\rho = -0.48$).

Breaking the corpus down by source locates the gap. CHILDES tracks real infant input at $\rho = 0.55$ while the five other sources fall between 0.14 and 0.31, and the pooled corpus falls below CHILDES alone. The sources are not ordered by how speech-like they are: written text ranks above adult conversation. What separates them is child-directedness.

Against this we set a curriculum intervention. We order the corpus by the AoA of the words each sentence contains, derived from the MacArthur–Bates Communicative Development Inventory (CDI), train GPT-2 Small on it and on a content-matched shuffled baseline, across three seeds and three training budgets up to 100M words. The curriculum reliably produces a model whose acquisition order matches the human one ($\rho = 0.63$–0.78 against 0.09–0.22), but the curriculum encodes human AoA at $\rho = 0.96$ by construction and the alignment falls to $\rho \approx 0.3$ once presentation order is partialled out. It confers no advantage on BLiMP at any budget. Improving what a corpus contains may matter more than the order in which it is presented.

1 Introduction

A central goal of the BabyLM Challenge is to close the data-efficiency gap between human and machine language learners by using developmentally plausible training corpora (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). Most gains so far have come from changes to model architectures or training objectives rather than to the data itself (Charpentier et al., 2025). One exception is the use of variation sets—consecutive rephrasings common in child-directed speech—which improved some evaluation scores under certain conditions, though the effect varied across benchmarks (Haga et al., 2024). This motivates investigating other structural properties of child input.

We focus on the order in which words are learned. Age of acquisition (AoA) is a well-documented regularity in early word learning: children tend to acquire concrete, frequent words such as *mama*, *ball*, and *dog* before abstract words, and early vocabulary is strikingly consistent across children (Frank et al., 2017). The BabyLM 2026 evaluation suite includes an AoA task (Choshen et al., 2026) based on the method of Chang and Bergen (2022). This method tracks word surprisal across training checkpoints to estimate a model's implicit AoA and compares the resulting order with human norms from the MacArthur–Bates Communicative Development Inventory (CDI). The task therefore allows us to test directly whether a vocabulary curriculum shifts a model's learning trajectory toward the human order.

This paper makes two contributions. First, we audit the developmental plausibility of the Strict-Small corpus itself, comparing its lexical statistics against SEEDLingS, a corpus of real infant input (Bergelson and Aslin, 2017; Bergelson et al., 2019; Kalenkovich et al., 2025). The comparison

Proceedings of the Second BabyLM Workshop, pages 179–186
October 29, 2026 ©2026 Association for Computational Linguistics

requires no model and no training. We find that the corpus reproduces real input frequency only moderately, that real input frequency is a far better predictor of human AoA than any in-corpus measure, and, breaking the corpus down by source, that a single child-directed component carries most of what fidelity the corpus has while pooling the remaining sources reduces it.

Second, we test the intervention this audit bears on. We construct an AoA-staged curriculum, train it against a content-matched shuffled baseline across three seeds and three training budgets, and measure where each model's word-learning order falls relative to children's. The curriculum reliably reproduces human AoA order; we then show that most of this is the ordering reproducing itself, and that none of it reaches grammatical competence. Prior work using distributional measures has found that language model learning trajectories do not track children's (Ficarra et al., 2025); our result is compatible with that once the mechanical component is removed.

Read together, the two contributions suggest that the composition of a developmentally plausible corpus is a more consequential design choice than the order in which its contents are presented.

2 Related Work

Curriculum-learning approaches have appeared repeatedly across BabyLM iterations but have not produced consistently strong aggregate results (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). CLIMB (Diehl Martinez et al., 2023) explored vocabulary, data and objective curricula together and reported no reliable gain over a non-curriculum baseline; Salhan et al. (2024) applied cognitively motivated curricula across languages with similarly mixed results. ConcreteGPT (Capone et al., 2024) ordered training data according to lexical concreteness ratings (Brysbaert et al., 2014), but it did not derive its ordering from child acquisition data or organize the data into developmental stages. Variation sets, by contrast, improved some benchmarks—including BLiMP and GLUE, but not EWoK—depending on their proportion in the training data (Haga et al., 2024). Our curriculum instead uses lexical acquisition order derived directly from CDI production norms, and our finding that it confers no benchmark advantage is consistent with this broader pattern.

For evaluation, Chang and Bergen (2022) introduced a method for deriving model-internal AoA from surprisal trajectories across training checkpoints, and established that both model and human AoA are strongly conditioned by word frequency. The 2026 challenge adopts this method as an AoA task evaluated against CDI norms (Frank et al., 2017; Choshen et al., 2026). We apply the same core procedure using context sentences drawn from the training corpus itself rather than the evaluation pipeline's released context set (§6). Ficarra et al. (2025) take a distributional approach to the same question and report that model learning trajectories do not correlate with children's. Our contribution is not to revisit whether such alignment exists, but to ask what a corpus intervention can do about it, and to measure the corpus these interventions are applied to against a record of what infants actually hear.

We do not compare our models with the field-wide competition baseline (Charpentier and Samuel, 2024). Because our research question concerns the effect of word order within an otherwise fixed data-and-architecture setup, our control is a model with the same architecture and training content but a shuffled presentation order.

3 Method

3.1 AoA Curriculum and Baseline

We derive human AoA norms from the CDI production data distributed with the BabyLM 2026 evaluation package (Frank et al., 2017). The dataset provides, for each of 504 words, the proportion of children producing that word at each age from 16 to 30 months. We define a word's human AoA as the earliest age at which production reaches 50%. Words that never reach 50% are assigned an AoA of 31 months as a proxy for late acquisition, mirroring the 50%-of-range criterion later used for the model.

We score every sentence in the six Strict-Small source files—CHILDES, BNC-Spoken, Switchboard, Gutenberg, OpenSubtitles, and Simple Wikipedia; 1,104,106 sentences in total—using the mean AoA of the words covered by the CDI. Sentences containing no CDI words are assigned an AoA of 31.

We group the scored sentences into five stages based on mean sentence AoA (Table 1) and concatenate them in stage order. We repeat the earlier stages to up-weight simpler vocabulary, reflecting its prominence in early child-directed speech.

Stage	AoA range	Sentences	Reps.
1	$\leq$18 mo.	13,314	$\times$3
2	19–21 mo.	33,922	$\times$2
3	22–24 mo.	99,499	$\times$1
4	25–27 mo.	192,727	$\times$1
5	$\geq$28 mo.	764,644	$\times$1

Table 1: Curriculum stages based on mean sentence AoA. Stage 5 contains 69% of all sentences. Because the CDI covers only 504 words, many lexically simple sentences are assigned to this stage solely because they contain no CDI word (§6).

Within each stage, sentences are shuffled, so that stage membership is the only ordered variable and the curriculum does not also order the data by source corpus. The resulting curriculum contains 1,164,656 lines and 10,421,796 words, drawn from a unique corpus of exactly 10M words.

To isolate the effect of presentation order, the flat-shuffle baseline contains the identical multiset of lines, including the same sentences and per-stage repetitions, but is shuffled randomly using seed 1337. The two conditions therefore have identical word frequencies, vocabularies, and corpus sizes.

3.2 Model and Trajectory Evaluation

We train GPT-2 Small models with 124M parameters from scratch on each corpus with the standard GPT-2 tokenizer and AdamW. Training uses a cosine learning-rate schedule with 1% warmup, a learning rate of 5×10^{-4}, and an effective batch size of 32 sequences of 512 tokens. Each corpus is presented in its own fixed order, so that presentation order is the only respect in which the two conditions differ.[1] Every condition is trained with three random seeds (42, 43, 44), and we report the mean and standard deviation across seeds rather than a single run.

We report three training configurations. The first trains for one full pass over the Strict-Small curriculum (1,056 steps). The second repeats that curriculum for ten epochs (10,556 steps), and the third trains for one full pass over the Strict 100M curriculum (10,814 steps). The second and third hold total optimisation roughly constant while varying whether the additional compute buys repetition or new data. Word budgets are counted in words rather than tokens, using each corpus's measured tokens-per-word ratio, and checkpoints are saved at the word-exposure intervals the challenge requires (Choshen et al., 2026).

For each CDI word, we sample up to 20 corpus sentences containing that word. Of the 504 CDI words, 502 occur in at least three corpus sentences; *vagina* and *teddybear* are excluded. At each checkpoint, we compute the summed teacher-forced surprisal, in bits, of the word's subword tokens and average it across sentences to obtain a learning curve for each word.

Because a model's surprisal on a word falls as it learns that word, the checkpoint at which the surprisal drops marks the model's acquisition of it. We define a word's model AoA as the first checkpoint at which its surprisal falls halfway between its value at initialisation and its minimum value. This mirrors the CDI's 50%-of-range criterion, allowing the human and model measures to be defined analogously. We take the reference point at initialisation rather than at the first training checkpoint because a curriculum has, by then, already taught the words it front-loads: their surprisal has little room to fall further, so they never cross the threshold and are recorded as acquired last when they were in fact acquired first. An untrained reference makes the criterion well defined for every word regardless of when it is taught. Words that never cross the threshold are censored one checkpoint past the last. We report the Spearman correlation between model and human AoA, as well as a partial correlation controlling for log corpus frequency, a known confound in both measures. We treat $\alpha = 0.05$ as the significance threshold throughout.

3.3 SEEDLingS Ecological Validation

To assess how closely the training corpus reflects real infant experience, we compare it with SEEDLingS, a longitudinal corpus of day-long home recordings from 44 infants between 6 and 17 months of age (Bergelson and Aslin, 2017; Bergelson et al., 2019). Its publicly released noun-annotation dataset (Kalenkovich et al., 2025) contains 358,300 hand-annotated noun tokens with lemmatized `global_basic_level` labels. We use each lemma's token count as a measure of its frequency in real infant input. This comparison requires no model and no training: it relates two frequency distributions over the same vocabulary.

Matching these lemmas to the CDI words in

[1]The version of this work submitted for review used a training script that shuffled the corpus at load time, which neutralised the ordering manipulation in both conditions. All results reported here use the corrected configuration.

our analysis yields 299 words with at least 10 SEEDLingS tokens. Over this set, we calculate three Spearman correlations: the correlation between training-corpus frequency and real input frequency; the correlation between human AoA and real input frequency; and the correlation between human AoA and training-corpus frequency. All analyses use log frequencies. Corpus frequency is the number of sentences containing the word. Lines in the CHILDES portion begin with a speaker code, which the tokenizer would otherwise read as an ordinary word, so these prefixes are stripped before counting.

We then repeat the same three correlations separately for each of the six source corpora, which asks not whether the corpus as a whole diverges from real infant input but which of its components do. Because the sources differ in size, we report the number of matched words behind each correlation. We run this breakdown on both the Strict-Small corpus and the larger Strict corpus, where the same words are estimated from roughly ten times as much text.

4 Results

4.1 How Far Is the Corpus from Real Infant Input?

Table 2 presents the SEEDLingS comparison across the 299 matched CDI nouns. Two findings stand out.

First, word frequencies in the Strict-Small corpus correlate only moderately with frequencies in real infant input ($\rho = 0.42$). The corpus therefore approximates the overall scale of early linguistic experience but not its lexical distribution.

Second, real input frequency predicts human AoA much more strongly ($\rho = -0.79, p < 10^{-63}$) than the training corpus's own frequencies do for the same words ($\rho = -0.48$). The frequency signal most strongly associated with children's acquisition order, how often they hear a word, is therefore only weakly represented in the corpus from which a curriculum must be built.

4.2 Which Sources Diverge?

Table 3 repeats the comparison for each source corpus separately. Two further findings stand out.

First, CHILDES stands apart. Its word frequencies correlate with real infant input at $\rho = 0.55$ on the Strict corpus, against 0.27 ± 0.06 for the five remaining sources. The same ordering holds on Strict-Small, and CHILDES frequencies also predict human AoA far better ($\rho = -0.63$) than any other source. This is what one would expect of the only component that is itself child-directed speech.

Comparison ($n = 299$ nouns)	ρ	p
Corpus vs. real input freq.	0.42	5.8×10^{-14}
Human AoA vs. real input freq.	-0.79	4.6×10^{-64}
Human AoA vs. corpus freq.	-0.48	3.2×10^{-18}

Table 2: Ecological validation against real infant input. Values are Spearman correlations of log frequencies across CDI nouns matched to a SEEDLingS lemma with at least 10 tokens. The corpus only moderately reproduces real input frequency (row 1), while real input frequency is much more strongly associated with human AoA than corpus frequency is (rows 2 and 3).

	Strict-Small		Strict	
Source	n	ρ	n	ρ
CHILDES	298	0.48	298	0.55
Simple Wikipedia	261	0.29	298	0.29
OpenSubtitles	291	0.28	298	0.30
Gutenberg	264	0.25	293	0.31
BNC-Spoken	234	0.20	282	0.29
Switchboard	63	0.09	192	0.14
All pooled	299	0.42	299	0.45

Table 3: Correlation between each source corpus's word frequencies and real infant input frequency, computed separately per source. n is the number of CDI nouns clearing the matching threshold in that source. CHILDES is markedly closer to real infant input than any other source, and the pooled corpus falls below CHILDES alone.

Second, pooling lowers fidelity rather than raising it. The full corpus reproduces real input frequency at $\rho = 0.45$, below CHILDES alone at $\rho = 0.55$. Adding the non-child-directed sources therefore moves the corpus away from the distribution infants receive, even as it adds data.

The remaining five sources are not ordered by how speech-like they are. Written text (Gutenberg, $\rho = 0.31$) sits above adult conversation (BNC-Spoken, $\rho = 0.29$), and the lowest value belongs to a spoken corpus rather than a written one. What separates the sources is child-directedness, not spokenness. We note that Switchboard is the smallest source and clears the matching threshold for only 63 words on Strict-Small, so its value there is the least reliable in the table.

4.3 Does Ordering the Corpus Help?

Table 4 reports the curriculum and baseline models across the three training configurations. In every configuration the curriculum-trained model's acquisition order aligns closely with human AoA order, and the flat-shuffle baseline's does not. On one pass over Strict-Small the curriculum reaches $\rho = 0.77 \pm 0.02$ against 0.09 ± 0.03 for the baseline; the gap persists at ten epochs (0.63 ± 0.02 against 0.16 ± 0.02) and on the 100M corpus (0.78 ± 0.01 against 0.22 ± 0.02). Seed variability is small throughout, and the difference is significant at the level of training runs in all three configurations ($p < 10^{-5}$).

This alignment is, however, largely mechanical. The curriculum is constructed by ordering sentences by the AoA of the words they contain, so the position at which a word first appears in the training stream is itself almost perfectly correlated with its human AoA ($\rho = 0.96$). The model in turn acquires words roughly in the order it encounters them ($\rho = 0.78$ on one pass). Table 5 partials out corpus position: the residual alignment with human AoA falls to $\rho = 0.32$, and takes similar values at ten epochs (0.25) and on 100M words (0.29). Most of what the headline correlation measures is therefore the curriculum reproducing the ordering it encodes, though a smaller association survives that presentation order alone does not explain.

The ordering also does not transfer to grammatical competence. On BLiMP the curriculum and baseline models are indistinguishable in every configuration, with differences of -1.00, -0.10 and -0.19 points (all $p > 0.16$; Appendix A). The differences are uniformly small, uniformly negative, and shrink as training increases. We find no evidence that ordering the corpus by age of acquisition improves what the model learns about language, at any of the three budgets.

Corpus / schedule	Condition	ρ	partial
10M, 1 pass	Curriculum	0.77 ± 0.02	0.82
	Flat-shuffle	0.09 ± 0.03	-0.06
10M, 10 epochs	Curriculum	0.63 ± 0.02	0.58
	Flat-shuffle	0.16 ± 0.02	0.03
100M, 1 pass	Curriculum	0.78 ± 0.01	0.81
	Flat-shuffle	0.22 ± 0.02	0.09

Table 4: Spearman correlations between model AoA and human CDI AoA, as mean $\pm$ standard deviation over three random seeds. The partial column controls for log corpus frequency. In every configuration the curriculum-trained model aligns with human AoA order and the baseline does not.

	10M 1 pass	10M 10 ep.	100M 1 pass
Corpus position vs. human AoA	0.96	0.96	0.97
Model AoA vs. corpus position	0.78	0.53	0.79
Model AoA vs. human AoA	0.80	0.57	0.81
Residual, controlling position	0.32	0.25	0.29

Table 5: How much of the curriculum's AoA alignment is attributable to presentation order. The curriculum encodes human AoA almost perfectly by construction (row 1) and the model largely follows the order it is given (row 2). Controlling for corpus position leaves a smaller residual association (row 4). All values are Spearman correlations over seed-averaged model AoA; $p < 10^{-8}$ for every residual.

5 Discussion

Taken together, the two analyses point the same way. Ordering the corpus by age of acquisition reliably produces a model whose acquisition order matches the human one, but most of that match is the ordering reproducing itself, and none of it reaches grammatical competence. Meanwhile the corpus being ordered is only a moderate approximation of the language infants actually hear. The lever that was available to us turns out to matter less than the material it was applied to.

The sub-corpus breakdown locates the gap more precisely. It is not that the corpus is uniformly unlike infant input; it is that one component resembles it and the rest do not, and that pooling them together moves the whole further away. Nor is the difference explained by how speech-like a source is: written text ranks above adult conversation, and the lowest source is spoken. What distinguishes CHILDES is that it is addressed to children. This bears on how such corpora might be assembled. If the objective is a developmentally plausible corpus, the sampling weights across registers are a design parameter, and one that could be set against a measured distribution of real input rather than by availability.

We do not read this as an argument that child-directed speech should simply crowd out the other sources. Infants hear a great deal of speech not addressed to them, and day-long recordings capture that mixture as it actually occurs; this is precisely why SEEDLingS, rather than CHILDES alone, is the appropriate reference. The finding is not that

the corpus should contain more CHILDES, but that its mixture does not match the mixture infants receive, and that the mismatch is measurable.

Two caveats attach to the curriculum result. The first is that our curriculum is built from CDI norms and then evaluated against CDI norms, so a positive correlation is close to guaranteed; the position-controlled analysis is what separates the mechanical part from the remainder. The second is that the residual we do observe, at $\rho \approx 0.3$ across all three configurations, is not itself explained. It is stable under a tenfold change in either data or compute, which suggests it reflects something about the words rather than about the training budget, but we have not isolated what.

We therefore read the result as informative rather than encouraging. AoA curricula reproduce the order they encode and, on the evidence here, buy nothing beyond it. Improving the developmental fidelity of the training corpus itself may matter more than reordering its existing contents.

6 Limitations

The residual is unexplained. Partialling out corpus position leaves an association between model and human AoA of $\rho \approx 0.3$ in every configuration (Table 5). We do not know what produces it. It is stable under a tenfold change in either training data or compute, which argues against its being an artefact of training budget, but we have not established whether it reflects properties of the words themselves or a residual frequency effect that the position control does not absorb. The two controls we report, for frequency and for position, were applied separately rather than jointly.

Circular construction. The curriculum is built from CDI norms and evaluated against CDI norms, so some positive correlation is guaranteed by construction. This is why we report the position-controlled analysis alongside the raw correlation, and why we do not treat the headline value as evidence that the curriculum induces human-like learning. A design that ordered the corpus by a measure independent of the evaluation would test the underlying claim more directly.

Sparse and proxy norms. The CDI covers only 504 words. Consequently, any sentence containing none of these words is assigned to the latest AoA bin regardless of its actual complexity; such sentences constitute roughly 58% of the curriculum, and the majority of stage 5 (Table 1). Extending the vocabulary coverage with the reading-AoA norms of Kuperman et al. (2012) could produce more precise later stages, but this extension fell outside the scope of the present study.

Operationalization of model AoA. Our definition—the first checkpoint at which surprisal crosses the 50%-of-range threshold from an untrained reference—is one of several possible choices. It may be sensitive to the number of context sentences used, up to 20, and to the particular contexts selected, which differ from the official pipeline's fixed, released context set. We note that the choice of reference point is not innocuous: measuring the range from the first training checkpoint rather than from initialisation inverts the measure for precisely those words a curriculum front-loads (§3.2).

SEEDLingS matching. The ecological comparison is based on 299 CDI nouns matched by lemma to the SEEDLingS noun annotations (Kalenkovich et al., 2025). It therefore covers only nouns, the category annotated in the dataset, rather than the complete set of CDI words. Lemma-level token counts are also a coarse proxy for the frequency of a word in an individual child's input. Nevertheless, the analysis draws on real longitudinal recordings from 44 infants rather than on corpus-derived proxies.

Uneven source sizes. The per-source breakdown estimates each correlation from the text available in that source, and the sources differ by two orders of magnitude in size. Switchboard clears the matching threshold for only 63 of the 299 words on Strict-Small, so its value there should be read as the least reliable in Table 3; the larger Strict corpus mitigates but does not remove this.

Scope. All experiments are in English. The curriculum results cover the Strict-Small and Strict corpora; we make no claims about the Multilingual track, and we do not compare against the official competition baseline, which we did not run.

Ethical Considerations

All training corpora come from the officially distributed BabyLM Strict-Small release. The CDI norms come from Wordbank's public developmental data (Frank et al., 2017), and the SEEDLingS

noun annotations are available as a public, de-identified, CC-BY release (Kalenkovich et al., 2025). No new human-subject data were collected, and we did not access the raw recordings. We train and release only small, 124M-parameter models and make no claims regarding deployment.

Code and data availability. All corpus-construction, training and analysis code, together with the derived result files underlying every reported correlation, is available at `https://github.com/imrulhuda/babylm-aoa-audit`. The SEEDLingS noun annotations are obtained from the CC-BY release of Kalenkovich et al. (2025); the BabyLM corpora are used as distributed by the challenge organisers.

Acknowledgments

We thank the anonymous reviewers for comments that materially reshaped this paper, and the Bergelson Lab for releasing the SEEDLingS noun annotations that make the corpus comparison possible.

References

Elika Bergelson, Andrei Amatuni, Shannon Dailey, Sharath Koorathota, and Shaelise Tor. 2019. Day by day, hour by hour: Naturalistic language input to infants. *Developmental Science*, 22(1):e12715.

Elika Bergelson and Richard N. Aslin. 2017. Nature and origins of the lexicon in 6-mo-olds. *Proceedings of the National Academy of Sciences*, 114(49):12916–12921.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, 46(3):904–911.

Luca Capone, Alessandro Bondielli, and Alessandro Lenci. 2024. ConcreteGPT: A baby GPT-2 based on lexical concreteness and curriculum learning. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 189–196. Association for Computational Linguistics.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? In *Proceedings of the 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Richard Diehl Martinez, Zébulon Goriely, Hope McGovern, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2023. CLIMB – curriculum learning for infant-inspired model building. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 112–127. Association for Computational Linguistics.

Filippo Ficarra, Ryan Cotterell, and Alex Warstadt. 2025. A distributional perspective on word learning in neural language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11184–11207, Albuquerque, New Mexico. Association for Computational Linguistics.

Michael C. Frank, Mika Braginsky, Daniel Yurovsky, and Virginia A. Marchman. 2017. Wordbank: An open repository for developmental vocabulary data. *Journal of Child Language*, 44(3):677–694.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM challenge: Exploring the effect of variation sets on language model training efficiency. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 252–261. Association for Computational Linguistics.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21. Association for Computational Linguistics.

Evgenii Kalenkovich, Sharath Koorathota, Shaelise Tor, Andrei Amatuni, Shannon Egan-Dailey, Charlotte Moore, Catherine Laing, Hallie Garrison, Gladys Baudet, Federica Bulgarelli, Sarp Uner, Lillianna Righter, and Elika Bergelson. 2025. A year

of nouns from English-learning infants' daily lives: The SEEDLingS-Nouns dataset. *Behavior Research Methods*, 57(11):298.

Victor Kuperman, Hans Stadthagen-Gonzalez, and Marc Brysbaert. 2012. Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods*, 44:978–990.

Suchir Salhan, Richard Diehl Martinez, Zébulon Goriely, and Paula Buttery. 2024. Less is more: Pre-training cross-lingual small-scale language models with cognitively-plausible curriculum learning strategies. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 174–188. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34. Association for Computational Linguistics.

A Zero-Shot Linguistic Evaluation

Table 6 reports filtered BLiMP accuracy for every model discussed in §4.3: two conditions, three random seeds, and three training configurations. We report BLiMP alone; the remaining zero-shot tasks in the challenge suite were not rerun for these models.

Two observations follow. First, accuracy rises with training as expected, from 60 on a single pass over Strict-Small to 69 at ten epochs and 75 on the 100M corpus, against a chance level of 50. The models underlying the curriculum comparison are therefore trained well past the point at which BLiMP is uninformative.[2] Second, the curriculum and baseline models are indistinguishable at every budget. The differences are -1.00, -0.10 and -0.19 points, none significant at $n = 3$ per condition ($p = 0.169, 0.850, 0.447$), and they narrow as training increases.

Corpus / sched.	Condition	s42	s43	s44	mean
10M, 1 pass	Curriculum	59.29	60.15	60.84	60.09
	Flat-shuffle	61.42	61.54	60.31	61.09
10M, 10 ep.	Curriculum	68.86	69.04	67.90	68.60
	Flat-shuffle	69.34	68.43	68.32	68.70
100M, 1 pass	Curriculum	75.40	75.10	74.97	75.16
	Flat-shuffle	75.11	75.70	75.22	75.34

Table 6: Filtered BLiMP accuracy (%) at the final checkpoint, by condition, seed and training configuration. Chance is 50. Accuracy increases with training budget, and the curriculum confers no advantage over the shuffled baseline in any configuration.

[2]For reference, the version of this work submitted for review reported 55.99 on a single, shorter run; see §3.2.

Linguistically Inspired Multilingual Pre-Pretraining

Masato Mita[1] **Max Müller-Eberstein**[1,2] **Misaki Yamaguchi**[1] **Yohei Oseki**[1]

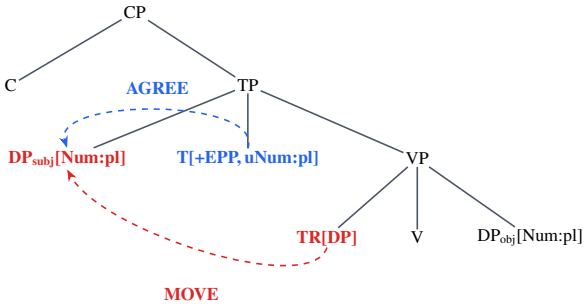
[1]The University of Tokyo [2]IT University of Copenhagen
{mita, iijimax, yamami, oseki}@g.ecc.u-tokyo.ac.jp

Abstract

Encoding Universal Grammar principles (UG) during pre-pretraining (PPT) has been shown to produce inductive biases in language models, which facilitate natural language acquisition. While UG *principles* should be language-agnostic, each language follows different typological *parameters*. To empirically validate this theory, we operationalize three levels of typological flexibility via PPT: non-typological k-shuffle Dyck, as well as MP-STRUCT, which encodes the Minimalist Program operations MERGE, AGREE, MOVE following fixed typological parameters. Additionally, we introduce MP-STRUCT BCC (BCC), which parameterizes typological features following the Borer–Chomsky Conjecture to allow for a more typologically flexible parametrization. Through a multilingual, subtask-level evaluation across 207 grammatical tasks for English, Dutch, and Chinese, we find that principles-driven PPT significantly improves average accuracy, but selectively degrades language-specific phenomena such as wh-island constructions. In contrast, structure-only PPT, as well as typologically flexible PPT eliminate wh-island degradation while preserving overall gains, demonstrating that typological parameterization offers a path toward multilingual syntactic priors. Exploratory model-internal analyses are consistent with these findings at the levels of output probabilities, attention structure, as well as cross-lingual representation alignment.

1 Introduction

Language models (LMs) remain far less data-efficient than human learners, particularly for syntactic phenomena that depend on hierarchical rather than linear structure (Warstadt et al., 2023). Children converge on such knowledge from comparatively little linguistic input, whereas LMs typically require orders of magnitude more text to reach comparable performance (Warstadt and Bowman, 2022). Under the Universal Grammar (UG) hypothesis, human language acquisition is guided by an innate bias—the Language Acquisition Device (LAD; Chomsky, 1965), grounded in UG—that constrains the learner's hypothesis space toward natural-language-like structure from the outset. If a comparable structural bias could be injected into LMs prior to natural language training, the acquisition of more complex, hierarchical syntactic structures should be facilitated.



MP-STRUCT (linearized):
[CP [C] [TP [DP[Num:pl] [D][N]]
[T(+EPP,uNum:pl)]
[VP V [DP[Num:pl][D][N]] [TR[DP]]]]]

Figure 1: Example MP-STRUCT derivation, reproduced from Mita et al. (2026). **MERGE** builds the hierarchical vP/TP/CP skeleton; **AGREE** matches the subject's Num feature to T (uNum ← Num:pl); **MOVE** raises the subject DP to Spec-TP (EPP), leaving a trace TR[DP] at its base position. In this work, we parametrize these operations to be more typologically flexible.

Pre-pretraining (PPT) instantiates this strategy: LMs are first trained on formal languages that encode abstract hierarchical structure (e.g., "{ (} [ )]"), before proceeding to standard natural language pretraining, where genuine syntactic structure must be acquired—injecting a structural inductive bias for subsequent learning (Papadimitriou and Jurafsky, 2023; Hu et al., 2025). Extending prior heuristic approaches, MP-STRUCT (Mita et al., 2026) provides a PPT paradigm, which incorporates the fundamental UG-operations: MERGE

Proceedings of the Second BabyLM Workshop, pages 187–198
October 29, 2026 ©2026 Association for Computational Linguistics

(hierarchical structure building), AGREE (feature checking), and MOVE (displacement) as shown in Figure 1. This added structure has been shown to improve the efficiency of English language acquisition. However, prior work has left open the questions of (1) which specific grammatical phenomena PPT improves or degrades, and (2) whether PPT is effective on languages other than English.

By definition, UG should be language-universal, and its inductive bias should accordingly support the acquisition of any natural language. Since MP-STRUCT is designed to operationalize UG's core operations, its inductive bias should likewise extend to languages beyond English. However, the current implementation of MP-STRUCT contains English-specific design choices, such as obligatory wh-fronting (e.g., Chinese instead allows wh-in-situ), and head-initial word order (e.g., compared to Dutch's underlying SOV). Such typological differences within UG are characterized by the Principles and Parameters (P&P) framework (Chomsky, 1981) as *parameters*, which delimit the range of variation possible across languages, while *principles* must be satisfied by all languages.

In this work, we provide, to the best of our knowledge, the first subtask-level study on PPT's effects across typologically different languages, complementing prior multilingual PPT work (Papadimitriou and Jurafsky, 2020, 2023) with a fine-grained phenomenon-level analysis for additional languages, as well as methods for making PPT itself more typologically flexible following the P&P framework. Towards these goals, we provide the following contributions:

1. **Empirically differentiating P&P.** Using PPT to operationalize different levels of *principles* and *parameters* (§3), we identify that a balance between the two is crucial to achieve performance gains across BLiMP (Warstadt et al., 2020) subtasks over the non-PPT baseline, while avoiding language-specific biases, such as wh-island degradation (-10.3 pp) driven by English-centric parametrization (§4).

2. **Typologically flexible PPT.** By making the *parameters* within P&P language-adaptable and typologically coherent, we introduce the BCC framework for PPT (§3.3), which eliminates wh-island degradation while maintaining significant overall improvements, better balancing cross-lingual performance.

3. **Exploratory analysis of model-internal changes driven by PPT.** Model-internal differences suggest how PPT induces structurally-oriented attention and reshapes cross-lingual representations. BCC exhibits balanced cross-lingual convergence, whereas fixed parametrization over-converges toward English, and a lack of parametrization produces low language differentiation (§5).

2 Related Work

2.1 Universal Grammar and Minimalism

Under the UG/LAD hypothesis (Chomsky, 1965), human language acquisition is constrained by an innate linguistic bias, grounded in UG. P&P theory (Chomsky, 1981) decomposes UG into two kinds of constraints in order to characterize typological variation: *principles*, which every language must satisfy without exception, and *parameters*, which delimit the range of variation possible across languages, with each language fixing its own values.

We treat the core structure-building operations of the Minimalist Program (Chomsky, 1995)—MERGE, AGREE, and MOVE—as instantiating the principles of P&P, since every language is assumed to build hierarchical phrases, check feature agreement, and displace constituents in some form, without exception.[1] Parameters, in contrast, instantiate typological variation along dimensions such as wh-movement, head directionality, and richness of agreement, each admitting a small set of possible values that a given language fixes. The Borer–Chomsky Conjecture (BCC; Borer, 1984; Chomsky, 2001) further holds that cross-linguistic variation reduces to the feature composition of functional heads, providing the theoretical foundation for our parameterization approach (§3.3).

2.2 Pre-Pretraining

Pre-Pretraining (PPT; Papadimitriou and Jurafsky, 2023) injects a structural inductive bias into LMs through short-term pretraining on formal languages. Specifically, k-shuffle Dyck, a context-sensitive relaxation of k-Dyck, captures complex dependency

[1]These operations reformulate, rather than replicate, the specific principles proposed in earlier P&P work (Chomsky, 1981); we adopt this operational mapping for computational tractability rather than as a claim about their historical equivalence. Note also that the *availability* of an operation (e.g., AGREE) is considered universal even where its morphological realization is minimal or absent in a given language (e.g., Chinese); we return to this distinction in §3.2.

	eng	nld	zho	MS
Wh-mvt	oblig.	oblig.	in-situ	**oblig.**
Word order	SVO	V2/SOV	SVO	**head-init.**
Agreement	limited	rich	none	**limited**

Table 1: Target languages and typological contrasts with MP-STRUCT (MS).

Head	Feature	Values	Langs
v, V	complement dir.	init./final	en,zh/nl
C	[+wh] strength	strong/weak	en,nl/zh
C	V-attraction	act./inact.	nl/en,zh
T	φ-features	rich/lim./$\emptyset$	nl/en/zh
Top	EPP	act./inact.	zh/en,nl

Table 2: Typological parameters in BCC.

structures within the computational limits of Transformers, providing an effective structural bias (Hu et al., 2025). MP-STRUCT (Mita et al., 2026) instantiates the principles identified above (MERGE, AGREE, MOVE; §3.2) as bracketed formal languages (see Figure 1). Prior work shows that attention head subnetworks acquired during PPT persist through pretraining (Hu et al., 2025) and that low dependency resolution ambiguity determines PPT effectiveness (Mita et al., 2026). To extend PPT more effectively to multilingual settings, this work studies the gap between PPT without any UG principles (i.e., Dyck languages), and PPT with fixed English-type UG parameters (i.e., MP-STRUCT).

3 Methods

We evaluate three PPT paradigms with increasing cross-lingual flexibility: k-shuffle Dyck, MP-STRUCT, and BCC—across English, Dutch, and Chinese (Table 1). These languages differ in wh-movement (obligatory fronting in English/Dutch, Chomsky, 1977, vs. wh-in-situ in Chinese, Huang, 1982) and word order (SVO in English/Chinese vs. V2/SOV in Dutch, den Besten, 1983).

3.1 k-shuffle Dyck: Inducing General Language Structure

k-shuffle Dyck (SD; Hu et al., 2025) encodes interleaved bracket structures (e.g., "{ (} [ )]") with cross-serial dependencies, providing a language-agnostic structural bias. It serves as a structure-only control to validate whether any performance gains or losses on specific grammatical phenomena are a result of UG instantiation or a general characteristic of PPT. Following the main experimental settings of Hu et al. (2025), we adopt $k = 64$ with shuffling, which provides a more robust structural bias than non-shuffled, strictly nested alternatives, and a high-enough value of k as to better approximate the cross-serial dependencies found in natural language.

Algorithm 1 BCC Data Generation

Notation: $\mathcal{L}$: lexicon, θ: typological parameters, c: clause syntax tree, S: output sequence, $\oplus$: string concatenation, L_{max}: sequence length limit.

Input: lexicon $\mathcal{L}$, target length L_{max}
Output: token sequence S
$S \leftarrow \emptyset$
while $|S| < L_{\text{max}}$ **do**
  Step 1: Clause-level Parameter Sampling
  $\theta \leftarrow \text{SAMPLEPROFILE}(\{\text{en, nl, zh}\})$
  // Determines the cross-linguistic syntactic variation for the current clause
  Step 2: Clause Generation and Linearization
  $c \leftarrow \text{GENERATECLAUSE}(\mathcal{L}, \theta)$
  // Applies MERGE, AGREE, MOVE *following parameters* θ
  $c_{\text{lin}} \leftarrow \text{LINEARIZE}(c)$
  Step 3: Concatenation (Code-switching Simulation)
  $S \leftarrow S \oplus c_{\text{lin}}$
  // Appends the linearized clause to simulate typological code-switching
end while
return S

3.2 MP-STRUCT: Operationalizing Universal Grammar Principles

MP-STRUCT (MS; Mita et al., 2026) generates syntactic derivations encoding Minimalist Program operations (Figure 1): non-terminal labels (CP, TP, vP), functional heads (C, T, v), and φ-features are combined using the principle operations MERGE (hierarchical structure building; Chomsky, 1995), AGREE (feature checking; Chomsky, 2000), and MOVE (displacement; Chomsky, 1981, 1995), as well as movement traces (TRACE). The resulting derivations are linearized into bracketed sequences (e.g., [CP MERGE [C [+wh]] [TP ...]]) with language-independent abstract lexical tokens, so the LM learns syntactic operations before exposure to natural language. Beyond the principle operations (MERGE, AGREE, MOVE), the design includes *parameters* that govern how these operations are applied. Following its original work, we fix these parameters to follow English-like syntactic rules: (**wh-movement**) wh-elements obligatorily move to Spec-CP (English-type fronting); wh-in-situ is never generated; (**head directional-**

ity) always head-initial; V2 and head-final orders are not generated; (**agreement features**) partially encoded; rich agreement systems and the absence of agreement are not represented.

3.3 BCC: Parameterizing Typological Features

To address MS's English-centric parameter settings, we propose MP-STRUCT BCC (BCC), which introduces explicit typological parameters following the Borer–Chomsky Conjecture (Borer, 1984; Chomsky, 2001). For each generated clause, functional head features are sampled from typological profiles (Table 2). We introduce five parameters: First, **complement directionality** on the v and V heads determines whether verbs precede (English, Chinese) or follow their objects (Dutch). Second, **[+wh] strength** on the C head dictates whether wh-movement is obligatory (English, Dutch) or whether wh-in-situ is allowed (Chinese). Third, **V-attraction** on the C head controls verb-second (V2) word order, which is active in Dutch but inactive in English and Chinese. Fourth, **φ-features** on the T head specify the richness of morphosyntactic agreement, ranging from rich (Dutch) to limited (English) or absent (Chinese). Finally, **EPP** on the Top head regulates topic-prominence, which is active in Chinese but inactive in English and Dutch.

Each PPT sequence concatenates multiple independent clauses whose typological parameters are dynamically sampled from plausible language profiles (English, Dutch, Chinese; Algorithm 1). Since lexical items and structural operations are identical to MS, any observed differences are attributable solely to typological parameterization.

4 Experiments

4.1 Setup

Training. Following the BabyLM 2026 Multilingual track (Jumelet et al., 2025; Choshen et al., 2026), we train GPT-2-small (Radford et al., 2019; 124M parameters) on the official `en_nld_zho_equal` pre-training data (100M words $\times$ 10 epochs). PPT conditions observe 500 initial steps of synthetic language pre-pretraining (lr = 5×10^{-4}, warmup = 100). Four conditions are tested with three random seeds each: **Baseline (BL)** without PPT; **Shuffle Dyck (SD)** with k-shuffle Dyck ($k = 64$) PPT (§3.1); **MP-STRUCT (MS)** with English-type parameter-fixed PPT (§3.2); **BCC** with typological parameterization (§3.3). All conditions share identical model initializations, natural-language data ordering, and training schedule per seed; only the PPT phase differs.

	eng	nld	zho	avg
BL	.700±.001	.790±.005	.752±.004	.747±.003
SD	.707±.003	.792±.001	**.762**±.002	.754±.002*
MS	**.710**±.005	.794±.005	.759±.008	**.755**±.006*
BCC	.708±.007	**.795**±.003	.760±.004**	.754±.005*

Table 3: Group-level BLiMP accuracy across all 4 conditions (mean ± std, 3 seeds). Bold: best per column. * $p < .05$; ** $p < .01$ vs BL (paired t, $n = 3$). Paired mean differences vs BL: SD +0.6 pp; MS +0.7 pp; BCC +0.7 pp (95% CIs overlap but all exclude zero).

Evaluation. All models are evaluated on BabyLM 2026's multilingual BLiMP setup: 67 English subtasks from BabyLM-BLiMP (Warstadt et al., 2020); 22 Dutch subtasks from BLiMP-NL (Suijkerbuijk et al., 2025); and 118 Chinese subtasks from ZhoBLiMP (Liu et al., 2026)—totaling 207 evaluated subtasks. Each subtask uses minimal pairs to test grammaticality judgments. Benchmark accuracy is reported as mean ± stddev (3 seeds), with paired t-tests and 95% CIs.[2] should be interpreted cautiously as descriptive summaries rather than population-level inferential evidence. For win/loss rates across subtask-level results, each subtask's 3-seed mean difference Δ_{mean} is classified as Win ($\Delta \geq 2$ pp), Loss ($\Delta \leq -2$ pp), or Tie, with 95% binomial CIs (Wilson score). The 2 pp threshold exceeds typical cross-seed variance (mean within-condition std ≈ 0.4 pp), and we find conclusions to remain stable across ± 1 to ± 5 pp thresholds (Table 7, Appendix A).

4.2 Benchmark-level Results

All PPT conditions significantly improve the 3-language average over BL (Table 3). MS shows the largest English improvement, while SD and BCC show stronger gains on Chinese. BCC is **not strictly superior to MS** ($p = .217$ for the BCC–MS difference), with each condition winning on different subtasks.

4.3 Subtask-level Results

Roughly half of subtasks show no meaningful change (Table 4). Among those with $|\Delta| \geq 2$ pp,

[2] Due to the sample size $n = 3$, all effect sizes and significance values

	#Sub	W	L	T	W/(W+L)
SD vs BL					
eng	67	18	10	39	64.3%
nld	22	2	3	17	40.0%
zho	118	37	23	58	61.7%
Total	**207**	**57**	**36**	**114**	**61.3%**
MS vs BL					
eng	67	21	5	41	80.8%
nld	22	2	2	18	50.0%
zho	118	36	26	56	58.1%
Total	**207**	**59**	**33**	**115**	**64.1%**
BCC vs BL					
eng	67	17	8	42	68.0%
nld	22	4	2	16	66.7%
zho	118	35	28	55	55.6%
Total	**207**	**56**	**38**	**113**	**59.6%**

Table 4: Win/Loss analysis ($|\Delta_{\text{mean}}| \geq 2$ pp, 3-seed means). SD and MS both improve overall, but with different per-language profiles. BCC shows the most balanced per-language win rates. 95% CIs: SD [51.1–70.6%]; MS [53.9–73.2%]; BCC [49.5–68.9%].

all three PPT conditions favor wins. PPT **acts selectively on specific phenomena**: MS achieves the highest English win rate (80.8%) but the lowest Dutch rate (50.0%), hinting at an English-specific bias outlined below. BCC shows the most balanced performance profile across languages (66.7% median vs. 61.7% for BL and 58.1% for MS), consistent with its typologically diverse parametrization.

Language-bias in MS. Larger improvements involve language-specific structural phenomena: in Chinese, BEI construction and quantifier scope; in English, `wh_obj_gap` and `adjunct_island` (Table 5). All PPT conditions improve on these subtasks, with SD showing the largest `npi_renhe` gain (+22.1 pp), thus reflecting the structural inductive bias operating universally.

However, a striking dissociation emerges in wh-related subtasks (Table 6). MS improves accuracy where wh-movement occurs but degrades `wh_island` substantially. Critically, **SD shows no wh-island degradation**. This dissociation suggests that the degradation is not caused by PPT *per se*, but is consistent with MP-STRUCT's obligatory wh-fronting parameter (this being one parameter that differs from MS to BCC; §6). A parallel pattern appears in Dutch, where MS degrades `wh_movement_restrictions` while SD shows no change (Table 8), though the Dutch subtask differs in item construction from the English `wh_island`, so this cross-lingual parallel is suggestive rather than exact. Chinese question-formation subtasks

Subtask	Lang	SD Δ	MS Δ	BCC Δ
Top improvements				
BEI_deletion	zho	+11.6	+28.4	+28.9
superlative_quant_2	zho	+16.2	+26.3	+12.3
npi_renhe_wh_subj	zho	**+22.1**	+8.3	+17.8
question_V_not_VP_1	zho	+9.0	+8.7	**+15.4**
Key regressions				
wh_island	eng	−0.3	**−10.3**	+0.9
q_daodi_A_not_A	zho	−11.4	−17.7	−5.8
renhe_no_sup_neg	zho	+4.2	+14.8	−10.4

Table 5: Selected top changes vs. BL (Δ in pp). Subtasks are selected post-hoc by largest $|\Delta|$ across any condition (top improvements) and by relevance to wh-phenomena (key regressions); no a priori selection was applied. SD shows the largest npi_renhe improvement (+22.1) but no wh_island degradation (−0.3), consistent with parameter-free structural bias.

Subtask	Structure	BL	SD Δ	MS Δ	BCC Δ
wh_obj_gap	wh-mvt (gap)	.708	+6.7	+3.5	**+7.5**
adjunct_island	adjunct extr.	.752	+2.6	**+5.7**	+4.7
wh_island	**wh prohibited**	.672	−0.3	**−10.3**	**+0.9**
wh_vs_that	wh vs. that	.251	+4.9	+4.5	+1.3

Table 6: Wh-related subtask effects across all conditions (Δ from BL in pp). SD shows no wh-island degradation (−0.3); MS severely degrades where wh-movement is prohibited (−10.3); BCC successfully eliminates this degradation (+0.9).

(31 subtasks including A-not-A, *nandao*, topicalization) show no systematic MS degradation, consistent with wh-in-situ being unaffected by the obligatory-fronting bias.

Table 8 (Appendix B) summarizes this cross-lingual pattern. Structural bias improves performance across all languages for both SD and MS (UG *principles*), whereas the `MOVE`-derived bias specific to MS is skewed toward English (*parameters*). Within the tested typological space, the P&P distinction thus maps onto computational PPT.

BCC eliminates wh-island degradation (Table 6) while maintaining overall gains (Table 3). SD avoids degradation by not encoding wh-movement; BCC avoids it by encoding diverse typological settings. BCC also yields the largest `wh_obj_gap` improvement among the three PPT conditions, suggesting that diverse movement patterns promote a more general wh-gap detection mechanism. However, BCC, MS, and SD produce different subtask-level trade-offs, so parameterization alone does not yield a strictly superior design.

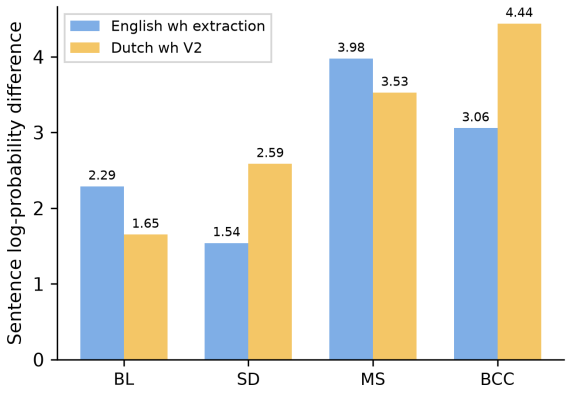


Figure 2: Wh-preference margin (correct/incorrect log-probability difference). MS is largest on English wh (3.98); BCC is largest on Dutch wh V2 (4.44). SD shows the lowest English margin (1.54), consistent with its lack of wh-movement encoding. Chinese is omitted because wh-in-situ does not involve overt wh-extraction.

5 Exploratory Model-internal Analysis

To better understand how the different PPT paradigms affect model-internal processing of multilingual inputs relative to the non-PPT baseline, we next analyze differences at the level of output probabilities (§5.1), intermediate parameters (§5.2), and representational alignment (§5.3).

5.1 Wh-preference Margins

For wh-movement, Figure 2 revealed significant differences depending on the PPT paradigm. Measuring the magnitude of this effect via the log-probability margin between correct and incorrect wh-extraction items, we find that MS shows the highest margin for correct versus incorrect English cases (3.98), consistent with English-type specialization. SD shows the *lowest* English margin (1.54)—which places it below even BL (2.29)—consistent with its lack of any explicit `MOVE` operation. Nonetheless, SD suffers no wh-island degradation, suggesting that a weaker but unbiased wh-representation is preferable to a strong but English-skewed one. BCC has an English margin of 3.06, placing it closest to MS, while also exhibiting higher margins for other languages, including the highest Dutch wh V2 margin (4.44), reflecting **more balanced cross-lingual wh-preferences**.

5.2 Attention Head Specialization

At the level of model parameters, we classify each attention head as either **structurally-oriented (S)** or **proximity-oriented (P)** based on its correlation with dependency structures versus linear token distance, following the intuition of Hu et al. (2025). Concretely, for each head we compute the Pearson correlation between its attention weights and a linear-proximity matrix $\frac{1}{1+|i-j|}$ over three English sentences containing relative clauses and PP modifiers. Heads with $r < 0.5$ are classified as S, the remainder as P.[3] The S/P ratio $= \frac{|S|}{|S|+|P|}$ quantifies the strength of the structural bias induced by each PPT paradigm.

All PPT models exceed BL's S/P ratio (0.847). SD achieves the highest ratio (0.997), followed by MS (0.975) and BCC (0.925). Since the classification is based on English stimuli, the ordering also reflects how strongly each paradigm specializes toward English-like structural patterns: SD's near-ceiling ratio indicates an extreme structural bias with little proximity-based attention, while BCC's comparatively moderate ratio is consistent with its typologically diverse training producing more balanced attention patterns that are not narrowly tuned to any single language's dependency structure.

5.3 Cross-Lingual Representation Alignment

At a broader multilingual level, we next investigate how PPT affects a model's overall cross-lingual representational alignment.

Embedding Alignment. To measure embedding alignment across languages, we first group token embeddings by language. Concretely, for each token in the models' vocabulary, we trace which language split of the pre-training data (`en_nld_zho_equal`) it originates from, assigning tokens to multiple languages if they occur in more than one split (e.g., digits). Since the tokenizer is trained on the same corpus, this allows us to group token IDs and their corresponding embeddings by language. Similarly, this means that if a token only occurs in one language (e.g., most Chinese characters), their embeddings will also only be directly affected by pre-training data in that language.

The resulting groupings in Figure 3 show that PPT substantially increases cross-lingual embedding similarity, with MS and BCC exhibiting the highest gains. These alignment increases are particularly pronounced for non-English and multilingual embeddings, jumping from 0.00–0.05 (BL) to 0.10–0.18 (PPT).

[3] Note that low proximity-oriented correlation may also reflect diffuse, non-local attention; the S/P ratio should therefore be interpreted as a coarse proxy for structural biases.

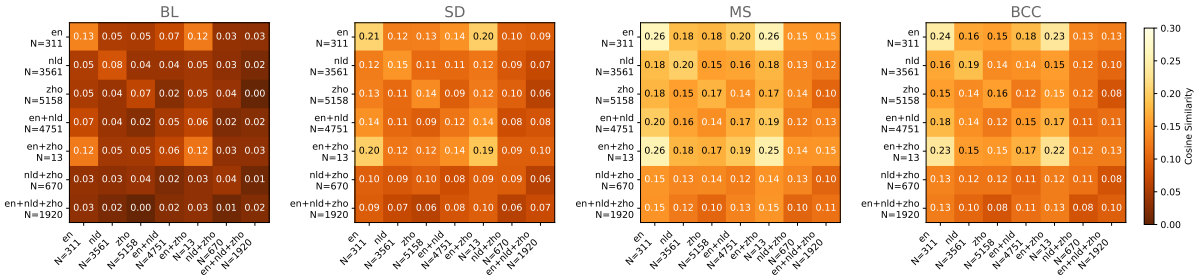


Figure 3: Mean cosine similarities across token embeddings from different language groups (multi-language groups if a token occurs in more than one language data subset). N indicates the number of embeddings per group, summing to the total vocabulary size of 16,384. PPT models show higher cross-lingual similarity than the baseline.

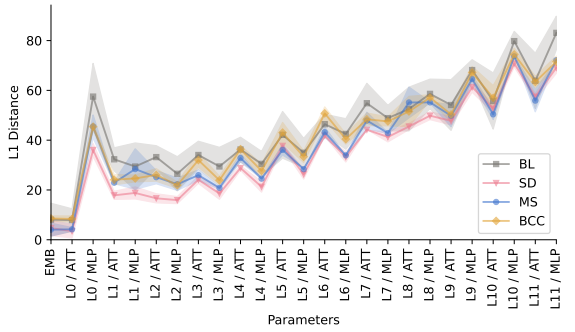


Figure 4: Cross-lingual L1 residual distance across embeddings (EMB), as well as layer-wise attention (ATT) and feed-forward blocks (MLP).

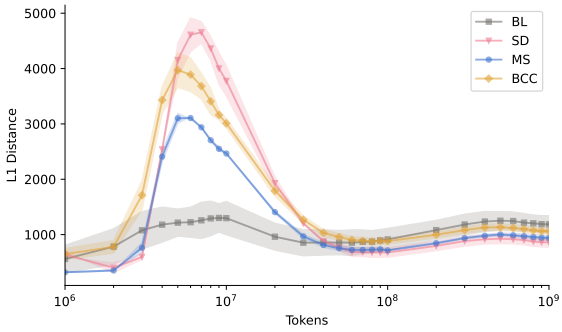


Figure 5: Cross-lingual total L1 residual distance across training (indicated by the number of observed tokens).

Residual Distance. As a proxy for cross-lingual alignment across model components, we measure L1 distances between language-specific representations in each model's residual stream (details in Appendix D) using 1k parallel sentences from FLORES (NLLB Team, 2024). Figure 4 shows that PPT models exhibit lower residual distance (i.e., higher cross-lingual alignment) in the earlier half of layers (L0–L5), before following a trajectory of increasing separation similar to the baseline.

Interestingly, PPT models initially show significantly *larger* distances after PPT (∼10M tokens), but this reverses beyond 100M tokens (Figure 5). A Wilcoxon signed-rank test at $p < 0.05$ with Holm-Bonferroni correction on the final ordering of residual distances, **SD** $\approx$ **MS** $<$ **BCC** $<$ **BL**, shows that SD and MS exhibit significantly lower typological diversity (i.e., convergent representations) compared to BL's higher cross-lingual separation. Meanwhile, BCC's intermediate level of alignment reflects typologically balanced representations that resist under and over-convergence.

6 Discussion

6.1 Principles vs. Parameters in PPT

In the P&P framework (§2), *principles* are constraints every language must satisfy, while *parameters* are constraints that delimit the range of possible variation across languages. Within the tested typological space, the P&P distinction between "principles" and "parameters" maps onto PPT. Language-agnostic structural effects improve performance consistently across languages (SD and MS; §4.2), while parameter-level `MOVE` effects are specific to MS and skewed toward English (§4.3). Since SD encodes hierarchical structure without any typological parameters, yet produces cross-lingual improvements without language bias, this isolates the source of the bias in MS' parameter settings rather than in PPT itself.

Formal language design is thus **not theoretically neutral**: parameter settings affect the induced bias, and should be treated as explicit design variables. BCC addresses this issue by sampling parameters from plausible typological profiles rather than fixing them, eliminating wh-island degradation while preserving overall cross-lingual gains (§4.3).

Ruling Out Alternative Explanations. To establish the parameter fixation hypothesis, we consider four alternative explanations for the wh-island degradation. First, the degradation is unlikely due to insufficient PPT steps, as other wh-improvements (`wh_obj_gap`, `adjunct_island`) clearly emerge within the same 500-step budget. Second, it cannot be attributed solely to the intrinsic complexity of island structures. BL already achieves substantial accuracy on `wh_island` (Table 6), indicating that these constraints are learnable from natural language alone without PPT. Third, the hypothesis that PPT in general disrupts island learning is inconsistent with the SD control, which undergoes the same 500 steps of structural PPT yet shows no wh-island degradation (Table 6). Finally, the absence of island structures in the formal language is complementary rather than alternative to our hypothesis: the "always-move" and "never prohibit" biases may jointly contribute to the degradation. That is, while English itself exhibits both obligatory wh-fronting and strict island constraints, MS's formal language encodes only the former without the latter, creating an asymmetric bias that favors extraction without learning when to block it. The converging evidence from SD and BCC is consistent with parameter fixation as the primary source of the observed bias, though full causal identification requires targeted single-parameter ablation.

6.2 Subtask-level Multilingual Evaluation Is Essential

Behind the modest benchmark-level improvement (§4.2) lie phenomenon-specific subtask-level changes spanning a much wider range (§4.3, Table 5), including the wh-island degradation, which would have been invisible without subtask-level analysis. These results highlight the need to balance language-agnostic SD and language-specific features of MS, while reinforcing the practical implication that typologically plausible inductive biases (BCC) can lead to more balanced cross-lingual performance, while universal structural biases (SD) are preferable when the target language characteristics are unknown. Future work should likely explore how many parameter profiles can be included in BCC-style PPT, to test the limits of inducing biases for many more languages simultaneously. To surface these language-specific differences during evaluation, we thus recommend: (1) standardized reporting of subtask-level Win/Loss distributions and cross-lingual comparisons; (2) explicit documentation of typological parameter settings in formal language design; (3) inclusion of typologically diverse evaluation languages in PPT studies.

6.3 Model-internals Reflect PPT Paradigms

The exploratory analyses in §5 suggest that each PPT paradigm's characteristics are reflected in the model-internal differences induced by each method: parametrization choices surface in output probabilities, PPT reshapes the embedding space that later training builds on, and these early effects persist through subsequent learning phases.

Probabilities Reflect Parametrization. Across our different levels of typological parametrization, §5.1 showed how SD's low English wh-margin reflects its typologically unbiased parametrization, while MS' fixed English-type parameters manifest as an excessive English wh-preference. BCC, by contrast, maintains balanced wh-preferences across languages, matching its superior performance on wh-island resolution. Practically, the fact that SD leads to more balanced results than MS suggests that no typological bias (i.e., randomly sampled parametrization) may be preferable when the target typology is unknown.

PPT Improves Embedding Alignment. Beyond output probabilities, PPT also leaves a mark on the model's internal representations. Cross-lingual representational alignment has proven to be crucial for balanced multilingual performance (Basirat, 2025; Körner et al., 2026), yet in practice, multilingual embeddings are often skewed towards English (Wendler et al., 2024; Harrasse et al., 2025). We observe this same skew: despite our relatively small three-language setup and equal token ratios across languages in the pre-training data, multilingual embeddings still show slightly higher alignment with English (§5.3). It is therefore notable that PPT substantially increases cross-lingual embedding similarity for non-English embeddings compared to standard pre-training on the same data (from 0.00 to 0.18), even though it only encodes abstract structural information. This indicates that even a small amount of PPT promotes closer cross-lingual alignment of the embedding space that forms later in training. Encoding UG-principles appears to further aid this process, as both MS and BCC show additional alignment gains compared to SD alone.

Early PPT Affects Later Learning Phases. This improved alignment, however, does not emerge immediately. §5.3 shows that PPT models have lower cross-lingual residual stream distance at the end of pretraining, but that the training dynamics leading to this result are non-trivial: all PPT models initially show dramatically larger cross-lingual distance than BL in the early phases of pre-training, right after PPT (Figure 5). Beyond this phase, cross-lingual distance falls and remains consistently below the baseline from around 100M tokens onwards. Notably, this stage of pre-training coincides with a previously reported "critical learning phase" (Müller-Eberstein et al., 2023; Chen et al., 2024), during which LMs acquire most of their fundamental syntactic capabilities. Together with the high level of structural specialization of attention heads in PPT models (§5.2), this indicates that the inductive biases introduced by PPT establish a structure that shapes training dynamics well beyond their initial phase.

7 Conclusion

Through the first subtask-level multilingual analysis of UG-inspired PPT across English, Dutch, and Chinese, we find evidence that MP-STRUCT's inductive bias based on UG principles is *partially universal but contains language-specific bias*. Structure-derived effects are cross-linguistically consistent, whereas parameter-derived effects—most prominently obligatory wh-fronting—are associated with an English-centric bias that degrades wh-restriction constructions in both English and Dutch. BCC-based parameterization eliminates this degradation while preserving overall gains. Exploratory analyses of model internals are consistent with the hypothesis that PPT's benefits are mechanistically decomposable: structural attention bias (present in all PPT conditions including k-shuffle Dyck) is separable from cross-lingual alignment (requiring language-specific tokens), with each paradigm's parametrization choices surfacing in both output probabilities and representational geometry. Within the tested typological space, the P&P distinction between principles and parameters thus maps onto computational PPT, indicating that for genuinely universal PPT, principled typological parameter design and subtask-level multilingual evaluation are essential.

Limitations

Typological Diversity. Only three languages are tested. SOV languages (e.g., Japanese), polysynthetic languages, and other typological profiles require further investigation. BLiMP phenomenon coverage is also benchmark-dependent; while Dutch `wh_movement_restrictions` tests subjacency-type constraints on wh-extraction (§4.3), it differs from the English `wh_island` subtask in item construction and coverage, so the cross-lingual parallel is suggestive rather than exact. Currently, dedicated island constraint benchmarks for Dutch and Chinese do not exist, limiting the scope of direct comparison.

Experiment Scale. Three seeds provide limited statistical power for individual subtask significance testing. We use only GPT-2 small (124M parameters) with 100M words of training data; whether the observed effects persist, diminish, or amplify with larger models and more data is untested, and small-scale artifacts cannot be ruled out. PPT uses 500 steps, and operates under BabyLM constraints; optimal step counts remain unexplored.

Single-factor Ablations. The attribution of wh-island degradation to wh-fronting fixation is supported by converging evidence (SD control + BCC elimination) but not by targeted single-parameter ablation; BCC changes five parameters simultaneously, so the individual contribution of each parameter remains unquantified. BCC also introduces increased structural diversity alongside typological parameterization; a random-parameterization baseline would help disentangle these factors. Additionally, while BCC uses the same vocabulary and operations as MS, its clause-level parameter sampling may alter the marginal distributions of token frequencies, sequence lengths, and tree depths, meaning that some observed differences could partly reflect distributional rather than purely typological changes. The strongest bias evidence comes from English wh-islands with supporting evidence from Dutch wh-movement restrictions; broader claims about "language bias" should be interpreted with this scope in mind. Internal analyses are exploratory seed-0 observations without cross-seed replication. BCC is not strictly superior to MS, indicating that parameterization alone does not yield optimal PPT design.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP24H00087, JST BOOST Grant Number JPMJBY24B2, and JST CREST Grant Number JPMJCR2565. MME was supported by the Carlsberg Foundation, grant CF-25-0624. MY was supported by the Nippon Foundation HUMAI Program.

References

Ali Basirat. 2025. Multilingual learning strategies in multilingual large language models. In *Proceedings of the 5th Workshop on Multilingual Representation Learning (MRL 2025)*, pages 507–518, Suzhuo, China. Association for Computational Linguistics.

Hagit Borer. 1984. *Parametric Syntax: Case Studies in Semitic and Romance Languages*. Foris, Dordrecht.

Angelica Chen, Ravid Shwartz-Ziv, Kyunghyun Cho, Matthew L Leavitt, and Naomi Saphra. 2024. Sudden drops in the loss: Syntax acquisition, phase transitions, and simplicity bias in MLMs. In *The Twelfth International Conference on Learning Representations*.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Noam Chomsky. 1977. On wh-movement. In Peter Culicover, Thomas Wasow, and Adrian Akmajian, editors, *Formal Syntax*, pages 71–132. Academic Press, New York.

Noam Chomsky. 1981. *Lectures on Government and Binding*. Foris, Dordrecht.

Noam Chomsky. 1995. *The Minimalist Program*. MIT Press, Cambridge, MA.

Noam Chomsky. 2000. Minimalist inquiries: The framework. In Roger Martin, David Michaels, and Juan Uriagereka, editors, *Step by Step: Essays on Minimalist Syntax in Honor of Howard Lasnik*, pages 89–155. MIT Press, Cambridge, MA.

Noam Chomsky. 2001. Derivation by phase. In Michael Kenstowicz, editor, *Ken Hale: A Life in Language*, pages 1–52. MIT Press, Cambridge, MA.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Hans den Besten. 1983. On the interaction of root transformations and lexical deletive rules. In Werner Abraham, editor, *On the Formal Syntax of the Westgermania*, pages 47–131. John Benjamins, Amsterdam.

Abir Harrasse, Florent Draye, Punya Syon Pandey, Zhijing Jin, and Bernhard Schölkopf. 2025. Tracing multilingual representations in llms with cross-layer transcoders. *Preprint*, arXiv:2511.10840.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. Between circuits and Chomsky: Pre-pretraining on formal languages imparts linguistic biases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9691–9709, Vienna, Austria. Association for Computational Linguistics.

C.T. James Huang. 1982. *Logical Relations in Chinese and the Theory of Grammar*. Ph.D. thesis, Massachusetts Institute of Technology.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2025. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. *Preprint*, arXiv:2510.10159.

Felicia Körner, Max Müller-Eberstein, Anna Korhonen, and Barbara Plank. 2026. When meanings meet: Investigating the emergence and quality of shared concept spaces during multilingual language model training. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3149–3169, Rabat, Morocco. Association for Computational Linguistics.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Masato Mita, Taiga Someya, Ryo Yoshida, and Yohei Oseki. 2026. Language acquisition device in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19564–19577, San Diego, California, United States. Association for Computational Linguistics.

Max Müller-Eberstein, Rob van der Goot, Barbara Plank, and Ivan Titov. 2023. Subspace chronicles: How linguistic information emerges, shifts and interacts during language model training. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13190–13208, Singapore. Association for Computational Linguistics.

NLLB Team. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning Music Helps You Read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839, Online. Association for Computational Linguistics.

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8402–8413, Singapore. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Alex Warstadt and Samuel R Bowman. 2022. What artificial neural networks can tell us about human language acquisition. *Algebraic structures in natural language*, 17.

Alex Warstadt, Leshem Choshen, Aaron Mueller, Adina Williams, Ethan Wilcox, and Chengxu Zhuang. 2023. Call for papers – the babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus. *Preprint*, arXiv:2301.11796.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

A Win/Loss Threshold Sensitivity

Table 7 tests whether the Win/Loss conclusions in §4.3 depend on the 2 pp threshold used to classify subtasks. For SD and MS, the W/(W+L) ratio remains within 3 pp of the ±2 pp value across all thresholds, confirming that the overall positive effect is robust. MS is particularly stable (64.1% at ±2–5 pp), suggesting that its wins tend to be large in magnitude. BCC shows a slight downward trend at the most stringent threshold (±5 pp: 56.0%), indicating that a larger share of its wins are moderate in size; this is consistent with BCC's balanced but individually smaller per-subtask gains compared to MS's more concentrated improvements.

Threshold	W	L	T	W/(W+L)
SD vs BL				
±1 pp	89	55	63	61.8%
±2 pp	57	36	114	61.3%
±3 pp	44	31	132	58.7%
±5 pp	28	17	162	62.2%
MS vs BL				
±1 pp	86	51	70	62.8%
±2 pp	59	33	115	64.1%
±3 pp	41	23	143	64.1%
±5 pp	25	14	168	64.1%
BCC vs BL				
±1 pp	82	51	74	61.7%
±2 pp	56	38	113	59.6%
±3 pp	44	29	134	60.3%
±5 pp	28	22	157	56.0%

Table 7: Win/Loss sensitivity to threshold choice. The Win/(Win+Loss) ratio remains stable across thresholds from ±1 to ±5 pp for all three PPT conditions, indicating that the conclusions are not sensitive to the specific 2 pp threshold.

B Effect Patterns by Language

Table 8 summarizes how PPT effects decompose across languages. Three patterns emerge: (1) structural improvements from MERGE-derived bias are universal across all conditions and languages; (2) wh-restriction degradation is specific to MS and observed in both English (−10.3 pp) and Dutch (−3.4 pp), consistent with its fixed obligatory-fronting parameter; (3) BCC eliminates this degradation (eng +0.9 pp) while maintaining balanced group-level gains across all three languages (+0.5–0.8 pp), though Dutch wh_movement_restrictions shows a modest decline (−2.1 pp). SD and BCC converge in showing no wh-island degradation despite differing mechanisms: SD avoids it by not encoding wh-movement at all, while BCC avoids it by encoding typologically diverse movement patterns.

C MS-BCC Generator Details

The MS-BCC generator samples typological parameters per clause following the profiles in Table 2, then applies MERGE, AGREE, and MOVE with the sampled settings to produce linearized sequences. Clauses are concatenated with code-switching across language profiles (Algorithm 1).

Effect	Cond.	eng	nld	zho
Structural ↑ (universal)	SD, MS, BCC	✓	✓	✓
Wh-detection ↑ (partial)	MS, BCC	✓	Lim.	N/A
Wh-restrict. ↓ (MS-spec.)	MS	−10.3	−3.4	—
Wh-restrict. (no degrad.)	SD	−0.3	−0.2	—
Wh-restrict. (no degrad.)	BCC	+0.9	−2.1	—
Group Δ (pp)	SD	+0.7	+0.2	+0.9
Group Δ (pp)	MS	+1.1	+0.4	+0.7
Group Δ (pp)	BCC	+0.8	+0.5	+0.8

Table 8: Effect patterns by language and condition. Structural improvement is universal across all PPT conditions. Wh-restriction degradation is MS-specific; SD and BCC show no systematic degradation. BCC maintains balanced group-level gains across all languages.

The vocabulary consists of the same abstract lexical tokens used in MP-STRUCT (**w1**, **w2**, etc.), ensuring that any performance differences between MS and BCC are attributable to parametrization rather than lexical content.

D Residual Stream Distances

To compute the representational distance between our three target languages (English, Dutch, Chinese), we extract the residual stream at 25 locations in the model: embeddings (EMB), attention blocks (ATT), and feed-forward components (MLP). As a source of parallel data, which allows us to control for language independently of semantic content, we use 1k sentences per language from the development data of the FLORES corpus (NLLB Team, 2024). Each sentence representation is computed from the mean over its tokens. We then compute the L1 distance between matching sentences across all language pairs (excluding self-similarity), taking the mean over all sentence distances.

BPE vs. Morphology-Aware Tokenization: A Case Study on Arabic Language Modeling at BabyLM Scale

Mohamed Badawy[1] **Marion Di Marco**[1,2] **Alexander Fraser**[1,2]
[1]School of Computation, Information and Technology,
Technische Universität München (TUM)
[2]Munich Center for Machine Learning
{muhamed.badawy|marion.dimarco|alexander.fraser}@tum.de

Abstract

Byte Pair Encoding (BPE) is blind to morphology, a poor fit for Arabic, whose words bundle conjunctions, articles, and inflection into one token. The leading morphology-aware strategy for Arabic, Farasa+MorphBPE, relies on Farasa, a supervised segmenter disallowed under the BabyLM 2026 rule against off-the-shelf linguistic tools. We study Arabic segmentation under the BabyLM GPT-2 training pipeline and propose **Morfessor+MorphBPE**, an unsupervised morphology-aware tokenizer.

We compare two dimensions: supervised vs. unsupervised *morphological segmentation* and the *merging constraints* used by BPE. All tokenizer variants cluster within a narrow accuracy band, with improvements rather on language understanding tasks than knowledge tasks, even though we do not observe consistent improvements over BPE. Furthermore, we find that holding the Farasa segmentation fixed, the MorphBPE constraint sharply lowers perplexity but barely changes downstream accuracy; and unsupervised Morfessor boundaries match supervised Farasa on aggregate while scoring higher on grammaticality (MultiBLiMP). Morfessor+MorphBPE thus offers a practical path to morphology-aware tokenization for Arabic.

1 Introduction

The BabyLM Challenge asks whether useful language competence can emerge from a developmentally plausible data budget of 100 million words or less (Warstadt et al., 2023; Hu et al., 2024). At this scale, implementation details like tokenization become central; we study Arabic segmentation to optimally represent its complex morphology. Arabic is a natural stress test, since its words bundle conjunctions, articles, and inflection into a single orthographic token (Habash, 2010), so a frequency-driven BPE tokenizer (Sennrich et al., 2016) learns merges that are statistically optimal but morphologically opaque.

MorphBPE (Asgari et al., 2025) addresses this mismatch with a single change to the BPE training loop. Standard BPE repeatedly merges the most frequent adjacent pair of symbols, wherever it occurs; MorphBPE first segments each word into morphemes with an external segmenter and then *forbids any merge whose two symbols lie on opposite sides of a morpheme boundary*. Frequency still decides which merge to apply, but the boundaries act as walls that merges cannot cross, so every learned token stays inside a single morpheme. We refer to this rule as the *MorphBPE constraint*. The MorphBPE recipe for Arabic obtains segmentation boundaries from *Farasa* (Darwish and Mubarak, 2016), a supervised segmenter trained on the Arabic Treebank. BabyLM 2026 forbids the use of off-the-shelf linguistic tools (Choshen et al., 2026).

In this study, we assess the respective impact of the MorphBPE *constraint* and the *segmentation quality*: we contrast the constrained MorphBPE with plain BPE which allows to merge across morpheme boundaries, and thus is able to compensate over-splitting and to leave high-frequency sequences intact. In addition, we replace Farasa with Morfessor 2.0 Baseline (Creutz and Lagus, 2007; Smit et al., 2014), an unsupervised segmenter trained on simple corpus data.

We compare **Morfessor+MorphBPE** to six reference tokenizers: vanilla BPE, three pure Morfessor variants, Farasa+plain-BPE, and Farasa+MorphBPE. All seven variants cluster within a narrow accuracy band, implying that the unsupervised Morfessor is a sound substitute for Farasa at BabyLM scale. Our contributions are:

1. **Morfessor+MorphBPE** a morphology-aware tokenization strategy for Arabic based on an unsupervised tokenizer competitive with its supervised counterpart.
2. We show a dissociation between the MorphBPE *constraint* and the *segmentation*: with

Proceedings of the Second BabyLM Workshop, pages 199–208
October 29, 2026 ©2026 Association for Computational Linguistics

the Farasa segmentation fixed, the no-cross-boundary constraint sharply improves intrinsic boundary quality and perplexity but leaves downstream accuracy largely unchanged - at this scale, downstream accuracy is robust to both the constraint and the choice of segmenter.

3. While BPE remains strong overall, we find that morphology-aware tokenization yields *task-dependent* effects, with improvements observed rather in language understanding tasks than knowledge or reasoning tasks.

2 Related Work and Background

Tokenization for morphologically rich languages. A tokenizer maps raw text to the discrete units a language model predicts and is the first step in LLM pre-training. One dominant approach, Byte Pair Encoding (BPE) (Sennrich et al., 2016), starts from characters and iteratively merges the most frequent adjacent pair until a target vocabulary size is reached; merges are driven purely by corpus frequency and are blind to morphology. Previous work has questioned whether standard tokenization approaches like BPE, WordPiece (Schuster and Nakajima, 2012) and SentencePiece (Kudo and Richardson, 2018) can adequately capture morphological structures, and several studies observed that linguistically guided segmentation can lead to improvement in downstream tasks.

Klein and Tsarfaty (2020) argue that linear segmentation is not suitable to capture morphology by investigating the task of multi-tagging Hebrew. Hofmann et al. (2021) find that linguistically aligned segmentation improves sentiment and topic classification.

Further studies evaluate language-specific approaches for morphologically aligned segmentation, for example Antoun et al. (2020) and Asgari et al. (2025) for Arabic relying on Farasa segmentations, and Nzeyimana and Niyongabo Rubungo (2022) relying on a custom morphological tool for Kinyarwanda. Mager et al. (2022) compare morphologically grounded tokenization strategies with BPE for very low-resource languages. In particular, they compare supervised and unsupervised segmentation and find that unsupervised segmentation improves over BPE, whereas supervised morphological segmentation lead to comparatively low results. Hou et al. (2023) explore tokenization with two unsupervised segmentation methods (Morfessor and a finite-state based system) for Russian, Turkish and Finnish in comparison to BPE, and observe advantages of a morphological segmentation with regard to perplexity, efficiency and performance on downstream tasks. Jabbar (2023) explores non-surface preserving tokenization into a canonical representation requiring an additional detokenization step; he reports improvements on the GLUE Benchmark. Hudspeth et al. (2026) implement a context-aware tokenization strategy for Latin where multiple segmentation options are dismabiguated based on POS information.

Fusco et al. (2025) evaluate tokenization strategies in a 10M BabyLM setting. While they find no clear winning tokenizer, with BPE as a strong baseline, they observe that morphological tokenization obtains the best intrinsic segmentation which leads to a more robust generalization in comprehension tasks. This finiding is not unsimilar to our general observations.

The BabyLM Challenge and BabyBabelLM. The BabyLM Challenge (Warstadt et al., 2023; Hu et al., 2024) studies pretraining under human-scale data budgets (10M–100M words), motivated by the observation that human learners acquire language from orders of magnitude less input than production-scale LMs receive. Its 2026 edition expands the challenge to a multilingual setting and introduces an explicit prohibition on off-the-shelf linguistic tools. The training data for self-trained ancillary models is counted towards the total training budget; we follow Fusco et al. (2025) and train our tokenizers on the full corpus.

MorphBPE. MorphBPE is a morphology-aware BPE method (Asgari et al., 2025), which modifies standard byte-pair encoding (Sennrich et al., 2016) by a single algorithmic constraint: pair frequencies are counted only *within* morpheme boundaries provided by an external segmenter, such that no merge can cross a boundary. On Arabic, the paper reports a morphological consistency F1 rising from 0.00 (vanilla BPE at 96K vocabulary) to 0.66 (MorphBPE at 96K). Separately, across four typologically diverse languages, MorphBPE achieves lower language-model training cross-entropy than vanilla BPE.

Two aspects in their evaluation are important for positioning our contribution. First, the reported results are collected at 300M–1B parameter scale on the FineWeb2 corpus, orders of magnitude beyond BabyLM scale, and the authors explicitly leave extrinsic downstream evaluation as future work.

Second, the Arabic experiments use Farasa (Darwish and Mubarak, 2016), a supervised segmenter trained on the Arabic Treebank, disallowed under BabyLM rules. In this paper, we evaluate the difference between Farasa and Morfessor to provide segmentation boundaries for tokenization.

Morfessor and its variants. Morfessor (Creutz and Lagus, 2007) learns morpheme inventories from raw word lists via Minimum Description Length; it is fully unsupervised and requires no external annotation. We use the Baseline model from the Morfessor 2.0 toolkit (Smit et al., 2014), which supports three canonical weighting variants that yield systematically different segmentation behaviour: **Type** (every word type counted once, morphologically faithful), **Token** (log-transformed frequency, a statistical compromise), and **Frequency** (highly frequent words kept unsegmented). All three were compared as tokenizers for English BabyLM in the SIGMORPHON 2024 shared-task submission (Li, 2024), providing the direct precedent for our experiment.

Morfessor was designed for concatenative morphology (Finnish, Turkish), and its behaviour on Arabic's templatic root-and-pattern morphology (Habash, 2010) is an open question we test empirically. To our knowledge, no prior work has evaluated Morfessor variants as BPE boundary providers for Arabic under BabyLM scale constraints.

3 Methodology

This section describes the data, pre-processing, training pipeline and evaluation.

3.1 Data and preprocessing

All seven tokenizers are trained on the BabyBabelLM Arabic Tier-2 corpus (Jumelet et al., 2025): 30,533 documents, ∼7.85M whitespace tokens, and 701,698 unique word types. We use a 95/5 train–validation split, identical across 5 runs with different seeds. We apply *no* Arabic-specific letter normalization (no diacritic stripping, no alef folding); the BabyBabelLM upstream pipeline does not apply such normalization either.

3.2 Seven tokenizers under comparison

The tokenizers span three axes: presence or absence of a morphological prior; whether that prior is unsupervised (Morfessor) or supervised (Farasa); and whether morphological information is used as the sole segmentation signal (pure Morfessor), as a hard constraint on BPE merges (MorphBPE), or merely as a pre-segmentation that BPE may later merge across (Farasa+plain-BPE).

1. **Vanilla BPE**. A byte-level BPE tokenizer (Sennrich et al., 2016) trained on raw text with frequency-only merges. This is the standard baseline used by the BabyBabelLM Arabic monolingual reference.
2. **Morfessor Type**. Pure Morfessor 2.0 Baseline (Smit et al., 2014) in the Type-weighted variant: every observed word type contributes to the Minimum Description Length objective with equal weight, producing morphologically faithful segmentations.
3. **Morfessor Token**. Pure Morfessor with token-weighted frequency smoothing ($f_{\text{new}} = \log_{30}(f_{\text{old}} + 1)$), a compromise that biases the model toward frequency-typical segmentation.
4. **Morfessor Frequency**. Pure Morfessor with a frequency cutoff: words above a corpus-frequency threshold are left unsegmented, while lower-frequency words are segmented normally. All three Morfessor variants (Type, Token, Frequency) follow the variant taxonomy of Li (2024).
5. **Farasa+plain-BPE**. Farasa's supervised segmentations are used to pre-segment each word, but plain BPE is then trained on the result with *no* constraint - merges are free to cross the Farasa boundaries. This isolates the effect of the MorphBPE constraint: it shares Farasa's segmentation with Farasa+MorphBPE but drops the no-cross-boundary rule.
6. **Farasa+MorphBPE**. MorphBPE with Farasa's supervised segmentations, where merges cannot cross morpheme boundaries (equivalent to the original MorphBPE (Asgari et al., 2025).
7. **Morfessor+MorphBPE**. Our proposed tokenizer: the MorphBPE algorithm using Morfessor Type as its boundary provider.

3.3 The MorphBPE algorithm

Tokenizers 6 and 7 both instantiate the MorphBPE algorithm, which differs from standard BPE by one line: pair frequencies are counted only *within* morpheme boundaries, so no merge crosses a boundary.

```
Algorithm 1 (MorphBPE)
1. Initialize vocab with individual chars
```

```
2. Segment corpus using morph segmenter
3. while |vocab| < vocab_size:
4.   Count byte-pair frequencies
5.     [morph-aware: only within-morph]
6.   Merge highest-frequency pair
7.   Add merged symbol to vocab
```

The morphological segmenter is invoked only during *training*. At inference, the learned merge table is replayed on the raw character stream as in standard BPE. Because the morpheme-boundary constraint acts only on which merges are allowed to form during training, a merge crossing a morpheme boundary never enters the merge table and can therefore never be applied at inference; the morphological information is baked into the learned vocabulary rather than reapplied per input.

3.4 Boundary providers for MorphBPE

We compare the supervised off-the-shelf tool Farasa and Morfessor, an unsupervised model trained on the BabyLM corpus.

Farasa (supervised). Farasa (Darwish and Mubarak, 2016) is a supervised state-of-the-art Arabic segmenter trained on the Arabic Treebank with hand-annotated morphological segmentation. Crucially, we use Farasa in a *surface-preserving* mode: it only inserts boundaries between characters and never rewrites, expands, or normalizes them, so the concatenation of the segments is identical to the input.

Morfessor Type (unsupervised). Morfessor 2.0 Baseline (Smit et al., 2014) learns a morpheme inventory from raw word-count data via Minimum Description Length, without any external annotation. We train it from scratch on the BabyBabelLM Arabic corpus, restricted to word types with frequency ≥ 30 to filter typos and single-occurrence noise. Type weighting is chosen because it is the most morphologically faithful of the three variants (Section 3.2), which matters when Morfessor's role is to supply boundaries rather than to segment directly.

Comparison Table 1 shows the different tokenizer outputs on three Arabic words. The contrast is clearest for يكتبونها (*ya-ktub-ūna-hā*, "they write it"): Morfessor+MorphBPE recovers four morphologically meaningful pieces (prefix, root, plural, object clitic), whereas both Farasa variants keep the verb stem يكتب fused. On بمكتبتهم (*bi-maktabati-him*, "in their library") the three non-Morfessor tokenizers coincide, illustrating that morphology-aware and frequency-driven tokenizers often agree and diverge only on specific constructions.

Tokenizer	Tokenization
والمدرسة *wa-l-madrasa*, "and the school"	
Vanilla BPE	وال | مد | رسة
Farasa+plain-BPE	وال | مد | رسة
Farasa+MorphBPE	وال | مدرس | ة
Morfessor+MorphBPE	و | ال | درسة
بمكتبتهم *bi-maktabati-him*, "in their library"	
Vanilla BPE	بم | كت | بت | هم
Farasa+plain-BPE	بم | كت | بت | هم
Farasa+MorphBPE	بم | كت | بت | هم
Morfessor+MorphBPE	ب | م | كت | بت | هم
يكتبونها *ya-ktub-ūna-hā*, "they write it"	
Vanilla BPE	ي | كتب | ونها
Farasa+plain-BPE	يكتب | ونها
Farasa+MorphBPE	يكتب | ونها
Morfessor+MorphBPE	ي | كتب | ون | ها

Table 1: Example tokenizations of three Arabic words (| marks token boundaries). Tokens are shown in right-to-left display order; the leading word-boundary marker is omitted. Colour marks morphological quality: green = a meaningful morpheme (or a valid fusion of adjacent morphemes, e.g. وال = *wa+al*), red = a spurious sub-morpheme fragment with no standalone meaning. Morfessor+MorphBPE is fully green on the clitic-heavy verb يكتبونها but oversplits the stem مكتب in بمكتبتهم.

The examples also reveal a systematic tendency of Morfessor to *oversplit*. On this word, Farasa keeps the noun stem مكتب (*maktab*, "library") intact - segmenting the word as ب \| مكتب \| ت \| هم - whereas Morfessor+MorphBPE fragments that stem into م \| كت \| بت, yielding the five-piece ب \| م \| كت \| بت \| هم. Because Morfessor is unsupervised and lacks a lexicon of Arabic stems, it sometimes breaks a single morpheme into sub-units that recur frequently in the corpus but are not themselves meaningful. This oversplitting is the main qualitative weakness of the unsupervised boundaries; it is mild on the clitic-heavy verb يكتبونها, where Morfessor's prefix/root/suffix split is linguistically apt, but visible on stems that Farasa's supervised lexicon keeps whole.

3.5 Intrinsic boundary agreement

To quantify the morphological alignment of each tokenizer, we measure segmentation quality with Farasa's segmentation as reference. For every word in the corpus, we compute precision, recall and F1 of each tokenizer's splits.

Table 2 reports the result over all word types

Tokenizer	P	R	F1
Vanilla BPE	0.172	0.312	0.222
Farasa+plain-BPE	0.157	0.310	0.209
Farasa+MorphBPE	0.304	0.634	0.411
Morfessor+MorphBPE	0.304	0.682	0.420
Morfessor (pure, Type)	0.334	0.825	**0.476**

Table 2: Intrinsic boundary agreement with Farasa morpheme boundaries, over all BabyLM-Arabic word types with ≥ 1 reference boundary. (Highest F1 score in bold face.)

that carry at least one reference boundary (∼369K types). Because the reference is Farasa, the two Farasa-derived tokenizers could in principle be favoured; in practice they are not near-ceiling (their F1 is around 0.41), because the metric scores the *final* tokenizer output after BPE merges, not the raw segmentation.

Two patterns stand out. First, the *constraint* matters intrinsically: adding the no-cross-boundary rule to the same Farasa segmentation nearly doubles boundary F1 (0.209 → 0.411), since plain BPE otherwise merges straight back across the morpheme boundaries it was given. Second, the *unsupervised* boundaries are competitive: Morfessor+MorphBPE matches or exceeds Farasa+MorphBPE (0.420 vs. 0.411), and pure Morfessor is the most morpheme-aligned tokenizer of all (0.476), confirming that Morfessor recovers Arabic morpheme boundaries well despite using no labeled data - though it also oversplits, which lowers its precision.

3.6 Implementation choices

We *seed* the vocabulary with a 281-character alphabet that covers ASCII letters, digits, punctuation, Arabic punctuation, and common Unicode symbols, before any morpheme-aware merge is performed. This guarantees that non-Arabic characters have their own tokens, which is necessary for downstream benchmarks whose scoring templates rely on ASCII option labels (`A`, `B`, `C`, `D`) or digits. Second, we adopt GPT-2's (Radford et al., 2019) Ġ-prefix word-boundary convention with `add_prefix_space=True`, so that a given word tokenizes identically at every sentence position. Position consistency is a prerequisite for minimal-pair evaluation frameworks that score candidates at arbitrary offsets.

3.7 Model, training, and evaluation

Model. We match the BabyBabelLM Arabic monolingual baseline (Jumelet et al., 2025): a GPT-2 style causal LM (Radford et al., 2019) with 4 transformer layers, 8 attention heads, hidden size 512, FFN size 2048, context length 512, GELU activations, dropout 0.1, and vocabulary size 8,192 - totalling 17.1M parameters.

Training. All seven models are trained for 10 epochs with batch size 16, learning rate 10^{-4}, using five random seeds (42, 41, 5, 11, and 12). Validation loss is tracked on the 5 % held-out split and the checkpoint with best validation loss is used for evaluation.

Evaluation. We use the BabyBabelLM Arabic benchmark suite, scored with a single evaluation harness (`lm-eval` v0.4.12) and identical task configurations across all tokenizers. The BabyBabelLM Arabic evaluation group (Jumelet et al., 2025) spans minimal-pair grammaticality (MultiBLiMP Arabic), compositional semantics (XCOMPS), natural language inference (XNLI Arabic, MultiNLI Arabic), MMLU-style reasoning, story completion, commonsense reasoning, factual recall, reading comprehension, and topic classification.

4 Experimental Setup

Evaluation harness. All numbers reported in Section 5 are produced by `lm-evaluation-harness` v0.4.12, invoked with a custom task group `babybabellm_ara` covering the subtasks in Table 3. Scoring is zero-shot with no in-context examples. Each candidate answer is scored by loglikelihood under the model, and the highest-likelihood option is selected. We use `acc,none` as the primary metric for most tasks; TruthfulQA and XStoryCloze use `acc_norm,none` (length-normalised accuracy), following the harness defaults for those tasks. No answer normalization, prompt engineering, or output post-processing is applied.

The BabyBabelLM Arabic task group. Table 3 lists the 14 subtasks retained for analysis; they span formal linguistic competence (MultiBLiMP, XComps), natural language inference (XNLI, MultiNLI), knowledge and reasoning (MMLU, Include-44, BMLama, ARC-Easy, COPA, WinoGrande, TruthfulQA), reading comprehension (Belebele), topic classification (SIB200), and story

Subtask	Type	Metric
MultiBLiMP	Grammaticality (min. pair)	acc
XComps	Compositional semantics	acc
XNLI	NLI	acc
MultiNLI	NLI	acc
MMLU	Multitask knowledge	acc
Include-44	Multi-domain knowledge	acc
BMLama	Factual recall (LAMA)	acc
ARC-Easy	Grade-school science QA	acc
COPA	Causal commonsense	acc
WinoGrande	Coreference / commonsense	acc
TruthfulQA	Truthfulness QA	acc_norm
Belebele	Reading comprehension	acc
SIB200	Topic classification	acc
XStoryCloze	Story completion	acc_norm

Table 3: The 14 subtasks reported for the `babybabellm_ara` score.

completion (XStoryCloze). The aggregate reported here is the arithmetic mean over the 14 displayed subtask scores. All tasks are sourced from the BabyBabelLM Arabic evaluation suite (Jumelet et al., 2025) and use the task templates distributed with that suite.

5 Results

Table 4 reports accuracy for all seven tokenizers on the 14 BabyBabelLM-Arabic subtasks retained for analysis plus the aggregate score. Per-column best scores are shown in bold, including the best aggregate score.

Aggregate and per-task patterns. On the corpus-level aggregate score, Morfessor Type is the strongest tokenizer (0.3611), followed by Farasa+MorphBPE (0.3377), Morfessor+MorphBPE (0.3361), Vanilla BPE (0.3330), Farasa+plain-BPE (0.3315), Morfessor Token (0.3302), and Morfessor Freq (0.3288). All seven sit within a $\sim$3-percentage-point band, and no tokenizer dominates uniformly across the 14 subtasks. Because the aggregate spread is narrow even with five seeds, we interpret small aggregate differences as *trends* and focus our analysis on the larger, directionally consistent per-task effects and on the controlled pairwise comparisons (constraint on/off; supervised/unsupervised boundaries) that hold all other factors fixed.

Column-wise best scores are distributed across five of the seven tokenizers. Vanilla BPE wins on four subtasks that either scaffold on ASCII characters (ARC-Easy, TruthfulQA) or reduce to lexical overlap (MultiBLiMP, Include-44). Morfessor Type wins on four semantics-/classification-heavy subtasks (XComps, MultiNLI, ARC-Easy tie, SIB200). Morfessor Freq wins on two commonsense reasoning subtasks (COPA, WinoGrande). Farasa+MorphBPE wins on four subtasks that mix NLI, knowledge, reading comprehension, and story completion (XNLI, MMLU, Belebele, XStoryCloze). Morfessor+MorphBPE wins on one factual-recall subtask (BMLama). Farasa+plain-BPE and Morfessor Token win on no subtask.

MorphBPE constraint. Farasa+plain-BPE and Farasa+MorphBPE share an identical Farasa segmentation and differ only in whether BPE merges may cross morpheme boundaries. The constraint (Farasa+MorphBPE) yields a much lower language-model validation perplexity ($\sim$50 vs. $\sim$81), yet the downstream aggregate difference is small (0.3377 vs. 0.3315, $\sim$0.6 points). The effect is task-dependent: dropping the constraint helps on MultiBLiMP and COPA but hurts on MultiNLI, Belebele, and XStoryCloze. Thus, the constraint substantially improves the model's next-token prediction while producing only marginal, mixed downstream effects.

Supervised vs. unsupervised boundaries. At the aggregate level, Farasa+MorphBPE and Morfessor+MorphBPE are essentially tied (0.3377 vs. 0.3361, a $\sim$0.16-point gap). Morfessor+MorphBPE outperforms on MultiBLiMP, XComps, BMLama, and TruthfulQA and underperforms on MultiNLI, COPA, Belebele, and XStoryCloze. The largest single-task gap is on MultiBLiMP, where the unsupervised variant improves grammaticality accuracy by 4.4 percentage points, indicating that the unsupervised boundaries are at least as useful as Farasa's for the task most directly targeting morphosyntactic competence.

6 Discussion

Validation perplexity. Perplexity is the exponential of the model's mean per-token cross-entropy on the held-out validation split; lower values indicate a better next-token predictor.[1] We report the three MorphBPE-family tokenizers and pure Morfessor Type in Table 5, where we observe two effects.

[1]Perplexity is only loosely comparable across tokenizers, since each induces a different token count for the same text. The three MorphBPE-family tokenizers compared here share vocabulary size 8,192 and a similar segmentation granularity, so their comparison is reasonable but not a normalised per-byte measure. The pure Morfessor Type result is included only as an exploratory reference.

	BabyLM-ara	Grammar & Semantics		Language Understanding				Knowledge & Reasoning							Story
	Aggregate	MultiBLiMP	XComps	XNLI	MultiNLI	Belebele	SIB200	MMLU	Include-44	BMLama	ARC-Easy	COPA	WinoGrande	TruthfulQA	XStoryCloze
Vanilla BPE	0.3330 ± 0.002	**0.7539**	0.5193	0.3313	0.3166	0.2311	0.1127	0.2307	**0.2355**	0.1877	**0.2586**	0.4944	0.4938	**0.3333**	0.5163
Morfessor Type	**0.3611 ± 0.003**	0.7531	**0.6485**	0.3333	**0.3590**	0.2289	**0.2500**	0.2295	0.2246	0.1036	**0.2586**	0.5056	0.4905	0.2058	0.5201
Morfessor Token	0.3302 ± 0.001	0.7416	0.5161	0.3333	0.3244	0.2289	0.1961	0.2295	0.2246	0.1624	0.2556	0.5169	0.4905	0.2857	0.5147
Morfessor Freq	0.3288 ± 0.001	0.7259	0.5226	0.3333	0.3154	0.2289	0.1961	0.2295	0.2246	0.1533	0.2518	**0.5281**	**0.5021**	0.2874	0.5170
Farasa+plain-BPE	0.3315 ± 0.002	0.7160	0.5253	0.3325	0.3238	0.2244	0.1961	0.2280	0.2264	0.1740	0.2484	0.5169	0.4773	0.3010	0.4992
Farasa+MorphBPE	0.3377 ± 0.003	0.7004	0.5273	**0.3353**	0.3396	**0.2456**	0.1961	**0.2342**	0.2192	0.1857	0.2548	0.4831	0.4946	0.2772	**0.5217**
Morfessor+MorphBPE	0.3361 ± 0.001	0.7449	0.5333	0.3333	0.3168	0.2333	0.1961	0.2307	0.2264	**0.1888**	0.2488	0.4607	0.4938	0.2925	0.4992

Table 4: BabyBabelLM Arabic evaluation across the seven tokenizers, with the 14 subtasks grouped into four capability categories (Grammar & Semantics; Language Understanding; Knowledge & Reasoning; Discourse). **BabyLM-ara** is the official `babybabellm_ara` aggregate; it is the benchmark-defined score and is *not* the arithmetic mean of the displayed subtask columns. Per-subtask scores are single-seed (42). Per-column best is bolded.

Tokenizer	Val. PPL ↓
Morfessor Type	50.7
Farasa+plain-BPE	80.9
Farasa+MorphBPE	50.2
Morfessor+MorphBPE	**40.5**

Table 5: Validation perplexity (lower is better). Adding the MorphBPE constraint to a fixed Farasa segmentation lowers perplexity by ∼38 % (80.9→50.2); replacing Farasa's supervised boundaries with unsupervised Morfessor boundaries lowers it by a further ∼19 % (50.2→40.5). Pure Morfessor Type is included as an exploratory reference and is not used in the controlled comparisons.

The *constraint* effect: adding the no-cross-boundary rule to a fixed Farasa segmentation cuts perplexity from 80.9 to 50.2. The *supervision* effect: replacing Farasa's supervised boundaries with Morfessor's unsupervised ones cuts it further to 40.5 - so the rule-compliant, unsupervised tokenizer is in fact the strongest next-token predictor of the three matched MorphBPE-family configurations. The exploratory pure Morfessor Type result (50.7) is higher than that of Morfessor+MorphBPE (40.5), suggesting that Morfessor segmentation alone does not explain the latter's low perplexity.

Perplexity and downstream accuracy are not connected. The clearest single result is that the MorphBPE constraint buys a large reduction in validation perplexity (∼50 vs. ∼81 against its unconstrained twin) but only a ∼0.6-point aggregate change downstream. Because the two tokenizers being compared share an identical Farasa segmentation, this is not a confound of segmentation quality: the constraint genuinely helps the language model predict the next token, yet that intrinsic gain does not propagate to the multiple-choice and minimal-pair tasks that make up the benchmark suite. The practical lesson for BabyLM-scale tokenizer design is that validation perplexity is a poor proxy for downstream utility, and tokenizers should be selected on task metrics rather than on training loss.

At this scale, segmentation matters more than the constraint. Reading the three Farasa-segmented tokenizers together - vanilla BPE (no Farasa), Farasa+plain-BPE (Farasa segmentation, no constraint), and Farasa+MorphBPE (Farasa segmentation, with constraint) - the downstream scores move little once the segmentation is fixed. The no-cross-boundary rule reshapes the learned merge table and the model's perplexity, but the units the model ultimately sees are dominated by the segmentation it was given. This suggests that, at 10M tokens, the value of morphology-aware BPE is carried mostly by *where the boundaries are drawn*, not by *whether merges are forbidden from crossing them*.

Unsupervised boundaries are competitive with supervised ones. Holding the MorphBPE constraint fixed, replacing Farasa's supervised boundaries with Morfessor's unsupervised ones costs essentially nothing on aggregate (0.3377 vs. 0.3361) and *improves* the benchmark most directly tied to morphosyntax (MultiBLiMP, +4.4 points). Our experiments indicate that not relying on a supervised segmenter does not degrade morphological competence at this scale. Morfessor+MorphBPE is thus not a compromise, but is competitive on its own terms.

Gains live on formal, not functional, competence. Every tokenizer performs near chance on the knowledge- and reasoning-oriented subtasks (MMLU, BMLama, ARC-Easy, Belebele), and tokenizer choice barely moves them. Where tokenizers separate at all is on minimal-pair grammaticality and compositional semantics. In the formal-vs-functional framing of Mahowald et al. (2024), our results place any benefit of morphology-aware tokenization squarely on the *formal linguistic competence* axis: it can help a small model learn the shape of the language, but it cannot manufacture world knowledge that the 10M-token budget never contained. This is consistent with prior BabyLM findings that knowledge tasks are largely unlearnable at this scale (Jumelet et al., 2025; Hu et al., 2024), and it clarifies what a tokenizer intervention can and cannot be expected to do.

7 Conclusion

We studied morphology-aware tokenization for Arabic at BabyLM scale through a controlled comparison of seven tokenizers - vanilla BPE, three pure Morfessor variants, and three Farasa/Morfessor combinations - all trained and evaluated under an identical pipeline. Overall, BPE remains a strong baseline and no tokenizer emerges as a clear winner.

Our central contribution is **Morfessor+MorphBPE**, an unsupervised morphology-aware BPE tokenizer that, to our knowledge, is the first Arabic morphology-aware tokenizer compliant with the BabyLM 2026 prohibition on off-the-shelf linguistic tools. It is competitive with its supervised counterpart on aggregate accuracy and matches or exceeds it on minimal-pair grammaticality, without consulting any labeled resource. Two comparisons provide further insights: holding the Farasa segmentation fixed, the MorphBPE no-cross-boundary constraint substantially lowers perplexity but barely changes downstream accuracy, indicating that at this scale the segmentation - not the constraint - drives downstream behaviour; and holding the constraint fixed, unsupervised Morfessor boundaries are competitive with supervised Farasa boundaries. Any benefit of morphology-aware tokenization concentrates on formal linguistic competence rather than on knowledge-based tasks, which remain near chance at 10M tokens. We hope these results offer a practical, rule-compliant default for tokenizing low-resource morphologically rich languages, and a note of caution against selecting tokenizers by perplexity alone.

Limitations

Limited seed budget. Every model is trained with five random seeds (42, 41, 5, 11, and 12). This improves stability over a single run, but the aggregate scores still fall within a $\sim$3-percentage-point band, so we deliberately frame small aggregate differences as trends (Section 5). The larger per-task effects (e.g. the $\sim$4-point MultiBLiMP gap and the perplexity gap between the constrained and unconstrained Farasa tokenizers) are more likely to be robust, but broader replication would be needed to establish fine-grained significance.

Single corpus, scale, and architecture. All experiments use the BabyBabelLM Arabic Tier-2 corpus ($\sim$10M tokens), one vocabulary size (8,192), and one GPT-2-style architecture (17.1M parameters). We do not test whether the observed patterns hold at the 100M-token tier, at larger vocabularies (the original MorphBPE paper uses 96K), or under a masked or GPT-BERT objective. The near-chance performance on knowledge and reasoning tasks is a property of this scale, not of any tokenizer.

Morfessor and templatic morphology. Morfessor was designed for concatenative morphology and segments Arabic by splitting the linear character string; it cannot represent the non-concatenative root-and-pattern structure of Arabic templatic forms. It nonetheless reliably captures Arabic's frequent linear affixation - prefixed conjunctions and prepositions, and suffixed inflection and object clitics - directly from raw word counts and without any labeled segmentation data, which is exactly the boundary information our morphology-aware tokenizer relies on. We explore only its three standard Baseline variants and do not evaluate Morfessor's EM/Categories models or other unsupervised segmenters.

No intrinsic morphological metric. We report downstream task accuracy but not the intrinsic morphological consistency metrics (μ_c, μ_e) used by the original MorphBPE paper, so we cannot directly relate our downstream results to tokenizer-level morphological fidelity. We also did not audit Farasa's segmentation errors, which propagate into the two Farasa-based tokenizers.

Evaluation scope. We evaluate the models only on the official BabyBabelLM Arabic benchmark

suite used in the BabyLM evaluation. We do not test them on additional external Arabic downstream tasks or generation benchmarks. Our conclusions are therefore limited to the tasks represented in the official suite.

Acknowledgments

This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools*.

Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. 2025. MorphBPE: A morpho-aware tokenizer bridging linguistic complexity for efficient LLM training across morphologies. *arXiv preprint*.

Leshem Choshen, Alex Warstadt, Aaron Mueller, Candace Ross, Ryan Cotterell, Tal Linzen, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1).

Kareem Darwish and Hamdy Mubarak. 2016. Farasa: A new fast and accurate Arabic word segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*.

Achille Fusco, Maria Letizia Piccini Bianchessi, Tommaso Sgrizzi, Asya Zanollo, and Cristiano Chesi. 2025. Linguistic units as tokens: Intrinsic and extrinsic evaluation with BabyLM. In *Proceedings of the First BabyLM Workshop*, pages 496–507, Suzhou, China. Association for Computational Linguistics.

Nizar Y. Habash. 2010. *Introduction to Arabic Natural Language Processing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Jue Hou, Anisia Katinskaia, Anh-Duc Vu, and Roman Yangarber. 2023. Effects of sub-word segmentation on performance of transformer language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7413–7425, Singapore. Association for Computational Linguistics.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Leshem Choshen, Ryan Cotterell, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Marisa Hudspeth, Patrick J. Burns, and Brendan O'Connor. 2026. Contextual morphologically-guided tokenization for Latin encoder models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5761–5775, Rabat, Morocco. Association for Computational Linguistics.

Haris Jabbar. 2023. MorphPiece: A linguistic tokenizer for large language models. *arXiv preprint arXiv:2307.07262*.

Jaap Jumelet, Bastian Bunzeck, Manar Ali, Mohammad Amin Ghanizadeh, Laurent Prevot, Sheng-Fu Wang, Jou-An Chi, Shu-Kai Hsieh, Francesca Padovani, Alex Warstadt, Sina Zarrieß, and Ryan Cotterell. 2025. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. *arXiv preprint*.

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Zilong Li. 2024. Team Ryu's submission to SIGMORPHON 2024 shared task on subword tokenization. *arXiv preprint*.

Manuel Mager, Arturo Oncevay, Elisabeth Mager, Katharina Kann, and Ngoc Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine

translation of four polysynthetic languages. In *Findings of the Association for Computational Linguistics: ACL 2022*.

Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2024. Dissociating language and thought in large language models. *Trends in Cognitive Sciences*, 28(6):517–540.

Antoine Nzeyimana and Andre Niyongabo Rubungo. 2022. KinyaBERT: a morphology-aware Kinyarwanda language model. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5347–5363, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

A Hyperparameters

Table 6 lists the hyperparameters used for all experiments. The model architecture and training recipe are identical across all seven tokenizers; only the tokenizer varies.

Model (GPT-2)	**Value**
Transformer layers (n_{layer})	4
Attention heads (n_{head})	8
Hidden size (n_{embd})	512
FFN size (n_{inner})	2048
Context length	512
Activation	`gelu_new`
Dropout (attn / embd / resid)	0.1
Initializer range	0.02
LayerNorm ε	10^{-5}
Vocabulary size	8,192
Total parameters	17.1M
Training	**Value**
Epochs	10
Batch size	16
Learning rate	10^{-4}
Max sequence length	512
Validation split	5 %
Dataset	`BabyLM-community/babylm-ara`
Tokenizer	**Value**
Vocabulary size (all)	8,192
Min frequency (BPE)	2
Arabic normalization	None
Add-prefix-space	True (GPT-2 Ġ)

Table 6: Hyperparameters shared across all tokenizer variants.

Structural priors for data-efficient language learning

Yana Veitsman* **Jonas Mayer Martins*** **Jonathan Lautenschlager** **Lisa Beinborn**
University of Göttingen, Germany
firstname.lastname@uni-goettingen.de

Abstract

Efficient language learning requires methods to reduce the reliance on large data and computational resources. We investigate *structural transfer*: First training models on non-language data to induce useful priors for natural language. This approach is a form of weight initialization for multilingual language modeling. We evaluate transfer via next-token-prediction loss, weight shifts in the model, and downstream linguistic benchmarks. Several symbolic data types—notably music, probabilistic grammars, and cellular automata—yield lower language-modeling loss than random initialization. These gains coincide with smaller weight shifts during subsequent language training, suggesting that structural transfer positions models in a more favorable region of the parameter space. However, a lower loss does not translate consistently into better downstream linguistic performance, and transfer from non-language data is less efficient than additional language data. We conclude that non-language data can serve as a partial substitute for language data for the training objective of next-token prediction but does not reliably support broader linguistic generalization.

Models and data | Code repository

1 Introduction

The performance of language models scales strongly with data and compute, yet both resources are fundamentally limited. Both modeling low-resource languages and cognitively plausible modeling, for instance, require learning from little input, making data efficiency an engineering problem and a scientific challenge (Hoffmann et al., 2022). The BabyLM shared task addresses this constraint directly by evaluating systems under low-data conditions and encouraging cognitively inspired approaches (Choshen et al., 2026).

A natural source of inspiration for data-efficient learning is human language acquisition. Despite exposure to relatively limited linguistic input, infants detect and exploit statistical regularities in the language they perceive (Saffran and Kirkham, 2018). We examine whether it is possible to introduce structural priors into language models that facilitate more data-efficient language learning. We therefore aim to understand to what extent non-language data give rise to representations that facilitate subsequent language learning.

Such priors can be instilled in language models through *pre-pretraining*: an additional stage in which models are trained on non-language data before natural-language pretraining (Papadimitriou and Jurafsky, 2020; Hu et al., 2025; Lee et al., 2026). Pre-pretraining constitutes a form of structural transfer, in which a model first learns to predict a structured symbolic signal and then reuses the resulting parameter state during language training. Previous work suggests that this approach can improve language-modeling convergence and downstream performance in domains such as coding, mathematics, and reasoning (Lee et al., 2026; Shinnick et al., 2025). However, the relationship between the parameter changes induced by pre-pretraining and performance on the target objective remains underexplored.

Data efficiency is particularly important in multilingual settings, where models must generalize to typologically and distributionally diverse languages with varying amounts of data. At the same time, multilingual training introduces challenges absent from monolingual settings, including cross-lingual interference from data mixtures (Ye et al., 2025) and conflicting optimization signals (Wang et al., 2023). The multilingual setting thus offers the opportunity to test not only whether gains from structural transfer in monolingual or task-specific settings persist, but also whether structural transfer is language-specific.

* These authors contributed equally to this work.

Proceedings of the Second BabyLM Workshop, pages 209–226
October 29, 2026 ©2026 Association for Computational Linguistics

Approach and contributions We investigate whether structural transfer can accelerate multilingual language learning. We first train GPT-2-style models on structural data—including probabilistic context-free grammars, cellular automata, piano music, and protein sequences—and then continue training on the multilingual BabyLM corpus (Jumelet et al., 2026a). We evaluate transfer along three axes: 1) efficiency on the natural-language next-token prediction objective, 2) model-internal weight shifts, and 3) downstream performance under the BabyLM evaluation framework (Choshen et al., 2026). This setup allows us to test whether certain types of structural data yield a weight initialization that benefits multilingual modeling.

2 Related work

Small-scale language models are motivated by both practical constraints, such as limited data and compute, and cognitively plausible learning scenarios. Work in this setting has shown that models can benefit from structural biases introduced during training (Papadimitriou and Jurafsky, 2020, 2023; Hu et al., 2025; Lee et al., 2026). We view such biases through the lens of initialization: prior exposure to structural data can move a model away from an uninformed random starting point and toward a parameter region that supports more efficient language learning.

2.1 Sample-efficient modeling

Previous entries in the BabyLM challenge (Choshen et al., 2026) have explored a range of mechanisms for improving sample efficiency, including learning from interaction (Charpentier et al., 2025; Mayer Martins et al., 2025), curriculum learning (Warstadt et al., 2023; Diehl Martinez et al., 2023), and multilingual learning contexts (Jumelet et al., 2026a). These approaches share the goal of improving learning when additional data or compute cannot be scaled up.

Multilingual modeling Studying how models generalize across diverse linguistic distributions and data scarcity is especially relevant in multilingual settings. An open question is whether performance disparities across languages reflect model or data biases, or instead arise from specific linguistic properties (Mielke et al., 2019; Poelman et al., 2025; Shani et al., 2026). Multilingual training requires careful design decisions about data mixtures (Ye et al., 2025) and optimization across languages (Wang et al., 2023). One line of work suggests that transfer is most effective between languages with similar linguistic properties (Pires et al., 2019; K et al., 2020; Snæbjarnarson et al., 2023); another suggests that similarity may matter less than matching the informational complexity of the training signal (Lee et al., 2026). The BabyLM challenge offers a controlled setting for examining whether structural transfer can support learning of heterogeneous distributions.

2.2 Weight initialization and inductive biases

In neural language models, inductive biases can be introduced not only through architecture, but also through the model parameter state. Classical initialization schemes use random weights with carefully chosen scaling coefficients to break symmetry and stabilize activations and gradients (Rumelhart et al., 1986; Glorot and Bengio, 2010; He et al., 2015). Although these schemes do not encode task-specific structure, they shape the starting conditions for optimization. Structural pretraining extends this idea: rather than relying only on random initialization, a model can first be trained on a controlled signal that may induce representations useful for later language learning (Papadimitriou and Jurafsky, 2023; Hu et al., 2025; Lee et al., 2026).

2.3 Structural transfer

We understand structural transfer as an umbrella term, encompassing the closely related concepts of curriculum learning, transfer learning, meta-learning, and pre-pretraining. In meta-learning, prior training across related tasks supports rapid adaptation; in curriculum learning, examples are ordered so that simpler data prepares the model for harder cases; and in transfer learning, knowledge from source tasks or languages provides a useful starting point for zero-shot performance or target-task fine-tuning (Wang et al., 2021; Lee et al., 2022; Soviany et al., 2022). Pre-pretraining on structural data (Papadimitriou and Jurafsky, 2020; Hu et al., 2025; Lee et al., 2026; Jiang et al., 2026) combines aspects of these approaches. It introduces an earlier training stage, often on structural non-language data to induce a useful initialization for subsequent language training. The relevant question is therefore not only whether structural data can be learned, but whether the induced parameter state is beneficial for natural language training.

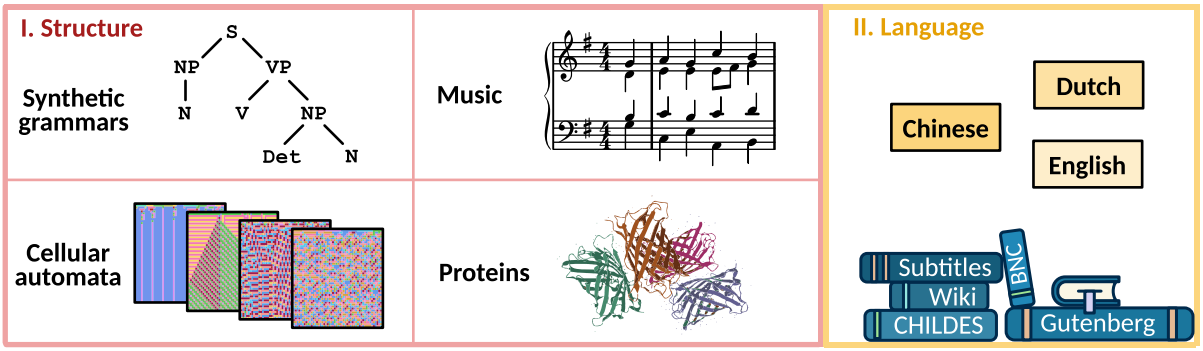


Figure 1: Schematic of the experimental setup. To investigate how training on structure transfers to learning language, we first train language models on one of four symbolic data types: 1) probabilistic context-free grammars; 2) cellular automata; 3) piano music; and 4) protein sequences. The example tree could refer to the sentence *Alice paints the cat.* The piece shown is the beginning of Reger (1900). The protein depicted is green fluorescent protein (GFP) of jellyfish adapted from the Uniprot entry (Boutet et al., 2007). Afterwards, we train these models on a multilingual corpus comprising Chinese, Dutch, and English texts.

Empirical findings. Structural data can vary in entropy, distributional shape, compositionality, hierarchy, and other regularities, which may induce different biases. Previous studies yield mixed conclusions about their benefits. Training on abstract grammars such as PCFGs can give rise to functional hierarchical representations (Jumelet and Zuidema, 2023; Rohweder et al., 2026), but downstream tasks are not always comparable: Hu et al. (2025) report limited gains on BLIMP, while Lee et al. (2026) focus more on reasoning benchmarks. It remains an unanswered question which properties of the structural source interact with the target language data and how they influence the performance on the evaluation task. We therefore examine a range of structural data types with varying degrees of abstractness and local and global dependencies. In our setup, cellular automata provide local rule-based dynamics, synthetic grammars contribute hierarchical and compositional structure, music offers sequential structure with long-range dependencies, and protein biological complexity.

3 Data

The experimental setup consists of training a language model in two stages. In stage I, we train the model from scratch on a structural data type. In stage II, we switch to training on the target corpus.

3.1 Structural data

In stage I, we explore four structural data types: synthetic grammars, cellular automata, music, and protein sequences. As a baseline, we use unstructured sequences of random numbers.

Synthetic grammars A probabilistic context-free grammar (PCFG) is a formal grammar in which production rules for non-terminal symbols are insensitive to the context (Manning and Schütze, 2000). Every production rule is applied with a certain probability, making these synthetic grammars suitable for randomly sampling a wide variety of synthetic data. Created PCFGs mimic natural language closely by synthesizing grammar rules from the Penn Treebank (Marcus et al., 1993), see Appendix A for details. We test two synthetic datasets generated by the same grammar, differing only in the lexicalization: For $\text{PCFG}_{\text{ZIPF}}$, tokens follow a Zipfian distribution; for PCFG_{UNI}, tokens are distributed uniformly.

Cellular automata (CA) A cellular automaton is a row of cells, with each cell being in a discrete state. This row evolves over time under a local *transition rule* f defining a set of update patterns. At each step in time, every cell is updated based on itself and its two neighbors. Cellular automata are a canonical example of self-organized complexity emergent from simple, deterministic rules (von Neumann and Burks, 1966; Gardner, 1970; Wolfram, 1983). The training objective requires the model to infer these rules from diverse patterns, making cellular automata a compelling source for learning functional capabilities. Appendix B provides a visualization and technical details.

To assess the impact of structural complexity, we generate two dataset variants: CA_{16} (medium-complexity dynamics with $K = 16$ states) and CA_{256} (noisy dynamics with $K = 256$ states), each generated from about 1,820 rules with multi-

ple initial conditions per rule. The language model receives these space-time trajectories as concatenated rows. In training, the language model must infer the transition rule based on several trajectories generated from different initial conditions to succeed in next-token prediction.

Music Music is a highly structured non-language signal produced by humans. We use piano music from the ARIA-MIDI dataset (Bradshaw and Colton, 2025). For training, the language model receives a transcription of note events, timing and pedal state as a flat sequence of integers, see Appendix C. Musical data is characterized by long-range temporal dependencies with repetition and variations and rich hierarchical structure in the form of melody, harmony, and rhythm without encoding the semantics of natural language (Berezovsky, 2019).

Protein sequences Proteins lie at the heart of vital biological functions and their amino-acid sequences encode highly complex molecular structures. We choose Swiss-Prot as a list of proteins (Boutet et al., 2007) and normalize the dataset to the 20 canonical amino acids, each a single token, by replacing any rare amino acids or unknowns with an additional symbol X.

Random As a control condition, we generate sequences with uniformly distributed length between 1 and 2,048 from an alphabet of uniformly independently sampled integers $k \in \{0, \ldots, 512\}$.

3.2 Language data

We use the dataset provided for the multilingual track of the BabyLM challenge. The dataset comprises cognitively plausible multilingual data according to the criteria outlined in Jumelet et al. (2026a). The exact composition of the dataset is listed in Appendix D. For stage II training, we sample English, Dutch, and Mandarin Chinese in proportion of 1/3 for each language.

Baselines Our primary baseline model is trained directly in stage II from random initialization, without any prior exposure to structural data. We additionally train a secondary baseline on English Wikipedia (Wikimedia Foundation, 2023) during stage I, in order to assess whether structural transfer provides benefits beyond those obtained from additional natural-language data.

4 Experimental setup

We run experiments on 80/20 data splits with a GPT-2 architecture (Radford et al., 2019). Early stopping (with patience of 3 epochs and a threshold of 0.01) is used to identify the optimal amount of structural and natural-language data, respectively. For hyperparameters and implementation details, refer to Appendix E. We evaluate the gains from structural transfer using the validation loss achieved by the trilingual model on a held-out portion of the natural-language corpus, together with performance on a combination of zero-shot and fine-tuning tasks from the BabyLM framework (Choshen et al., 2026), see Appendix F.

Tokenization We train a trilingual byte-pair-encoding (BPE) tokenizer with a fixed vocabulary of 16,897 tokens using the same approach to data sampling as outlined in Section 3.2. The tokenizer also includes 512 non-overlapping tokens used exclusively in stage I. At the start of stage II, we reinitialize the embedding layer to prevent spurious lexical transfer from the structural datasets.

Loss metrics We measure the performance of the model on the next-token prediction through the cross-entropy loss $\mathcal{L}(t)$ over the training tokens t.

The average loss change relative to the baseline is defined by the *loss ratio* ρ as the ratio of the areas below the loss curves up to N tokens of a condition $\mathcal{L}_i$ and the primary baseline $\mathcal{L}_0$ for training on natural language,

$$\rho_i = \frac{\int_{t_{\text{cutoff}}}^{N} \mathcal{L}_i(t)\,\mathrm{d}t}{\int_{t_{\text{cutoff}}}^{N} \mathcal{L}_0(t)\,\mathrm{d}t} - 1\,, \tag{1}$$

with a minimal cutoff at $t_{\text{cutoff}} = 2\,\text{M}$, to avoid inflating the metric due to large loss differences early in training.

A complementary metric, as used by Hu et al. (2025), is the *token efficiency* τ, i.e., the ratio of tokens necessary to achieve the final baseline loss with structural transfer,

$$\tau_i = \frac{1}{N}\left(N_i + \underset{0 \le n \le N}{\arg\min}\{\mathcal{L}_i(n) \le \mathcal{L}_0(N)\}\right), \tag{2}$$

where N_i is the number of steps we train on structural data in stage I for condition i. Intuitively, this captures how many tokens are necessary to train on structural plus language data to beat the baseline.

5 Results

We examine the effect of structural transfer from each condition along three dimensions: 1) next-token prediction loss on a multilingual corpus, 2) model-internal weight shifts during training, and 3) linguistic benchmarks. Together, these analyses allow us to assess to which extent structural transfer can aid language-model training.

5.1 Transfer on next-token prediction

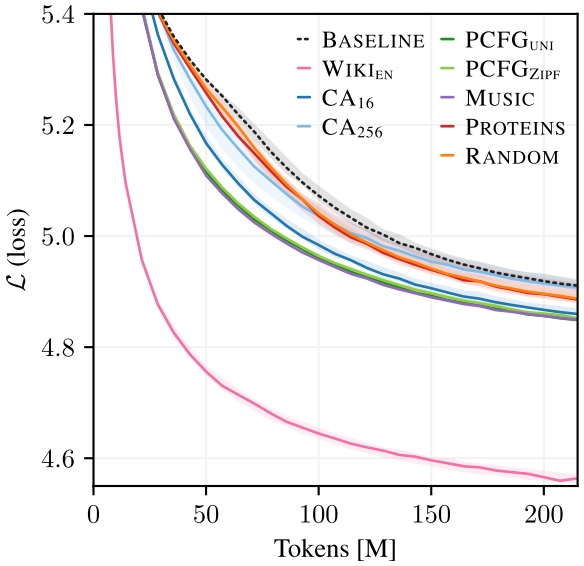


Figure 2: Cross-entropy loss $\mathcal{L}$ by training step of language models after having trained on structural data compared with the primary baseline of not training on any structure. The lines show the median over five seeds, except few runs that triggered early stopping. Shaded regions delineate minimum and maximum seed at each token count. Note that PCFG$_{UNI}$, PCFG$_{ZIPF}$, and MUSIC overlap.

Training on symbolic data can robustly improve the loss $\mathcal{L}$ on next-token prediction of natural language. Figure 2 shows that every condition matches or outperforms the baseline of not training on any structure (dashed black) in terms of achieving a lower loss. The clearest gains come from MUSIC, both synthetic grammars PCFG$_{UNI}$ and PCFG$_{ZIPF}$, and the cellular automata CA$_{16}$. Note that the two PCFG loss curves overlap, which implies that the Zipfian distribution does not benefit the structural transfer more than a uniform distribution. Although CA$_{256}$ yields an early advantage over the baseline, this condition converges to the baseline at 220 M tokens, suggesting that the noisier dynamics do not induce an equally useful initialization. PROTEINS and RANDOM lead to more modest but consistent improvements.

Overall, these results indicate that next-token-prediction loss can be robustly improved with a variety of symbolic data. However, the WIKI$_{EN}$ baseline puts these gains into perspective, showing that training on more natural language data for just one of the three languages from the multilingual corpus outperforms any of these symbolic data by a wide margin.

5.2 Model-internal representations

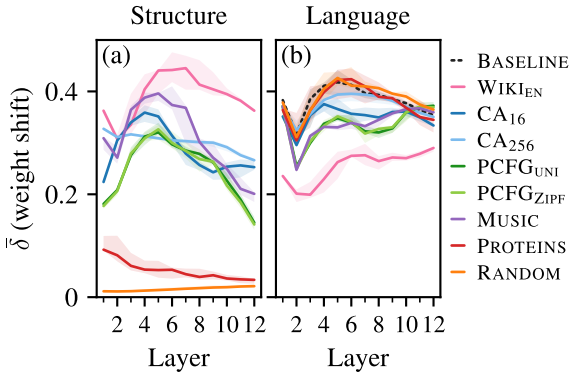


Figure 3: Weight shift $\bar{\delta}$ per layer in the self-attention matrices when training on (a) structure and subsequently (b) language. Median seed with shaded regions for minimum and maximum seed.

To understand how structural transfer impacts the model, we examine how much the weights shift in the two stages of training on structure and language, respectively. The relative weight shift from a time t before to after the respective training stage for layer l, component c, and seed s is

$$\delta_{l,c,s}^{(t)} = \frac{\|W_{l,c,s}^{\text{after}} - W_{l,c,s}^{\text{before}}\|_{\text{F}}}{\|W_{l,c,s}^{\text{before}}\|_{\text{F}}}, \tag{3}$$

where $\|\cdot\|_{\text{F}}$ is the Frobenius norm. We plot the mean across components c of the self-attention mechanism in the transformer architecture (query, key, value, and output matrices) per layer,

$$\bar{\delta}_{l,s}^{(t)} = \frac{1}{4}\sum_{c} \delta_{l,c,s}^{(t)}. \tag{4}$$

Shifting weights In Fig. 3 (a), showing stage I, we observe that PROTEINS and RANDOM induce much smaller weight shifts than the other better-performing conditions of stage II, which indicates that a random initialization is close to a local optimum for predicting these signals. Panel (b) shows how much the weights shift during stage II

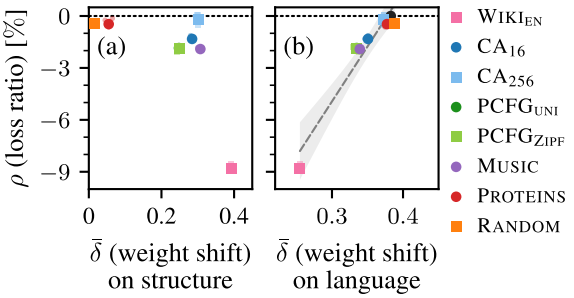


Figure 4: Relation of loss ratio on language data and weight shift, averaged over layers, for training on (a) structural data and (b) language data. Solid dots show the median, small transparent dots show each seed. PCFG and MUSIC perform best because the representations from learning on structure can be partially transferred to language, CA_{16} to a lesser extent, too. Dotted black line and black circle (b): baseline of training on language only. Dashed gray line in (b): ordinary-least-squares regression through condition-level medians; shaded band: 95 % confidence interval.

(language modeling). We see that training on sequences of PROTEINS and RANDOM numbers in stage I does not reduce the required weight shifts for modeling language compared to the baseline of no structural training. The weights of models trained on synthetic grammars, MUSIC, and CA_{16} change less than the baseline with an average weight shift of around $\bar{\delta} = 0.34$ across layers. Notably, CA_{256} is an exception. The weights change strongly in the structural-learning phase, but the resulting attention pattern is not beneficial for language modeling, that is, the required weight shift is as large as for the randomly initialized baseline.

Relating weights and loss In Fig. 4, we visualize the loss reduction relative to the weight shift. The respective weight shift correlates with loss improvement on language (panel (b) with Pearson correlation $r = 0.96, p < 0.01$) but less so on structural data (panel (a) with $r = -0.60$, $p = 0.11$). The smallest weight shift achieved in stage II is associated with the biggest loss improvement for the English Wikipedia baseline. These results indicate that the weights optimized for predicting structural data facilitate learning language by requiring a smaller weight shift.

Moreover, the best conditions achieve a token efficiency of approximately 60 % of the original token budget, meaning they require only about 60 % of the baseline token budget to reach equivalent language-modeling performance. The loss ratio ρ measures how much the training on structural data improves the cross-entropy loss (lower is better) on average over training steps and this ratio is closely related to token efficiency, see Appendix G.

5.3 Linguistic benchmarks

We employ the framework of the BabyLM challenge, which combines zero-shot and fine-tuning evaluation in English, Dutch, and Chinese (Choshen et al., 2026), covering testing suites for linguistic competence (BLiMP-style tasks), reading comprehension, knowledge probing, and others. For details on the exact tasks, refer to Table 8 in Appendix F.

The results show that the observed loss reduction on the language modeling objective does not translate to better performance on linguistic benchmarks. Figure 5 shows the evaluation results for several zero-shot and fine-tuning tasks. The only stage-I condition that robustly improves the baseline results for English and Dutch is training on English texts. All other structural-transfer conditions do not significantly affect the average evaluation results positively nor negatively.

Across languages, CA_{16} has a small positive effect. We note that some conditions improve or worsen the performance exclusively on fine-tuning or zero-shot. This inconsistency might indicate that the linguistic benchmarks cannot resolve the small differences induced by the structural data or that zero-shot and fine-tuning tasks are not entirely aligned, measuring different linguistic aspects. Even if some structural data tends to lead to an improvement, the magnitude of this effect appears small, on the order of one percentage point. We conclude that an improvement on the target objective (next-token prediction) does not generally entail an improvement on linguistic benchmarks.

Detailed effects We fit a Bayesian hierarchical model to estimate the effect of each structural-transfer condition relative to the natural language baseline. Benchmark scores (averaged across seeds) are modeled as $\text{score}_{ctl} \sim \mathcal{N}(\mu_c + u_t, \sigma)$ for each language and as $\text{score}_{ctl} \sim \mathcal{N}(\mu_c + u_t + u_l, \sigma)$ for the total, with a fixed effect μ_c per condition, a random effect u_t per task, and a random effect per language u_l. We report the posterior mean difference $\Delta_c = \mu_c - \mu_{\text{baseline}}$ relative to the effect of the primary baseline in percentage points with 95 % highest-density intervals (HDI) and posterior probability $p(\Delta_c > 0)$, estimated via NUTS sampling (Hoffman and Gelman, 2014; Abril-Pla et al.,

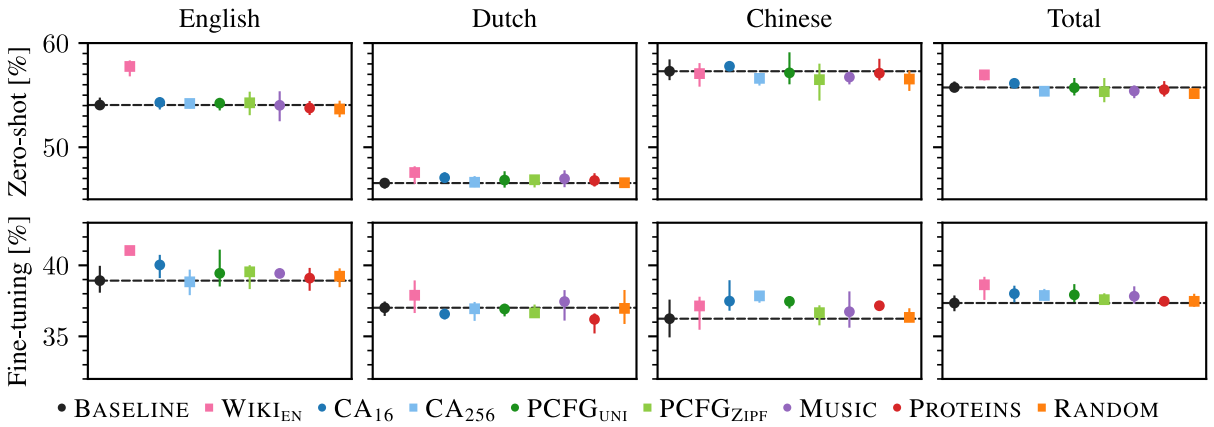


Figure 5: Average accuracy on linguistic zero-shot and fine-tuning benchmarks for English, Dutch, Chinese, and averaged over these three languages, see Table 8. The dashed line denotes the baseline of not training on structural data. There are no significant changes relative to the baseline for the total, except WIKI$_{EN}$ and CA$_{16}$. Dots show the median and error bars indicate the minimum and maximum seed.

2023) with four chains of 2,000 draws.

Figure 6 shows posterior estimates for all conditions across languages combined. Training on Wikipedia in stage I has robust positive effects on both zero-shot ($\Delta = +1.2$ pp, HDI: $[0.6, 1.9]$, $p = 1.00$) and fine-tuning ($\Delta = +1.3$ pp, HDI: $[-0.1, 2.7]$, $p = 0.97$) evaluations. The RANDOM condition shows a small negative effect on zero-shot performance ($\Delta = -0.6$ pp, HDI: $[-1.2, 0.1]$, $p = 0.04$). The CA$_{16}$ has a small positive effect on both zero-shot and fine-tuning performance ($\Delta = 0.4$ pp, HDI: $[-0.2, 1.1]$, $p = 0.88$ and $\Delta = 0.7$ pp, HDI: $[-0.8, 2.0]$, $p = 0.83$, respectively). All structural data have effects close to zero with HDIs that include zero.

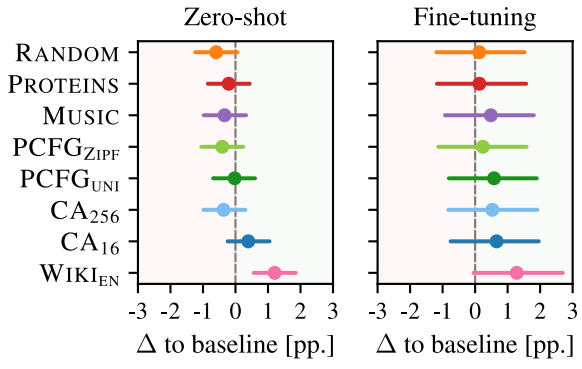


Figure 6: Mean and 95 % highest-density intervals estimated by a Bayes model for linguistic benchmarks of zero-shot and fine-tuning across languages.

Based on these results, we performed a hyperparameter search to train the submission model for the BabyLM challenge on a mixture of the most beneficial data types, see Appendix H for details. However, even the best model (in terms of validation loss) does not perform much better than the separately trained models.

Syntactic Tasks We hypothesize that structural transfer is more beneficial for syntax-level tasks due to its hierarchical dependencies. To check this, we select tasks explicitly marked as "syntactic" from English BLiMP (Warstadt et al., 2020), BLiMP-NL (Suijkerbuijk et al., 2025) and ZhoBLiMP (Liu et al., 2026). These tasks span syntactic features such as argument structure, ordering in wh-clauses, islands constraints, etc. However, both per-task and aggregated category results per data type do not improve over the baseline by more than two percentage points.

6 Discussion

In summary, our results indicate that learning structural patterns enables faster performance gains on the next-token prediction task. However, these dynamics do not translate to better performance in linguistic benchmarks (Chen et al., 2025). Rather, transfer depends on the task at hand: Both next-token prediction and linguistic benchmarks measure language competence yet via different metrics, so that optimizing for next-token prediction does not necessarily imply measurable transfer effects on linguistic tasks. Finding that none of the tested structural conditions match the efficiency of additional natural-language data, we conclude that the intricate correlational structures of language cannot easily be replaced by synthetic data.

Given that the two PCFG variants perform

equally, a Zipfian distribution alone does not appear to impact transfer significantly. This may be because a Zipfian distribution is a first-order approximation of the correlational structure, i.e., just unigrams, which are learned quickly when training on natural language.

Although protein sequences encode highly complex biological information, this structure does not appear to be readily accessible to a transformer, at least not under our naive tokenization approach used for the next-token prediction task of one token per amino acid. Since even random numbers perform slightly better than a randomly initialized model without any structural training, this might indicate that even a very simple tweak to the weight initialization can improve training results.

The structural data may initialize the weights to functionally useful representations leading to successful structural transfer. These representations might be interesting to explore in connection with the work on discovering capabilities emerging in attention heads during pretraining (Olsson et al., 2022; Elhage et al., 2021; Crosbie and Shutova, 2025; Aoyama et al., 2026), in which a learned matrix structure facilitates functions such as token copying or retrieval.

In some respects, the well-performing data are language-like. For example, music has highly complex structure produced by humans, including recursions and a form of syntax. PCFGs explicitly model syntactic structure extracted from natural-language texts. In cellular automata, on the other hand, complexity emerges from simple rules. However, any structural data we train on lacks the semantic content of language. Thus, it would be interesting to further explore the degree to which semantics are helpful and necessary for transfer.

7 Conclusion

We investigated whether structural transfer can improve sample-efficient multilingual language modeling. Our results show that training on structured symbolic data before natural language can improve next-token prediction learning efficiency. The weight-shift analysis supports this interpretation: Models that benefit most from structural transfer require smaller parameter changes during multilingual language training, indicating that prior structural exposure places models in a more favorable region of the parameter space.

However, downstream gains are limited. Improvements on the language-modeling objective do not reliably translate into stronger performance on downstream linguistic benchmarks. Moreover, when additional natural language data is available, it remains more efficient than structural data, even when training on a mix of partially typologically distant languages such as English, Dutch, and Chinese. Although this may be useful when language data is limited, such as for low-resource languages or other data-constrained settings, structural transfer is not a direct substitute for relevant language input.

More broadly, our findings suggest that structural data provides a direction for revisiting initialization strategies in language models. Carefully tailoring model weights before training may provide a useful head start, particularly in the context of limited compute and model capacity.

Limitations

Selected structures Although we aim for a wide range of structural data (synthetic grammars, music, proteins, CAs) that we expected a priori to possibly yield strong transfer to natural language, resembling language to varying degrees, we cannot account for all possible data types exhaustively.

Model size We conduct our experiments on the GPT-2-small architecture. While this setup provides us with a reasonable overview of structural transfer, a variation of the model size could clarify the limits of the loss improvements on next-token prediction.

Data-mixing effects We do not exhaustively explore the effects of data mixing during stage I. A wider variety of high-quality data in stage II would allow for a more detailed study of language-specific differences in structural transfer. Since the Wikipedia baseline is in English only, we cannot assess the benefits of structural transfer across different natural languages. Observing effects of structural transfer tailored to a specific language would be an interesting avenue for future research.

Representations We evaluate transfer by analyzing the magnitude of the weight shift. A more detailed analysis through the lens of mechanistic interpretability could target specifically the representations that the model learns on the structural data and possibly reuses on natural language.

Ethical considerations

While we aim to advance the design of cognitively inspired models, we do not speculate here on how human language processing works. Explicit parallels between the processing capabilities of models and human cognition tend to be strained easily.

Acknowledgments

We thank the reviewers for their feedback. We thank Alexander Ecker and Edoardo Ponti for helpful discussions. This research is partially supported by the zukunft.niedersachsen program of the VolkswagenStiftung (L.B., Y.V.) and by a VENI grant (Vl.Veni.211C.039) from the Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO) (L.B.).

Author contributions

Y.V.: Supervision, Conceptualization, Analysis, Software (training), Literature review, Writing—original draft (Abstract, Introduction, Related work, Conclusion, parts of Methods), Writing—review & editing. **J.M.M.:** Supervision, Conceptualization, Analysis, Software (data and analysis), Literature review, Visualization, Writing—original draft (Results and Discussion; parts of Introduction, Related work, and Methods; App. on CAs and music), Writing—review & editing. **J.L..:** Software (PCFG data), Writing—original draft (PCFGs). **L.B.:** Supervision, Conceptualization, Analysis, Writing—review & editing.

References

Oriol Abril-Pla, Virgile Andreani, Colin Carroll, Larry Dong, Christopher J. Fonnesbeck, Maxim Kochurov, Ravin Kumar, Junpeng Lao, Christian C. Luhmann, Osvaldo A. Martin, Michael Osthege, Ricardo Vieira, Thomas Wiecki, and Robert Zinkov. 2023. PyMC: A modern, and comprehensive probabilistic programming framework in python. *PeerJ Computer Science*, 9:e1516.

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Tatsuya Aoyama, Ethan Wilcox, and Nathan Schneider. 2026. Predicting the formation of induction heads. In *First Workshop on CogInterp: Interpreting Cognition in Deep Learning Models*, San Diego, CA, USA. Curran Associates, Inc.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The belebele benchmark: A parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

Jesse Berezovsky. 2019. The structure of musical harmony as an ordered phase of sound: A statistical mechanics approach to music theory. *Science Advances*, 5(5):eaav8490.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*, pages 31–es, Barcelona, Spain. Association for Computational Linguistics.

Emmanuel Boutet, Damien Lieberherr, Michael Tognolli, Michel Schneider, and Amos Bairoch. 2007. UniProtKB/Swiss-Prot. In David Edwards, editor, *Plant Bioinformatics*, volume 406 of *Methods in Molecular Biology*, pages 89–112. Humana Press, Totowa, NJ, USA.

Louis Bradshaw and Simon Colton. 2025. Aria-MIDI: A dataset of piano MIDI files for symbolic music modeling. In *The Thirteenth International Conference on Learning Representations*, Singapore. ICLR.

Tyler A. Chang et al. 2026. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *Preprint*, arXiv:2510.24081.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025. BabyLM turns 3: Call for papers for the 2025 BabyLM workshop. *Preprint*, arXiv:2502.10645.

Yangyi Chen, Binxuan Huang, Yifan Gao, Zhengyang Wang, Jingfeng Yang, and Heng Ji. 2025. Scaling laws for predicting downstream performance in LLMs. *Transactions on Machine Learning Research*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind

Tafjord. 2018. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Joy Crosbie and Ekaterina Shutova. 2025. Induction heads as an essential mechanism for pattern matching in in-context learning. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5034–5096, Albuquerque, NM, USA. Association for Computational Linguistics.

Richard Diehl Martinez, Hope McGovern, Zebulon Goriely, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2023. CLIMB – Curriculum learning for infant-inspired model building. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 84–99, Singapore. Association for Computational Linguistics.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. Technical Report 12, Anthropic.

Martin Gardner. 1970. Mathematical games. *Scientific American*, 223(4):120–123.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 249–256, Sardinia, Italy. Journal of Machine Learning Research.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, Santiago, Chile. IEEE.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schuetze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81, Vienna. Austria. Association for Computational Linguistics.

Matthew D Hoffman and Andrew Gelman. 2014. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2176, pages 30016–30030, New Orleans, LA, USA. Curran Associates, Inc.

Hai Hu, Yunhao Zhang, Hong'ao Zhu, Siyuan Song, Rui Wang, Renfen Hu, Linyang He, Luan Li, Xiaozhe Ji, Shaonan Wang, Zhizheng Qian, and Yingxin Lin. 2026. Evaluation Pipeline.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. Between circuits and Chomsky: Pre-pretraining on formal languages imparts linguistic biases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9691–9709, Vienna, Austria. Association for Computational Linguistics.

Liangze Jiang, Zachary Shinnick, Anton van den Hengel, Hemanth Saratchandran, and Damien Teney. 2026. Procedural pretraining: Warming up language models with abstract data. *Preprint*, arXiv:2601.21725.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Jaap Jumelet and Willem Zuidema. 2023. Transparency at the source: Evaluating and interpreting language models with access to the true distribution. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4354–4369, Singapore. Association for Computational Linguistics.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual BERT: An empirical study. In *The Eighth International Conference on Learning Representations 2020*, Online. Curran Associates, Inc.

Victor Kuperman, Sascha Schroeder, Cengiz Acartürk, Niket Agrawal, Dominick M. Alexandre, Lena S. Bolliger, Jan Brasser, César Campos-Rojas, Denis Drieghe, Dušica Filipović Đurđević, Luiz Vinicius Gadelha De Freitas, Sofya Goldina, et al. 2025. New data on text reading in English as a second language: The Wave 2 expansion of the Multilingual Eye-Movement Corpus (MECO). *Studies in Second Language Acquisition*, 47(2):677–695.

Dan Lee, Seungwook Han, Akarsh Kumar, and Pulkit Agrawal. 2026. Training language models via neural cellular automata. *Preprint*, arXiv:2603.10055.

Hung-yi Lee, Shang-Wen Li, and Thang Vu. 2022. Meta learning for natural language processing: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 666–684, Seattle, WA, USA. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, and 2 others. 2022b. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in Chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Christopher D. Manning and Hinrich Schütze. 2000. Probabilistic parsing. In *Foundations of Statistical Natural Language Processing*, 2nd edition, pages 407–460. MIT Press, Cambridge, MA, USA.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank:. *Computational Linguistics*, 19(2):313–330.

Jonas Mayer Martins, Ali Hamza Bashir, Muhammad Rehan Khalid, and Lisa Beinborn. 2025. Once upon a time: Interactive learning for storytelling with small language models. In *Proceedings of the First BabyLM Workshop*, pages 454–468, Suzhou, China. Association for Computational Linguistics.

Sebastian J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4975–4989, Florence, Italy. Association for Computational Linguistics.

Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. 2020. Growing neural cellular automata. *Distill*, 5(2).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning music helps you read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839, Online. Association for Computational Linguistics.

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8402–8413, Singapore. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Wessel Poelman, Thomas Bauwens, and Miryam de Lhoneux. 2025. Confounding factors in relating model performance to morphology. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7273–7298, Suzhou, China. Association for Computational Linguistics.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2023. Cross-lingual consistency of factual knowledge in multilingual language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10650–10666, Singapore. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Max Reger. 1900. Der Mond ist aufgegangen. In *Sieben Geistliche Volkslieder*, WoO VI/14. Transcribed by the authors.

Jonas Rohweder, Subhabrata Dutta, and Iryna Gurevych. 2026. Hierarchical latent structures in data generation process unify mechanistic phenomena across scale. *Preprint*, arXiv:2603.06592.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A Haggag, Imanol Schlag, Marzieh Fadaee, Sara Hooker, and Antoine Bosselut. 2025. INCLUDE: Evaluating multilingual language understanding with regional knowledge. In *The Thirteenth International Conference on Learning Representations 2025*, pages 83291–83322, Singapore. Curran Associates, Inc.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Jenny R. Saffran and Natasha Z. Kirkham. 2018. Infant statistical learning. *Annual Review of Psychology*, 69(1):181–203.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. WinoGrande: An adversarial Winograd Schema Challenge at scale. *Communications of the ACM*, 64(9):99–106.

Chen Shani, Yuval Reif, Nathan Roll, Dan Jurafsky, and Ekaterina Shutova. 2026. The roots of performance disparity in multilingual language models: Intrinsic modeling difficulty or design choices? *Preprint*, arXiv:2601.07220.

Zachary Shinnick, Liangze Jiang, and Hemanth Saratchandran. 2025. Transformers pretrained on procedural data contain modular structures for algorithmic reasoning. In *Methods and Opportunities at Small Scale (MOSS)*, Vancouver, Canada. Proceedings of Machine Learning Research.

Noam Siegelman, Sascha Schroeder, Cengiz Acartürk, Hee-Don Ahn, Svetlana Alexeeva, Simona Amenta, Raymond Bertram, Rolando Bonandrini, Marc Brysbaert, Daria Chernova, Sara Maria Da Fonseca, Nicolas Dirix, and others. 2022. Expanding horizons of cross-linguistic research on reading: The Multilingual Eye-movement Corpus (MECO). *Behavior Research Methods*, 54(6):2843–2863.

Noam Siegelman, Sascha Schroeder, Yaqian Borogjoon Bao, Cengiz Acartürk, Niket Agrawal, Lena S. Bolliger, Jan Brasser, César Campos-Rojas, Denis Drieghe, Dušica Filipović Đurđević, Sofya Goldina, Romualdo Ibáñez Orellana, and others. 2025. Wave 2 of the Multilingual Eye-Movement Corpus (MECO): New text reading data across languages. *Scientific Data*, 12(1):1183.

Vésteinn Snæbjarnarson, Annika Simonsen, Goran Glavaš, and Ivan Vulić. 2023. Transfer to a low-resource language via close relatives: The case study on Faroese. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, Tórshavn, Faroe Islands. University of Tartu Library.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2022. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565.

Martin Spitznagel and Janis Keuper. 2026. A new kind of network? Review and reference implementation of neural cellular automata. *Preprint*, arXiv:2604.24990.

Michelle Suijkerbuijk, Zoë Prins, Marianne De Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

John von Neumann and Arthur Walther Burks. 1966. *Theory of Self-Reproducing Automata*, 1st edition. University of Illinois Press, Urbana, IL, USA.

Mingyang Wang, Heike Adel, Lukas Lange, Jannik Strötgen, and Hinrich Schuetze. 2023. GradSim: Gradient-based language grouping for effective multilingual training. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4631–4646, Singapore. Association for Computational Linguistics.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–6, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Wikimedia Foundation. 2023. English Wikipedia 2023-11-01.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, LA, USA. Association for Computational Linguistics.

Stephen Wolfram. 1983. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55(3):601–644.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. 2025. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. In *The Thirteenth International Conference on Learning Representations 2025*, pages 82263–82287, Singapore. Curran Associates, Inc.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Synthetic grammars

We induce a probabilistic context-free grammar from the UD-Treebank annotation. The grammar is delexicalized, i.e., the terminal symbols are word-class categories. We then create data by sampling parse trees from the grammar and instantiating word classes with vocabulary IDs, see Fig. 7. The two types PCFG_{UNI} and $\text{PCFG}_{\text{ZIPF}}$ differ only in the frequency with which the words are sampled in the last step. Table 1 summarizes shared settings; Table 2 details the vocabulary layout.

Parameter	Value
Corpus size	200 M tokens
Grammar	Penn Treebank PCFG
Max. parse depth	16
Sampling ($\text{PCFG}_{\text{zipf}}$)	Zipf ($\alpha = 1$)
Sampling (PCFG_{uni})	uniform

Table 1: PCFG hyperparameters.

English-based Grammar Induction and Normalization. We derive a PCFG from the Wall-Street-Journal portion of the English Penn Treebank that is distributed with NLTK (Marcus et al., 1993; Bird and Loper, 2004) by estimating relative frequencies. The grammar is entirely delexicalized during construction by removing word-level rules, resulting in 2,664 structural productions, 26 non-terminal symbols, and 22 terminal categories. Part-of-speech tags on the right-hand side are collapsed into coarse abstract categories, such as grouping various noun tags into a single noun category. Punctuation is mapped to dedicated terminal categories (e.g. `period`, `comma`), rather than being collapsed into a single punctuation class. Functional tags and trace indices are stripped from non-terminal names to merge similar Treebank labels, and productions containing empty nodes are excluded.

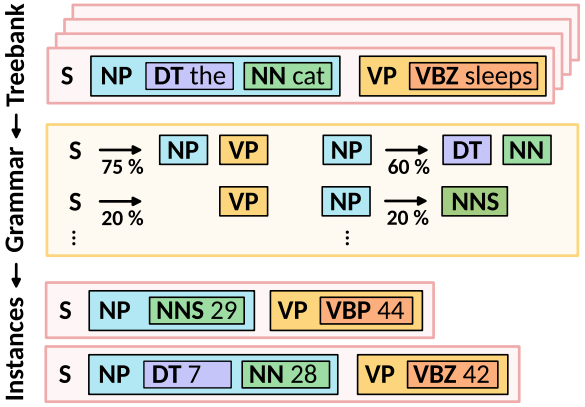


Figure 7: From the treebank, which consists of sentences with part-of-speech tags, we derive the transition probabilities, e.g., a sentence `S` splits into a noun phrase `NP` and a verb phrase `VP` or into only a verb phrase. Similarly, a noun phrase can split into a determiner `DT` and a noun `NN` or into a plural noun `NNS`. From the derived grammar, we can sample instances again, but the lexicalization is not semantic, represented by abstract integers instead. The model receives only the sequence integers, without part-of-speech tags.

After merging duplicate productions, the probabilities are normalized separately for each left-hand-side non-terminal. To guarantee that generation can terminate at a specified depth, some non-terminals require a fallback rule. For any non-terminal lacking a production that expands exclusively to terminal categories, we add a single fallback production with a raw probability of 10^{-6}. This fallback is included in the left-hand side group prior to normalization, allowing it to retain a small relative probability while slightly reducing the mass of the other rules.

Tree generation and depth constraints. Parse trees are generated via stochastic top-down sampling from the induced PCFG, starting at the root symbol `S`. At each non-terminal node, a production is drawn with a probability proportional to its weight among the rules for that symbol, and expansion continues until every leaf is a terminal category. For example,

S → NP VP
NP → det noun
VP → verb NP
NP → noun

⇒ det noun verb noun

To keep the trees finite, we enforce a maximum depth limit of 16, counting the root as depth 0. During expansion at depths 0 through 14, any available production for a given symbol may be used. When the sampler reaches depth 15, the active rule set is restricted strictly to terminal-only productions. This intervention truncates very deep structures and slightly shifts the generated distribution compared to the unbound grammar.

Lexicon layout and token sampling The 512 integer IDs are shuffled using a fixed random seed and partitioned into disjoint slices for each syntactic category, see Table 2. Slot sizes are derived from Penn Treebank *type* counts (distinct word forms per category), not from token frequencies in running text. To account for the fact that only a small portion of the vocabulary serves as function words, we unevenly distribute the 512 vocabulary IDs into 74 slots ($\approx 15\,\%$) for closed-class terminals and 438 slots for open-class terminals.

Once the tree is fully expanded, each terminal leaf is mapped to a concrete ID from the respective vocabulary slice. Under a Zipfian condition, tokens within a slice are sampled with weights proportional to their inverse rank using a global exponent of $\alpha = 1$. Under the uniform condition, every token within the slice has an equal selection probability.

Parameter	Value
Number of categories	22 (disjoint slices)
Vocabulary size	512 opaque IDs
Open / closed budget	438 / 74 (85 % / 15 %)
Allocation within budgets	PTB type counts
Largest closed slice	num (51)
Within-slice sampling	Zipf (α=1) or uniform

Table 2: Lexicon layout.

Final Output and Corpus Statistics. Each generated tree yields one sequence of tokens by extracting the terminal IDs in left-to-right order. The final output consists of simple integer sequences ranging from 0 to 511. We stop the generation process at exactly 200,000,000 tokens per corpus, possibly truncating the last sentence. This procedure results in 572,648 sentences for each file and a mean sequence length of about 349.

B Generating cellular automata

To illustrate the mechanics of cellular automata, consider the transition rule in Fig. 8: The update pattern $f(\blacksquare\blacksquare\blacksquare) = \square\blacksquare\square$ maps this three-cell neighborhood to the state $\blacksquare$ in the next time step.

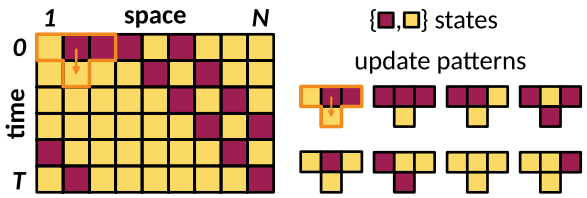


Figure 8: A cellular automaton consists of a row of cells, each in one of, e.g., $K = 2$ states $\{\blacksquare, \square\}$. The next time step is determined by applying a set of update patterns that define the *transition rule* of the cellular automaton to that row.

The rule space of classical cellular automata with K states consists of K^{K^3} rules, which is astronomically large even for $K = 16$, and many of the rules produce either fixed points or uncorrelated noise. Rather than sampling this space uniformly, we parameterize transition rules through a small neural network f_θ. Each set of randomly sampled, frozen parameters θ defines a rule.

This approach is known as neural cellular automata (NCA) (Mordvintsev et al., 2020; Spitznagel and Keuper, 2026). By defining a suitable probability distribution for sampling the network parameters θ, we avoid both entirely noisy and homogeneous dynamics, that are expected to be less beneficial for structure learning. Our setup of using NCA for structural transfer in language models is inspired by Lee et al. (2026) but uses a one-dimensional grid, a simplified architecture, and deterministic (argmax) updates for efficiency and interpretability.

Every NCA still defines a classical CA rule. Given the fixed parameters θ, one can simply enumerate the output of the NCA f_θ for all the three-cell neighborhoods to generate a classical lookup table. The neural parameterization is thus purely used to steer the data generation, not changing the nature of the simulated CAs but only which rules are likely to be drawn.

B.1 Technical implementation

In our implementation, each rule is defined by a randomly sampled, fixed parameter set θ, see Ta-

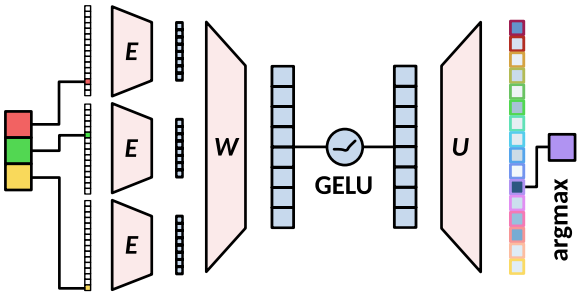


Figure 9: Schematic of the NCA architecture f_θ, which takes a three-cell neighborhood as input and outputs the next state for the middle cell. The middle cell in state ■ is updated based on itself and its neighbors ■ and ■ through the following steps: One-hot encoding of the state, projection E of this vector to a low-dimensional representation, then a convolution with a matrix W, applying a nonlinearity (GELU), unembedding E and taking the argmax over this vector to arrive at the next cell state ■ → ■.

ble 3. These parameters define a function f_θ that is applied to every cell and its 3-cell neighborhood with periodic boundary conditions. This function is a neural network, which updates a cell i to the state $c_i^{(t)}$ at time t.

Architecture The architecture for generating NCA is a simple sequence of three functional steps, which suffice to give rise to interesting dynamics by inducing a bottleneck, convolution, and nonlinearity on the input data. Figure 9 visualizes this architecture schematically.

First, the cell states (a one-hot encoding of the K integers, i.e., the vocabulary) are projected into a low-dimensional representation with an embedding matrix $E \in \mathbb{R}^{d \times K}$, such that

$$\mathbf{e}_j = E\mathbf{1}_{c_j^{(t)}} \tag{5}$$

for $j \in \{i-1, i, i+1\}$.

Second, these three embedding vectors $\mathbf{e}_j$ are concatenated. A matrix $\mathbf{W}$ and bias $\mathbf{b}$ act as a convolution of the three cell states, followed by a GELU nonlinearity to arrive at a vector

$$\mathbf{h}_i = \mathrm{GELU}(W[\mathbf{e}_{i-1} \| \mathbf{e}_i \| \mathbf{e}_{i+1}] + \mathbf{b}) \tag{6}$$

where $\|$ means concatenation of the vectors. Finally, we apply an unembedding matrix U to arrive at a distribution over K states and take the argmax to select an output state,

$$c_i^{t+1} = \arg\max_{k \in K}(U\mathbf{h}_i)\,. \tag{7}$$

The rules are thus fully deterministic given the parameters and an initial condition. For an initial condition $\mathbf{c}^{(0)}$, the update rule $\mathbf{c}^{(t+1)} = f_\theta(\mathbf{c}^{(t)})$ defines a trajectory $\tau = \left(\mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(T-1)}\right)$ with T time steps. The data that the language model receives is a row-wise flattened version of the trajectories τ, separated by a token for the end of a sequence.

Generation For each dataset, we sample 100,000 simulated trajectories from about 1,820 rules. Each rule instantiates a neural cellular automaton. The parameters θ are sampled from distributions scaled by a parameter s to elicit a variety of dynamical regimes, see Table 3. We generate two dataset variants, CA_{16} and CA_{256}, differing in the number of states K, embedding dimension d, and hidden dimension h, see Table 4. We tuned the hyperparameters for generating CA_{16} manually to instill an initial bias favoring a medium range of entropy, which we hypothesize to be the most promising, cf. the spectrum of NCA visualized with four examples in Fig. 1. The variant CA_{256} contrasts this with a larger vocabulary and more noisy dynamics.

We also randomly vary the grid size N, trajectory length T, and number of trajectories per rule for more variety in the dataset, to prevent the model from overfitting on a specific periodicity in the data, see Table 5. Each trajectory is initialized in a state $\mathbf{c}^{(0)} \in \{0, \ldots, K-1\}^N$ by drawing uniformly from four schemes: (i) *uni*, which draws each cell i.i.d. from the state set, $\mathcal{U}\{0, \ldots, K-1\}$; (ii) *gradient*, which interpolates linearly between two independently chosen states; (iii) *block*, which creates regions of length $\ell \sim \mathcal{U}\{1, \lfloor N/7 \rfloor\}$, each with a single random state; and (iv) *sparse*, which picks a uniform background state with a random fraction $\rho \sim \mathcal{U}(0.01,\ 0.2)$ of cells replaced by random states.

	Parameter	Shape	Distribution
s	Weight scale	scalar	$\log(s)$ $\sim \mathcal{U}(\log(0.1),\ \log(3))$
E	Embedding	$d \times K$	$\mathcal{N}(0,\ s^2)$
W	Conv. weights	$h \times 3d$	$\mathcal{N}(0,\ s^2/(3d))$
$\mathbf{b}$	Conv. bias	h	$\mathcal{N}(0,\ (0.2s)^2)$
U	Unembedding	$K \times h$	$\mathcal{N}(0,\ s^2/h)$

Table 3: Parameter initialization for each randomly sampled NCA rule θ. The weight scale s is shared across all parameters of a given rule and controls the dynamical regime of the resulting automaton.

	Hyperparameter	CA_{16}	CA_{256}
K	Number of states	16	256
d	Embedding dimension	8	16
h	Hidden dimension	8	6
	Neighborhood size	3	3

Table 4: Architecture hyperparameters for the two variants of the NCA dataset.

	Statistic	Value
	Number of rules (expected)	≈1,820
N	Grid size	$\mathcal{U}\{10, 100\}$
T	Trajectory length	$\mathcal{U}\{10, 100\}$
	Trajectories per rule	$\mathcal{U}\{10, 100\}$
	Total trajectories	100,000

Table 5: Statistics for generating the NCA datasets. Grid size N, trajectory length T, and trajectories per rule are sampled independently per rule.

C Music

We use the deduplicated "Unique" subset of the ARIA-MIDI dataset (Bradshaw and Colton, 2025), comprising 32,522 piano pieces in MIDI format. We convert each piece into an integer sequence for the language model to train on.

Event encoding From the MIDI data, we extract three event types: 1) ON means a note begins (followed by two integers expressing pitch and velocity); 2) OFF means a note ends (followed by an integer for the pitch, respectively); (3) PEDAL_ON and PEDAL_OFF mean the sustain pedal changes state. A SHIFT token precedes events not simultaneous with the previous one, encoding elapsed time measured in bins of 10 milliseconds, effectively moving the tape head forward in time. Left and right hand are a single event stream.

Integer mapping Each event type above is assigned a contiguous integer range for transcribing the events into a sequence of numbers. The four command tokens ON, OFF, PEDAL_ON, and PEDAL_OFF are encoded as 0 to 3, respectively. Pitch (MIDI range 0–127) maps to the range 4–131. Velocity (MIDI range 0–127) is quantized in 25 steps of size 5, mapping to 132–156. Time shifts are binned in steps of 10 milliseconds into 127 bins, mapping to 157–283. The full vocabulary thus contains 284 tokens. Figure 10 illustrates the encoding on a short excerpt.

D BabyLM natural-language data

The multilingual training data from the BabyLM corpus is listed in Table 6.

Language	Domain	Ratio (%)
Chinese	Conversational	79 %
	Educational	9 %
	Written/Books	12 %
Dutch	Conversational	3 %
	Educational	17 %
	Written/Books	16 %
	Subtitles	1 %
	Fallback/Other	63 %
English	Conversational	37 %
	Written/Books	42 %
	Fallback/Other	21 %

Table 6: Within-language domain mix of the BabyBabelLM Chinese, Dutch, and English corpora (Jumelet et al., 2026a). Total sizes: 138 M, 110 M, 99 M tokens, respectively. Percentages sum to 100 % per language.

E Training regime

We keep hyperparameters fixed across different structural data types and training stages, listed in the Table 7. Our Python environment uses `torch==2.12.0`, `transformers==4.47.0`, `datasets==3.0.0`, `flash_attn==2.83`. All experiments were performed on a single NVIDIA A100 80GB GPU for a total of ≈ 250 GPU hours.

F Evaluation

We use the evaluation pipeline of the 2026 BabyLM Challenge (Choshen et al., 2026). Table 8 lists the multilingual evaluation data for zero-shot and finetuning tasks.

G Metric correlation

The two evaluation metrics of loss ratio ρ and token efficiency τ are strongly related, Fig. 11, yet loss ratio is more robust to outliers due to early stopping. Token efficiency is highly sensitive to the chosen cutoff point in training steps. Integrating over the loss curve instead, allows for a more robust interpretation of the results. This robustness is due to the integration of the loss curve, which solves the problem of sensitivity to the cutoff point in training steps that token efficiency suffers from.

H Challenge submission

For the challenge submission, we train our model on an equal mix of the top three structural data

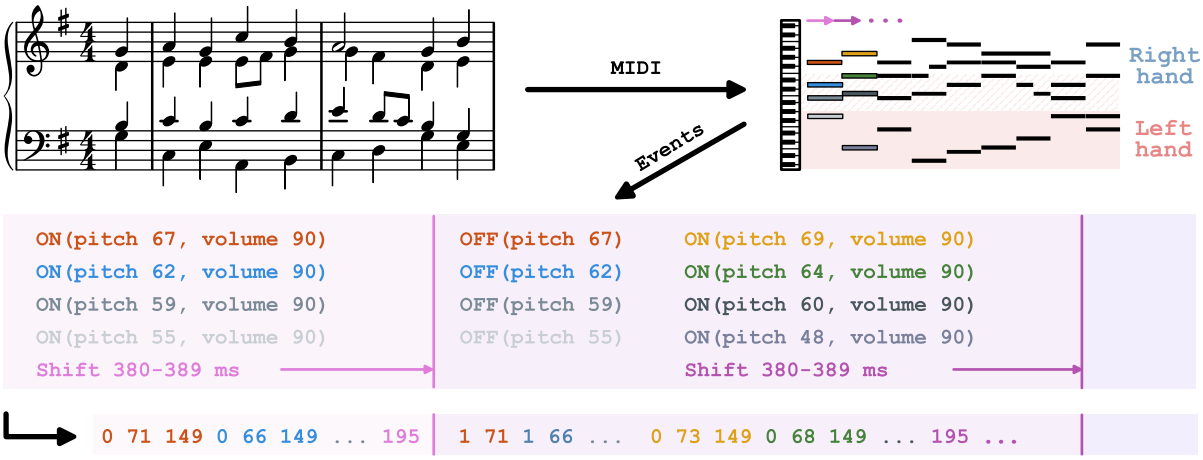


Figure 10: Piano pieces are encoded as MIDI format in the ARIA-MIDI dataset. We extract the pitch, volume, pedal state, and shifts of the tape head and map each event to integers. This examples shows the beginning of Reger (1900).

Hyperparameter	Value
Model	
Layers	12
Hidden size	768
Attention heads	12
Context length	512
Vocabulary size	16,897
Activation	GELU
Dropout	0.1
LayerNorm ϵ	1×10^{-5}
Initializer range	0.02
Training	
Sequence length	512
Batch size	16
Optimizer	AdamW
Optimizer learning rate	1×10^{-4}
Weight decay	0.05
Warmup ratio	0.01
Gradient clipping	1.0
Early stopping patience	3
Early stopping delta	0.01
Seed	11; 17; 42; 2,000; 3,407
Fine-tuning	
Learning rate	5×10^{-5}
Weight decay	0.05
Batch size	16
Sequence length	128
Early stopping patience	3
Seed	12

Table 7: Training and optimization settings for the main experiments.

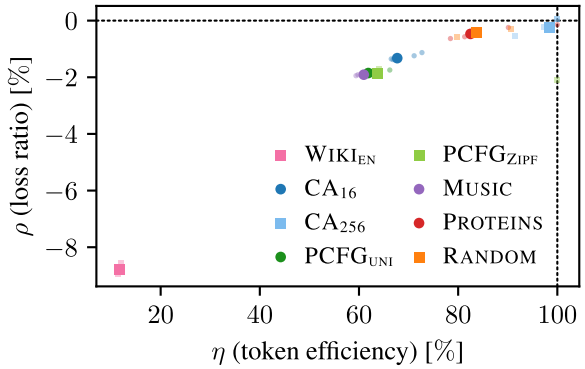


Figure 11: Relation between the two evaluation metrics loss ratio ρ and token efficiency τ.

types, namely, on MUSIC, CA_{16}, and $PCFG_{ZIPF}$. We perform a hyperparameter search with early stopping for the learning rate, weight decay, and warm-up ratio. Next, we select the best configuration and continue parameter search during stage II. Each stage includes tuning for five seeds and ten epochs. The best model (in terms of validation loss), initialized with the seed 2000, is then evaluated on the zero-shot and fine-tuning tasks of the challenge, and results along with the checkpoints are available on HuggingFace. An overview of the searched hyperparameter space is given in the Table 9; all other parameters are kept constant, as listed in Table 7. Model tuning was performed on a single NVIDIA A100 80GB GPU for $\approx$ 2250 GPU hours, where each stage-I run was $\approx$ 4 hours and each stage-II run was $\approx$ 15 hours. The best model along with the results has been uploaded to HuggingFace and BabyLM leaderboard. Results are available in Table 10.

	Dataset	Language	Prediction Task	Reference
Zero-shot	BLiMP	EN	Grammatical acceptability	Warstadt et al. (2020)
	BLiMP-NL	NL	Grammatical acceptability	Suijkerbuijk et al. (2025)
	ZhoBLiMP	ZH	Grammatical acceptability	Liu et al. (2026)
	Hanzi	ZH	Grammatical acceptability	Hu et al. (2026)
	MultiBLiMP	EN, NL	Grammatical acceptability	Jumelet et al. (2026b)
	GlobalPIQA	EN, NL, ZH	Commonsense reasoning	Chang et al. (2026)
	MECO	EN, NL, ZH	Eye-tracking	Siegelman et al. (2022, 2025); Kuperman et al. (2025)
	HellaSwag	EN, NL, ZH	Sentence completion	Zellers et al. (2019)
	Winogrande	EN, NL, ZH	Coreference resolution	Sakaguchi et al. (2021)
	XStoryCloze	EN, NL, ZH	Story completion	Lin et al. (2022b)
	XCOMPS	NL, ZH	Property knowledge	He et al. (2025)
Fine-tuning	ARC	EN, NL, ZH	Abstraction and reasoning	Clark et al. (2018)
	Belebele	EN, NL, ZH	Reading comprehension	Bandarkar et al. (2024)
	BMLama	EN, NL, ZH	Knowledge probing	Qi et al. (2023)
	INCLUDE	NL, ZH	Knowledge probing	Romanou et al. (2025)
	MNLI	EN, NL, ZH	Inference	Williams et al. (2018)
	SIB-200	EN, NL, ZH	Topic classification	Adelani et al. (2024)
	TruthfulQA	EN, NL, ZH	Knowledge probing	Lin et al. (2022a)
	POS	EN, NL, ZH	POS-tagging	Nivre et al. (2020)
	XNLI	EN, ZH	Inference	Conneau et al. (2018)

Table 8: Overview of evaluation datasets used in the BabyLM 2026 multilingual track (Choshen et al., 2026). EN = English, NL = Dutch, ZH = Chinese. All metrics, except MECO, accuracies. Note that MECO is therefore excluded from Fig. 5 and Fig. 6.

Hyperparameter	Value
Optimizer learning rate	1×10^{-4}, 3×10^{-4}, 6×10^{-4}, 1×10^{-2}
Weight decay	0.01, 0.05, 0.1
Warmup ratio	0.01, 0.03
Seeds	11; 17; 42; 2,000; 3,407

Table 9: Hyperparameter search space for the challenge submission.

Benchmark	English	Dutch	Chinese
MultiBLiMP	0.88	0.91	–
BLiMP	0.72	0.78	0.76
WinoGrande	0.52	0.49	0.49
XStoryCloze	0.49	0.48	0.49
HellaSwag	0.27	0.26	0.27
XCOMPS	–	0.53	0.53
Global PIQA (parallel)	0.27	0.23	0.18
Global PIQA (non-para.)	0.52	0.51	0.48
ARC	0.23	0.25	0.24
Belebele	0.30	0.23	0.27
BMLAMA	0.10	0.12	0.10
MNLI	0.58	0.59	0.57
SIB-200	0.33	0.26	0.31
TruthfulQA	0.25	0.26	0.29
XNLI	0.53	–	0.53
INCLUDE	–	0.29	0.26
POS	0.89	0.91	0.81

Table 10: Evaluation results for the submission model trained on the mixed dataset in stage I and on a mix of English, Dutch, and Chinese in stage II.

Arabic BabyLM: Effects of Diglossia and Tokenisation

Mozafar Haider **Burcu Can**
School of Computing, Data, and Mathematical Sciences
University of Stirling
Stirling, Scotland
mozafar@nyala.dev burcu.can@stir.ac.uk

Abstract

This paper explores building a baby language model for Arabic in the context of the BabyLM challenge, with a pretraining corpus limited to 10 million words. We focus on two areas: 1. morphologically-aware segmentation and tokenisation methods suitable for Arabic, and 2. the unique challenges of data curation in diglossic linguistic settings, where the language spoken at home and in everyday communication (Dialectal Arabic) differs substantially from the written language used in formal education and most media (Modern Standard Arabic). We evaluate our models using the `BabyBabelLM` evaluation pipeline. We achieved a significant improvement in the grammaticality scores using different tokenisation and data composition strategies. We discuss these improvements in detail, especially what they reveal about the effect of mixing Modern Standard Arabic and Dialectal Arabic in baby Arabic language models, as well as the challenges they pose for building developmentally plausible models in Arabic.

1 Introduction

The general trend in Large Language Models (LLMs) has been to scale up both model size and dataset size (Kaplan et al., 2020; Hoffmann et al., 2022). However, this leads to computationally inefficient models that are power and data hungry (Bender et al., 2021).

The contrast with how humans learn languages is stark. While humans master languages with exposure to around 100 million words (Gilkerson et al., 2017), a (very) old model like BERT (2018) (Devlin et al., 2019) is trained with 3 billion tokens, and GPT-3 (2020) was trained on 300 billion tokens (Brown et al., 2020). Modern models are trained over trillions of words, OLMo 2 for example, is an open-weight model trained on 6.6 trillion tokens (OLMo Team et al., 2025).

This substantial gap in data efficiency has motivated research on small and baby language models with fewer parameters that can be trained on substantially less data (Lu et al., 2024; Charpentier et al., 2025).

The BabyLM challenge (Warstadt et al., 2023) addresses this gap aiming to develop data-efficient approaches that more closely mirror human language acquisition, particularly that of children. The challenge introduces a multilingual track this year in BabyLM 2026 (BabyLM Team et al., 2026; Jumelet et al., 2025) after following English-only shared tasks between 2023-2025. A developmentally-plausible Arabic dataset and evaluation pipeline were released as part of the multilingual track (`BabyBabelLM`).

In this paper, we explore building a baby language model for Arabic. Extending BabyLM to Arabic introduces additional challenges due to the linguistic structure of the language. First, Arabic is a morphologically rich language with complex concatenative, derivational, templatic and inflectional processes (Habash, 2010). Second, data curation introduces its own challenges because of various reasons: 1. Due to the diglossic nature of Arabic, the spoken and written languages differ substantially (Holes, 2004), creating challenges for child speech data, as children are exposed primarily to spoken Dialectal Arabic (DA) while Modern Standard Arabic (MSA) is used in written language. 2. Available corpora combine text from different dialects which differ lexically, morphologically and syntactically.

To overcome these challenges, we focus on two areas: Tokenisation and dataset curation. We explore several segmentation and tokenisation methods suitable for Arabic morphology and we study their interaction with dataset composition and their performance in different tasks categories (i.e. grammaticality, reasoning, and knowledge). The results show a strong improvement in grammatical learning over the baseline, where morphologically-aware tokenisation paired with certain dataset com-

Proceedings of the Second BabyLM Workshop, pages 227–239
October 29, 2026 ©2026 Association for Computational Linguistics

position strategies, achieves 80% in MultiBLimP (multilingual extension of the grammatical benchmark) (Jumelet et al., 2026).

Our main contributions are:

- A comparative and statistical analysis of various segmentation and tokenisation methods and their effects on improving Arabic language models.
- An analysis of the effects of dataset composition on downstream tasks, especially in terms of mixing Arabic dialects and MSA.
- Five new Arabic datasets along with the best performing models published on HuggingFace.[1]

The implementation of the models as well as the notebooks used for analysis and dataset composition are publicly available[2].

2 Background and Related Work

2.1 The Arabic context

The trend towards increasingly large models, in terms of the parameter count and dataset size, has also extended beyond English to languages such as Arabic where the language modeling landscape is dominated by large headline-grabbing models (Mashaabi et al., 2024). Much of the Arabic LLM landscape is driven by large, government or semi-government-affiliated initiatives: ALLaM, developed by Saudi Arabia's National Center for AI (Bari et al., 2024), trained over 1.2 trillion tokens (of which 45% is Arabic); Fanar, developed at Qatar Computing Research Institute, Hamad Bin Khalifa University (Fanar Team et al., 2025), trained on a corpus of 1 trillion tokens (40% Arabic, 50% English, 10% programming languages); and Falcon-Arabic, developed at the Technology Innovation Institute, Abu Dhabi (Boussaha et al., 2025), trained over 600 billion tokens (40% Arabic).

On the smaller side of models, Kuwain is one of the few models described as a Small Language Model (SLM), although it is trained with 90 billion Arabic tokens (Hennara et al., 2025). AraBERT is trained with 70 million sentences (more than 2.5 billion tokens) (Antoun et al., 2020). To the best of our current knowledge, there is no Arabic model at the scale of a baby language model.

For Arabic, a language shared by more than 20 countries with extremely different socio-economic levels, and where the distribution of wealth does not necessarily equate to the distribution of talent, this is very relevant and ties into the BabyLM's mission of democratising research opportunities in language modeling, as it deprives many researchers from the less economically developed parts of the Arab world from the opportunity to contribute to research.

2.2 Related work

Beyond the baseline submission from BabyBabelLM (Jumelet et al., 2025), relatively little work has explored Arabic language models at this scale. However, several of the techniques investigated in this work have been studied extensively in the context of larger language models.

Tokenisation is an important topic in the Arabic NLP scene (Abdelali et al., 2016; Arnett and Bergen, 2025). This is driven by the morphological structure of Arabic that differs fundamentally from English and requires tokenisation strategies that are better suited to its rich morphology. Therefore, Arabic-aware tokenisation and segmentation is at the core of the Arabic LLMs (Boussaha et al., 2025; Fanar Team et al., 2025; Hennara et al., 2025). In this paper, we explore different techniques for Arabic tokenisation: rule-based Arabic-aware segmentation, stemming, unsupervised learning that learns the morphology automatically from a given corpus along Unigram and Byte Pair Encoding (BPE).

Dataset creation and dataset quality are crucial in SLMs in general, however, the Arabic case has its own specific challenges: the mixture and interactions between dialects and MSA (Habash, 2010; Holes, 2004), culturally-aligned corpora (Alrashed et al., 2024) among other challenges like code-switching and orthographic inconsistencies of written Dialectal Arabic. Dataset creation has also been one of the most effective approaches in previous years' BabyLM challenges. Prévot et al. (2024) uses speech data in French to create a linguistic corpora, Ghanizadeh and Dousti (2024) curate a dataset primarily sourced from child-directed transcripts, Velasco and Roque (2025) curate a dataset with simpler text, using shorter sentences, simpler sentence structures, and simpler vocabulary.

In the context of **BabyLM**, several studies have examined the role of morphology in language modeling for morphologically-rich languages. For ex-

[1]The datasets and models are available at https://huggingface.co/kabaros/datasets

[2]The code is on: https://github.com/kabaros/babylm-ara

Word	Meaning
kataba	to write
ma**ktab**	desk
ma**ktaba**	library
kitab	book
kutub	books
kutayyib	booklet
ma**ktub**	written
katib	writer

Table 1: An example of the non-concatenative root-and-pattern morphology of Arabic: The root **k-t-b** ("write") realised across multiple templatic patterns.

ample, Bölücü and Can (2025) looked at the effect of morphologically-aware tokenisation in English, Kriš and Šuppa (2025) researched building a BabyLM for Slovak which has an inflection-based morphology different from English, and Gelboim and Sulem (2025) explored learning grammatical rules in Hebrew, a Semitic language morphologically very close to Arabic. Fusco et al. (2025) also explored morphologically-aware tokenisation techniques finding that they improve comprehension generalisation at the cost of syntactic accuracy.

3 Arabic Morphology

Morphology plays a central role in Arabic NLP due to its deep interaction with both orthography and syntax (Habash, 2010, Chapter 4).

3.1 Dimensions of morphology

Arabic, similar to other semitic languages like Amharic and Hebrew, is fusional with an inflectional morphology, where consonant roots follow various templates that contribute to grammatical categories and lexical meaning. Contrast to concatenative morphology where affixes are added at the end of each word corresponding to a single grammatical feature (person, tense etc.), Arabic (like other fusional languages) has inflectional morphology and affixes can encode several grammatical categories at once.

Similarly to agglutinative languages such as Turkish and Finnish, Arabic is also considered morphologically rich. A single root can generate dozens of related word forms. This richness increases lexical sparsity and motivates morphologically-aware NLP models. Table 1 illustrates this richness with the triliteral root **k-t-b** ("write"), shown in Latin transliteration.

3.2 Does morphology matter?

Complex inflectional morphology is one of the factors contributing to differences between languages in NLP tasks (Arnett and Bergen, 2025). Morphological richness generates a high number of word forms from a single root and it substantially increases the vocabulary size in NLP applications (Erdmann and Habash, 2018). Arabic, specifically, can have out-of-vocabulary rate up to ten times higher than that of English (Habash, 2010) when using a 64K-word vocabulary.

To keep the vocabulary size optimum, morphologically-aware segmentation and tokenisation methods have been widely explored particularly in such morphologically rich languages, where they help reduce vocabulary size, improve lexical generalization, and mitigate out-of-vocabulary issues.

Tokenisation methods such as BPE and WordPiece generate subword tokens which are usually not aligned with linguistic morphemes, and this has motivated some research to better align subword tokens with linguistic morpheme boundaries (Bauwens and Delobelle, 2024). Morpheme-aware tokenisation has also been explored for Arabic language in a recent study (Asgari et al., 2025), where the results show that morpheme-aware tokenisation improves intrinsic morphological coherence and reduces language model cross-entropy, and not only in Arabic, but also in other morphologically rich languages such as Russian and Hungarian.

4 Dataset Creation for Arabic

4.1 Limitations of the `BabyBabelLM` corpus

`BabyBabelLM` provides an 8.3M dataset for Arabic[3] as part of the first multi-lingual track in BabyLM 2026 (Jumelet et al., 2025). The `BabyBabelLM` itself discusses the limitations of this dataset such as the need to mix up different dialects into one dataset due to the lack of developmentally plausible corpora in Arabic.

We looked into possible modifications to the dataset to make it more developmentally plausible and more coherent in terms of its composition.

The current corpus is over-reliant on two sources: OpenSubtitles and the Habibi corpus (El-Haj, 2020) for Arabic songs. Together, they make almost 80% of the total number of tokens. Subtitles make up

[3] https://huggingface.co/datasets/BabyLM-community/babylm-ara

Data Source	BabyLM Arabic Tokens	BabyLM Arabic %	Custom 8.3M blend† Tokens	Custom 8.3M blend† %	BabyLM Spanish‡ Tokens	BabyLM Spanish‡ %
Habibi (Arabic songs)	3.11M	37.18%	1.39M	16.78%	103.4K	1.1%
OpenSubtitles	3.52M	42.14%	1.40M	16.82%	1.34M	13.9%
The Arabic e-book corpus	1.57M	18.84%	1.57M	18.95%	3.95M	40.7%
vikidia	4.9K	0.06%	4.9K	0.06%	1.44M	14.8%
GlotStoryBook	12.1K	0.14%	12.1K	0.15%	–	–
Ririro	81.9K	0.98%	81.9K	0.99%	–	–
MagicHub Yemeni	36.0K	0.43%	–	–	–	–
MagicHub Egyptian	18.5K	0.22%	18.5K	0.22%	–	–
other-books (e-book genres; not in original)	–	–	697.1K	8.39%	–	–
Hindawi (YA fiction; not in original)	–	–	3.13M	37.64%	–	–
child-directed-speech (Spanish-only; not in babylm-ara)	–	–	–	–	2.87M	29.6%
Total	**8.35M**	**100.00%**	**8.30M**	**100.00%**	**9.71M**	**100.00%**

Table 2: Data source composition of babylm-ara (Arabic dataset from BabyBabelLM) vs. our 8.3M custom blend vs. babylm-spa (Spanish dataset from BabyBabelLM), by token count and percentage of each dataset's own total.

†custom-8.3M is the 8.3M Diverse blend. ‡babylm-spa figures are aggregated liberally to give a general sense of comparison. Habibi equivalent in Spanish are the child-available-speech. The Arabic e-book corpus equivalent are the child-books category (with many more data sources), and child-directed-speech is a major category that does not exist in Arabic.

42% of the corpus. For context, only 13% of the Spanish corpus (a comparable Tier 2 language in the BabyBabelLM datasets) consists of subtitles.

The subtitles dataset also has other issues. To analyse it further, we augmented the dataset with information about the subtitles, using Anthropic Haiku model building a script that infers the movie titles from the subtitles, and extracts the ratings from IMDB API.

From the 559 movies in the original datasets, 347 are duplicated. For example, four versions of *Back to the Future Part III* appear in the dataset with identical translations except for the translator's credits at the beginning. The subtitles also include a large number of movies that are not suitable for children. Table 9 in Appendix A.1 shows the breakdown of the dataset by movie age rating.

To address these issues, we rebuilt the subtitles portion of the dataset from OPUS OpenSubtitles dataset. We retrieved each movie's title and age rating using the IMDB ID used provided in the corpus, and filtered the list to retain only the movies suitable for children. We then ranked them from the most suitable (U Rating) to PG-13, while discarding movies for adults (age rating 15, 18+). We also kept the age rating metadata as part of the augmented dataset to help further experiments in the future (e.g. for setting up age-appropriate curriculum learning).

4.2 Modified datasets

As a result of the previous analysis, we built several 8.3 million datasets with an alternative composition. The datasets relied mostly on augmentation from the Arabic e-book corpus (Hallberg, 2024).

The original dataset already included all the children stories from the Arabic e-book corpus (which is built on freely available books from the Hindawi foundation [4]). We extended that pool with an additional subset of books from additional topics such as history, technology, arts, science, and science fiction, that we reviewed (manually) and identified as suitable for middle or high-school aged children.

We also supplemented the corpus with a popular Arabic Young Adult book series called "The Man of the Impossible"[5]. Although the series is part of the Hindawi project, it has not yet been included in the Arabic e-book corpus.

These additional texts offer several advantages. First, they are written in Modern Standard Arabic (MSA), consistent with the language used in the BabyBabelLM evaluation dataset. Second, their content is more age-appropriate than the movie-based material in the original dataset. We therefore argue that these texts more closely reflect the linguistic input to which a typical Arabic-speaking child might be exposed than the song and subtitle corpora.

While we believe that these new datasets improve upon the original dataset in terms of coherence and suitability for the evaluation tasks, the resulting corpus remains predominantly in MSA, with only a small proportion of DA content. This

[4]https://www.hindawi.org/en

[5]https://en.wikipedia.org/wiki/Ragol_Al_Mostaheel

distribution contrasts sharply with how Arabic is typically acquired: children are exposed predominantly, and sometimes exclusively, to DA until they reach school age (Holes, 2004). We therefore concur with Jumelet et al. (2025) on the importance of developing dialect-centric training corpora, alongside dialect-centric evaluation data, for future Arabic BabyLM research.

Table 3 presents a selection of our modified datasets and their composition. Table 2 further compares the source composition of the original dataset with one of our modified data blends and the Spanish BabyLM dataset, a comparable Tier 2 dataset from `BabyBabelLM`. This comparison provides context for our data composition choices while highlighting the greater diversity of sources represented in the Spanish dataset.

Name	Size	Composition
Original Data *BabyLM-community/babylm-ara, unmodified*	8.35M	Fixed children content 20.7%, subtitles 42.1%, habibi 37.2%.
no-habibi-5M *Original with Habibi and other DA*	5.25M	Fixed children content 32.9%, subtitles 67.1%, habibi 0%. Tests effect of removing DA.
8.3M blend (ebooks-heavy) + variations *wiki_0-movies_x-habibi_x-ebooks_100 blend*	8.30M	Fixed children content 20.2%, ebooks 79.8%, (habibi | movies) (0|5)%. Pure e-book augmentation (variations with 5% movies or Habibi songs).
8.3M Diverse *fixed-20%-movies-17%-habibi-17%-ebooks-8%-hindawi-38% blend*	8.30M	Fixed children content 20.2%, movies 17%, Habibi songs (Egyptian) 17%, ebooks 8%, hindawi 38% (YA books). Most evenly-spread source blend.
8.3M Impossible Man *fixed-20%-ebooks-42%-hindawi-38% blend*	8.30M	Fixed children content 20.2%, ebooks 42.2%, hindawi 37.7%. Augmented with popular Young Adult book series in MSA.

Table 3: 8M-scale modified corpora (size in words). All models contain a fixed 20% from the children books corpus.

5 Methodology

5.1 Experiments

We ran all the experiments on the same base model used in `BabyBabelLM` (Jumelet et al., 2025): GPT-2 light (17M parameters), trained for 10 epochs on Arabic corpora of varying composition (8.3M words similar to the original `BabyBabelLM` dataset). All experiments were conducted on a single `NVIDIA GeForce RTX 4090` GPU. The vocabulary size was fixed at `8192`. Experiments were run with three random seeds, and the reported evaluation scores are the mean of the three runs.

We deliberately kept the model architecture fixed to isolate the effects of Arabic-specific factors, focusing instead on variation in dataset composition, including content and dialectal distribution, as well as linguistic properties of Arabic, such as its morphology and syntax.

We use the multilingual evaluation suite introduced by `BabyBabelLM` (Jumelet et al., 2025) and report our results in the same format to facilitate direct comparison. We additionally provide a more detailed analysis of functional linguistic competence using MultiBLiMP (Jumelet et al., 2026). At the relatively small scale considered here (approximately 10M words), grammaticality judgments provide the most informative signal for Arabic, whereas performance on reasoning and knowledge-based tasks remains near chance level. We therefore focus much of our analysis on grammatical competence and its variation across different data compositions.

5.2 Tokenisation methodology

The lack of morphological alignment in tokenisation negatively impacts the performance of models for morphologically rich languages (Arnett and Bergen, 2025). Given the morphological complexity of Arabic, considerable effort has therefore been devoted to developing tokenisation and segmentation strategies that better capture its morphological structure. Fanar Team et al. (2025) created MorphBPE (Asgari et al., 2025) which incorporates Arabic morphology within BPE, and Koubaa et al. (2024) created AraNizer to address Arabic morphology in their ArabianGPT model. Allam LLM (Bari et al., 2024), Falcon (Boussaha et al., 2025) and Kuwain (Hennara et al., 2025) used tokeniser and vocabulary expansion to address the same issues of morphological misalignment in existing tokenisers. All these studies emphasise the importance of morphology-aware tokenisation in building Arabic LLMs or adapting multi-lingual LLMs to Arabic.

In this paper, we explore several tokenisation methods to compare their effectiveness across different tasks in the BabyLM challenge, particularly for the Arabic language:

- **BPE**: The most widely used tokeniser and the default baseline used by `BabyBabelLM`.

- **Unigram**: The Unigram model has been shown to be effective with Arabic (Kashirskiy et al., 2025) as its probabilistic approach can capture the numerous forms in Arabic morphology.

- **Farasa**: Farasa (Abdelali et al., 2016) is an SVM-based segmenter trained on Arabic-specific

lexicons and features.

- **CAMeL**: CAMeL (Obeid et al., 2020) is a library of tools for Arabic NLP. We used its specialised MSA morphological analyser for segmentation.

`Farasa` and `CAMeL` are segmenters used in pre-tokenisation. We paired them with both `BPE` and `Unigram` in our experiments.

We also explored **ISRI** (Taghva et al., 2005), an Arabic stemmer in the NLTK library, and **Morfessor** which uses unsupervised learning for morphology.

Although `Morfessor` (Virpioja et al., 2013) is theoretically language-independent, its segmentation approach mainly targets languages with a concatenative morphology such as Turkish. In our experiments, the results were inconsistent, and the resulting tokenisation was largely misaligned with the morphological structure of Arabic.

Our motivation for incorporating `ISRI` was that, although its relatively simple stemming procedure would not capture the morphological distinctions required for Arabic grammaticality judgments, its ability to identify roots might nevertheless benefit reasoning tasks. This intuition is consistent with the stemmer's reported effectiveness in information retrieval (Taghva et al., 2005). The approach is also interesting from the perspective of developmental plausibility: we hypothesise that an early emphasis on lexical roots may loosely resemble the vocabulary-building stage of early Arabic language acquisition. Moreover, root-based representations may be more robust to dialectal variation, as many dialectal forms can be traced to roots shared with MSA (Darwish et al., 2014). Our results provide some support for the reasoning hypothesis, albeit with an important caveat: `ISRI` outperformed the more sophisticated segmenters on zero-shot reasoning tasks, but, as in the other experiments, its absolute performance remained only marginally above chance. We therefore omitted both `ISRI` and `Morfessor` from the main results and discussion, as neither yielded additional insights beyond these largely negative findings.

6 Results

Zero-shot grammaticality scores on MultiBLiMP are reported in Table 4. Grammaticality showed the largest improvements across our experiments, making these results particularly informative for identifying potential optimisation strategies for small-scale Arabic language models.

We also report results on the reasoning and knowledge tasks. However, consistent with the baselines reported by Jumelet et al. (2025), performance showed no substantial improvement and remained close to chance level, even after fine-tuning.

6.1 Arabic BabyLMs can learn grammar

Across all experiments, grammaticality results improved with the dataset and tokeniser choices. The baseline score in Jumelet et al. (2025) is `75.9` (our multi-seed reproduction of the same setup is lower at `74.73`) was improved significantly (up to 5%) by changing the corpus composition and the tokeniser.

Across our five datasets, the morphologically-aware tokenisers `CAMeL` and `Farasa` achieved the best scores overall, when paired with `Unigram`. The e-books heavy dataset was the best performing dataset. The best score achieved on these datasets was `79.23`. For context, OLMo-2, a state-of-the-art 32 billion parameters model, trained over 6 trillion tokens, achieves 87% on MultiBLiMP in Arabic (Jumelet et al., 2026).

6.1.1 Tokenisation and dataset composition both affect grammatical learning

Both tokenisation and dataset composition have a significant effect on MultiBLiMP performance. The two-way ANOVA decomposition reported in Table 5 indicates that dataset composition is the stronger factor, accounting for 42.1% of the total variance, compared with 31.9% explained by tokeniser choice. Both effects are statistically significant, showing that improvements in small-scale Arabic language modelling depend on both the composition of the training data and the choice of tokenisation strategy.

Source	Sum Sq	df	F	p	% variance
Dataset	109.91	5	31.41	<0.001	42.1
Tokeniser	83.40	5	23.83	<0.001	31.9
Residual	67.89	97	–	–	26.0

Table 5: Two-way ANOVA decomposition of `multiblimp_arb` scores on the 6-dataset × 6-tokeniser grid from Table 4 (fit on per-seed scores, not dataset/tokeniser cell means). Both dataset and tokeniser choices are statistically significant.

Dataset	BPE	CAMeL (BPE)	CAMeL (Unigram)	Farasa (BPE)	Farasa (Unigram)	Unigram	Average by dataset
Original Data	74.73	75.33	**76.43**	74.57	75.33	75.14	75.25
8.3M (ebooks-heavy)	76.57	**79.23**	79.12	78.33	77.70	76.68	77.94
8.3M (ebooks w/ 5% habibi)	76.00	77.48	**78.85**	77.34	77.97	77.01	77.44
8.3M (ebooks w/ 5% movies)	75.28	77.04	**78.96**	77.47	77.39	75.42	76.93
8.3M (Impossible Man)	74.98	78.05	78.19	78.02	**78.90**	77.78	77.65
8.3M (Diverse)	74.29	75.94	**77.15**	74.54	76.49	75.61	75.67
Average by tokeniser	75.31	77.18	78.12	76.71	77.30	76.27	

Table 4: `multiblimp` scores (mean of 3 seeds) by dataset and tokeniser, with the best tokeniser per dataset in bold. `Original/BPE` (`74.73`) is our multi-seed reproduction of (Jumelet et al., 2025). The paper's reported score is `75.9`

6.1.2 Mixing Arabic varieties dilutes learning

Across all tokenisers, the highest MultiBLiMP grammaticality scores were obtained with datasets containing a high proportion of MSA. In particular, the e-books-heavy dataset achieved a mean score of `79.23%` across three random seeds, with the best individual run reaching `80.3%`.[6]

While non-Arabic speakers might find it reasonable, even expected, that a DA-heavy dataset would not perform well in an MSA-heavy benchmark, the notion that "it is all one Arabic" (Holes, 2004) is deeply entrenched for native Arabic speakers that we think this is a finding worth calling out explicitly.

Mixing dialects and MSA is common in larger models (Fanar Team et al., 2025; Hennara et al., 2025; Boussaha et al., 2025), but our results show that this is not beneficial for small models of BabyLM scale. Table 6 shows the effect of removing all DA content (i.e. Habibi songs, and MagicHub children conversations in Yemeni and Eygptian). Despite losing 3 million tokens, the grammaticality scores remained comparable, even improving with some tokenisers. The improvement, tellingly, happens when using vanilla `BPE` or `Unigram` rather than morphologically-aware tokenisation, further highlighting the linguistic differences between MSA and DA, where the grammar, syntax and phonology can differ substantially (Habash, 2010, ch. 1, p. 2).

6.1.3 Morphologically-aware segmenters pair better with Unigram than BPE

Pairing morphologically aware segmenters, such as `Farasa` and `CAMeL`, with `Unigram` tokenisation consistently yielded higher scores than pairing them with `BPE`.

	BPE	CAMeL	Farasa	Unigram
Original (8M)	74.73	75.33	74.57	75.14
No DA (5.1M)	75.12	74.78	74.21	76.00
delta	+0.39	-0.56	-0.36	+0.85

Table 6: The change in MultiBLiMP scores after removing DA content (without re-augmentation)

Although `Unigram` proved effective as a standalone tokeniser without segmentation or preprocessing, we hypothesized that it would not pair well with a segmenter as segmentation would remove much of its advantage in inferring morphological boundaries. However, our experiments showed Unigram consistently outperforms BPE as a tokeniser after segmentation. Moreover, `HuggingFace` implementation of Unigram outperformed `Google SentencePiece` implementation (both as a standalone tokeniser or when paired with the segmenters).

`CAMeL + Unigram` performs the best for 4 out of 6 datasets, while `Farasa + Unigram` outperforms the others in one of them. `CAMeL` improves 0.88% on average, while `Farasa` improves 0.31% by switching from BPE to Unigram. A morphologically-aware segmentation paired with `Unigram` beats `vanilla Unigram` in all the experiments.

6.2 The tokenisers in isolation

To further investigate the effects of mixing dialects in Arabic, we analyse the tokenisers in isolation independently of their downstream language models as part of an intrinsic evaluation framework. To this end, for each tokenisation method, we used a tokeniser trained on a pure MSA dataset (the 8.3M

[6]The model corresponding to this run is publicly available at `https://huggingface.co/kabaros/ar-camel-ebooks-heavy-8.3M`.

	Fertility			4+ token words %			1-char tokens %			Vocab util.		
Tokeniser	MSA	DA	Δ	MSA	DA	Δ	MSA	DA	Δ	MSA	DA	Δ
BPE	2.39	1.99	−0.40	16.56	5.48	−11.08	41.75	19.55	−22.21	0.97	0.93	−0.04
CAMeL	2.20	2.32	+0.12	14.93	15.65	+0.72	25.97	17.10	−8.87	0.94	0.86	−0.08
CAMeL+Unigram	2.54	2.71	+0.17	22.52	25.98	+3.46	44.56	38.26	−6.30	0.96	0.89	−0.07
Farasa	1.93	2.14	+0.21	10.24	9.54	−0.69	8.19	8.86	+0.66	0.88	0.86	−0.02
Farasa+Unigram	2.63	2.80	+0.17	23.11	29.62	+6.51	51.45	49.85	−1.60	0.96	0.90	−0.06
Unigram	3.10	2.42	−0.67	23.74	16.80	−6.93	61.31	46.09	−15.22	0.97	0.89	−0.08

Table 7: Tokenisation metrics on MSA vs. DA content, using a tokeniser trained on the `8.3M Impossible Man` MSA dataset. Δ is DA − MSA.

Impossible Man dataset) to tokenise the original BabyLM corpus (with its mix of MSA and DA). We then calculated tokenisation metrics following the evaluation framework that Gazit et al. (2025) applied to Hebrew, a language that shares Arabic templatic morphology. The metrics are calculated on the MSA part of the dataset, and the DA part separately. The metrics reported are: 1. fertility (the average number of tokens per word), 2. the percentage of words tokenised into four or more tokens or single-character tokens and 3. vocabulary utilisation indicating how much of the tokeniser's vocabulary is being used.

The evaluation results are presented in Table 7. However, these scores alone are not reliable indicators of downstream performance in Arabic. For example, although lower fertility is generally considered desirable, morphologically rich languages may benefit from further segmenting words into meaningful subword units, resulting in higher fertility while better preserving morphological information (Asgari et al., 2025).

What the numbers show are clear patterns within tokenisation categories lines. These patterns, when combined with downstream performance, can give an indication of the intrinsic tokenisation metrics that lead to better results in the case of Arabic.

For fertility, morphologically aware tokenisers show an increase in scores from the MSA to the DA portions of the dataset, whereas vanilla `BPE` and `Unigram` show a decrease.

For 4+ token words and 1-char tokens, the percentages decrease significantly between MSA and DA for vanilla `BPE` and `Unigram`, while the morphologically-aware segmenters remain relatively stable, possibly indicating better resilience to corpora with mixed dialects that explains their better performance on mixed datasets corpora.

`Unigram`, whether used standalone or paired with a segmenter, generates a higher percentage of words split into four or more tokens. While a lower percentage generally indicates better compression, which has been associated with improved downstream performance (Gazit et al., 2025), models using `Unigram` performed better on Arabic. One possible explanation is that its finer-grained tokenisation may better capture meaningful subword units in longer or morphologically complex words. This is in line with the findings of Schmidt et al. (2024), who show that better compression alone does not necessarily translate into improved downstream performance.

Vocabulary utilisation is high across tokenisers, with the exception of `Farasa`, which improves significantly when paired with `Unigram`. This provides a possible explanation for the findings reported in Section 6.1.3 where we reported that pairing with `Unigram` always outperforms pairing with `BPE`.

These scores provide additional context for the findings in Section 6.1.3, helping to explain why pairing morphologically aware tokenisation with *Unigram* yields the best overall results.

6.3 Reasoning and Knowledge

Consistent with the findings of Jumelet et al. (2025), the fine-tuning results are generally modest and exhibit no clear pattern. Unlike grammaticality, performance on the knowledge and reasoning tasks does not appear to respond systematically to changes in dataset composition or augmentation. This is consistent with the findings reported by Olsson et al. (2022) that such learning only starts in a much higher scale rather than in these tiny models.

For completeness, we provide fine-tuning results along with zero-shot results in Table 10 in Appendix A.2.

7 Conclusion and Future Work

In this work, we explore building a baby language model in Arabic. We focus on the interactions between tokenisation and dataset composition in the Arabic context, especially the interactions between Dialectal Arabic (DA) and Modern Standard Arabic (MSA).

The results show that models of this size can improve their grammatical learning despite being trained on a limited number of tokens. This finding contrasts with similar experiments on morphologically aware tokenisation in English (Fusco et al., 2025), where improvements were observed in comprehension but not in grammaticality.

We also explore the effects of mixing DA and MSA in the training corpus. We find that morphologically-aware tokenisers, paired with Unigram, perform the best. Our results confirm previous findings (and intuition) about the importance of tokenisation in morphologically-rich languages. However, consistent with previous work (Fusco et al., 2025), our results show that there is no single best approach. Rather, the interaction between dataset creation, data augmentation, and tokenisation plays a crucial role in determining the performance of language models on a small scale.

Future work could explore the trajectory of learning, both in terms of different training regimes or going beyond the 10M tokens. Other tokenisation strategies should be explored such as dedicated normalisation pipelines for Arabic orthographic peculiarities (especially for DA), as well as other techniques like splintering (Gazit et al., 2025) that are promising for Arabic. For the interaction between DA and MSA, ideas from curriculum learning where MSA can be introduced later than DA, or using age rating for an age-appropriate curriculum learning could be interesting to explore.

Cross-lingual transfer is also another potential research direction between Arabic, Amharic and Hebrew, which are typologically similar Semitic languages. However, its success likewise depends on the availability of suitable datasets in these languages.

8 Limitations

The main limitation in this study is the limited availability of child-appropriate datasets in Arabic, especially in DA, as well as the absence of suitable evaluation data for DA. The datasets we created are still far from being developmentally plausible, even compared to other Tier 2 languages in `BabyBabelLM`.

For this initial exploration, we focused on providing a general overview of two topics we thought are the most important in the Arabic context (tokenisation and dataset creation). Either of these topics, in Arabic, deserves its own research and a deeper investigation.

Some of these limitations are inherited from Jumelet et al. (2025). However, like Jumelet et al., we view this work as a starting point and hope that it will open the door to further research on Arabic and provide a foundation for future work to build upon.

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations (NAACL-HLT)*, pages 11–16, San Diego, California. Association for Computational Linguistics.

Sultan Alrashed, Dmitrii Khizbullin, and David R. Pugh. 2024. FineWeb-Edu-Ar: Machine-translated corpus to support Arabic small language models. *arXiv preprint arXiv:2411.06402*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resources Association.

Catherine Arnett and Benjamin K. Bergen. 2025. Why do language models perform worse for morphologically complex languages? In *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*, pages 6607–6623, Abu Dhabi.

Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. 2025. MorphBPE: A morpho-aware tokenizer bridging linguistic complexity for efficient LLM training across morphologies. *arXiv preprint arXiv:2502.00894*.

BabyLM Team, Leshem Choshen, Ryan Cotterell, Jaap Jumelet, Tal Linzen, and Aaron Mueller. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

M. Saiful Bari et al. 2024. ALLaM: Large language models for Arabic and English. *arXiv preprint arXiv:2407.15390*.

Thomas Bauwens and Pieter Delobelle. 2024. BPE-knockout: Pruning pre-existing BPE tokenisers with backwards-compatible morphological semi-supervision. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5810–5832, Mexico City, Mexico. Association for Computational Linguistics.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT '21)*, pages 610–623, Virtual Event, Canada. ACM.

Necva Bölücü and Burcu Can. 2025. A morpheme-aware child-inspired language model. In *Proceedings of the First BabyLM Workshop*, pages 279–287, Suzhou, China. Association for Computational Linguistics.

Basma El Amel Boussaha, Mohammed Alyafeai, Ahmed Alzubaidi, Leen AlQadi, Shaikha Alsuwaidi, and Hakim Hacid. 2025. Adapting falcon3-7b language model for arabic: Methods, challenges, and outcomes. In *Proceedings of The Third Arabic Natural Language Processing Conference*, pages 1–15. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third babylm challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–417, Suzhou, China. Association for Computational Linguistics.

Kareem Darwish, Hassan Sajjad, and Hamdy Mubarak. 2014. Verifiably effective Arabic dialect identification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1465–1468, Doha, Qatar. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mahmoud El-Haj. 2020. Habibi – a multi dialect multi national arabic song lyrics corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, pages 1318–1326.

Alexander Erdmann and Nizar Habash. 2018. Complementary strategies for low resourced morphological modeling. In *Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 54–65, Brussels, Belgium. Association for Computational Linguistics.

Fanar Team, Ummar Abbas, Mohammad Shahmeer Ahmad, Firoj Alam, Enes Altinisik, Ehsaneddin Asgari, Yazan Boshmaf, Sabri Boughorbel, Sanjay Chawla, Shammur Chowdhury, Fahim Dalvi, Kareem Darwish, Nadir Durrani, Mohamed Elfeky, Ahmed Elmagarmid, Mohamed Eltabakh, Masoomali Fatehkia, Anastasios Fragkopoulos, Maram Hasanain, and 23 others. 2025. Fanar: An Arabic-centric multimodal generative AI platform. *arXiv preprint arXiv:2501.13944*.

Achille Fusco, Maria Letizia Piccini Bianchessi, Tommaso Sgrizzi, Asya Zanollo, and Cristiano Chesi. 2025. Linguistic units as tokens: Intrinsic and extrinsic evaluation with BabyLM. In *Proceedings of the 3rd BabyLM Challenge at the 2025 Conference on Empirical Methods in Natural Language Processing*.

Bar Gazit, Shaltiel Shmidman, Avi Shmidman, and Yuval Pinter. 2025. Splintering nonconcatenative languages for better tokenization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 22405–22417, Vienna, Austria. Association for Computational Linguistics.

Anita Gelboim and Elior Sulem. 2025. TafBERTa: Learning grammatical rules from small-scale language acquisition data in hebrew. In *Proceedings of the 3rd BabyLM Challenge at the 2025 Conference on Empirical Methods in Natural Language Processing*.

Mohammad Amin Ghanizadeh and Mohammad Javad Dousti. 2024. Towards data-efficient language models: A child-inspired approach to language learning. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 22–27, Miami, FL, USA. Association for Computational Linguistics.

Jill Gilkerson, Jeffrey A. Richards, Steven F. Warren, Judith K. Montgomery, Charles R. Greenwood, D. Kimbrough Oller, John HL Hansen, and Terrance D. Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American Journal of Speech-Language Pathology*, 26(2):248–265.

Nizar Y. Habash. 2010. *Introduction to Arabic Natural Language Processing*. Number 10 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

Andreas Hallberg. 2024. The Arabic E-Book Corpus (version 1) [data set]. https://researchdata.se/en/catalogue/dataset/2024-145/1. 81M-word corpus of 1,745 books published by the Hindawi Foundation.

Khalil Hennara, Sara Chrouf, Mohamed Motasim Hamed, Zeina Aldallal, Mohammad Omar Hadid, and Safwan AlModhayan. 2025. Kuwain 1.5B: An arabic SLM via language injection. *arXiv preprint arXiv:2504.15120*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*. ArXiv:2203.15556.

Clive Holes. 2004. *Modern Arabic: Structures, Functions, and Varieties*, revised edition. Georgetown University Press.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2025. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. *arXiv preprint arXiv:2510.10159*.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Mark Kashirskiy, Artiom Lipinski, and Ilya Makarov. 2025. AraToken: Optimizing Arabic tokenization with normalization pipeline and language extension for Qwen3. *arXiv preprint arXiv:2512.18399*.

Anis Koubaa, Adel Ammar, Lahouari Ghouti, Omar Najar, and Serry Sibaee. 2024. ArabianGPT: Native Arabic GPT-based large language model. *arXiv preprint arXiv:2402.15313*.

L'uboš Kriš and Marek Šuppa. 2025. SlovakBabyLM: Replication of the babylm and sample-efficient pretraining for a low-resource language. In *Proceedings of the 3rd BabyLM Challenge at the 2025 Conference on Empirical Methods in Natural Language Processing*.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D. Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Malak O. Mashaabi, Shahad Z. Al-Khalifa, and Hend S. Al-Khalifa. 2024. A survey of large language models for Arabic language and its dialects. *arXiv preprint arXiv:2410.20238*.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, pages 7022–7032.

OLMo Team, Pete Walsh, Luca Soldaini, Dirk Groeneveld, et al. 2025. 2 OLMo 2 furious. *arXiv preprint arXiv:2501.00656*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Laurent Prévot, Sheng-Fu Wang, Jou-An Chi, and Shu-Kai Hsieh. 2024. Extending the BabyLM initiative : Promoting diversity in datasets and metrics through high-quality linguistic corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 147–158, Miami, FL, USA. Association for Computational Linguistics.

Craig W. Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. 2024. Tokenization is more than compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 678–702, Miami, Florida, USA. Association for Computational Linguistics. Accessed 9 September 2026.

Kazem Taghva, Rania Elkhoury, and Jeffrey Coombs. 2005. Arabic stemming without a root dictionary. In *International Conference on Information Technology: Coding and Computing (ITCC'05)*, pages 152–157, Las Vegas, NV, USA.

Dan John Velasco and Matthew Theodore Roque. 2025. Rethinking the role of text complexity in language model pretraining. In *Proceedings of the First BabyLM Workshop*, pages 1–28, Suzhou, China. Association for Computational Linguistics.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Technical Report 25/2013, Aalto University.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL 2023), Volume 2: The BabyLM Challenge*, pages 1–34. Association for Computational Linguistics.

A Appendix

A.1 Notes about the original BabyLM corpus

The original Arabic `BabyBabelLM` dataset relies heavily on OpenSubtitles, many movies are duplicated, shown in Table 8 and many are unsuitable for children. Table 9 shows the movie age-rating of the original corpus.

Duplicates	Sample movies
4	*Frankenstein*, *Spider-Man*, *The Matrix* and 4 more
3	*Shrek*, *Evil Dead* and 14 more
2	*The Polar Express*, *Rambo*, *Back to the Future* and 32 more

Table 8: Sample of duplicated movies in the original dataset (58 movies out of 211 appear more than once)

Rating	Classification	Movies	%
PG / PG-13	Child-friendly	97	45.8
G	Child-friendly	13	6.1
U	Child-friendly	9	4.2
12	Child-friendly	7	3.3
R	Adult	32	15.1
15	Adult	8	3.8
18	Adult	2	0.9
NC-17	Adult	1	0.5
(null)	Unknown	43	20.3
Total		**212**	**100.0**

Table 9: Rating composition of the OpenSubtitles movies in the original BabyLM dataset.

A.2 Complete results

Table 10 presents the complete set of results for all tasks and setups.

		Zero-shot					Knowledge (fine-tuned)			Reasoning (fine-tuned)					
	Tokeniser	multiblimp	xcomps	hellaswag	winogrande	xstorycloze	bmlama	include	mmlu	arc	belebele	mmli	truthfulqa	xnli	sib200
	Random	50.0	50.0	25.0	50.0	50.0	10.0	25.0	25.0	25.0	25.0	33.0	25.0	33.0	14.3
	Paper	75.88	52.92	25.76	47.40	46.75	15.3	36.8	28.5	27.8	25.6	44.3	28.0	37.1	45.8
Original Data	BPE	74.73	51.93	25.58	50.12	47.94	13.85	36.75	28.70	27.59	27.78	46.09	**31.36**	41.95	47.76
	CAMeL	75.33	52.12	25.40	49.11	47.37	13.19	41.03	27.55	**29.46**	25.56	44.74	26.27	40.90	40.30
	CAMeL+Uni	76.43	52.08	25.78	49.27	47.34	14.18	41.03	**29.40**	28.63	25.56	47.10	27.12	41.40	56.22
	Farasa	74.57	51.82	25.86	50.12	48.07	13.19	35.04	27.15	26.97	28.89	45.47	29.66	42.05	52.24
	Farasa+Uni	75.33	52.89	25.77	49.60	48.40	14.34	35.90	**29.15**	26.56	29.44	47.72	**31.36**	**45.20**	54.23
	Unigram	75.14	52.06	25.74	49.68	47.19	15.42	40.17	27.35	26.14	**33.89**	48.17	26.27	43.65	55.22
8.3M (ebooks)	BPE	76.57	52.57	25.20	51.14	48.56	14.43	38.46	27.05	26.76	26.67	46.54	27.12	43.20	55.22
	CAMeL	**79.23**	52.32	25.31	50.35	47.97	14.51	35.90	27.25	28.22	26.11	45.81	30.51	42.40	52.24
	CAMeL+Uni	**79.12**	51.74	25.48	50.81	48.81	14.92	34.19	27.55	**29.46**	26.11	**49.35**	27.97	**44.45**	57.21
	Farasa	78.33	52.41	25.57	51.22	49.33	**15.75**	**43.59**	27.05	26.14	22.78	45.64	26.27	**44.50**	55.22
	Farasa+Uni	77.70	**53.29**	25.53	50.32	49.20	14.51	36.75	27.50	27.80	26.11	48.40	27.97	44.25	**61.19**
	Unigram	76.68	52.54	25.43	50.78	48.30	15.17	32.48	27.65	24.90	26.11	46.76	26.27	43.65	55.22
8.3M (ebooks w/ 5% habibi)	BPE	76.00	52.51	25.25	51.22	48.99	15.75	36.75	28.45	25.93	27.78	46.37	27.97	43.55	52.74
	CAMeL	77.48	52.34	25.13	49.93	48.04	15.50	37.61	27.65	26.97	23.33	46.43	29.66	41.95	49.75
	CAMeL+Uni	78.85	52.18	25.38	51.00	48.99	14.84	31.62	27.00	**30.08**	22.22	46.15	27.97	42.40	58.71
	Farasa	77.34	52.43	25.52	51.19	49.25	15.17	36.75	27.50	27.18	25.00	46.20	29.66	41.55	48.76
	Farasa+Uni	77.97	52.78	25.53	**51.47**	**49.38**	14.51	36.75	27.70	26.76	21.67	47.72	27.12	43.20	57.71
	Unigram	77.01	52.43	25.32	50.29	48.12	15.42	40.17	27.75	26.76	27.22	46.37	25.42	43.05	57.21
8.3M (ebooks w/ 5% movies)	BPE	75.28	52.59	25.27	**51.47**	48.17	15.50	41.03	27.00	27.80	28.33	46.60	30.51	41.50	55.72
	CAMeL	77.04	52.26	25.26	50.48	47.68	15.33	36.75	26.55	25.31	**30.56**	45.64	30.51	42.50	51.74
	CAMeL+Uni	**78.96**	52.47	25.38	51.00	48.14	15.33	32.48	26.80	25.52	25.00	47.21	**35.59**	44.10	51.74
	Farasa	77.47	52.09	25.49	**51.77**	**50.13**	14.43	38.46	26.70	25.93	27.22	46.88	27.97	42.50	56.72
	Farasa+Uni	77.39	**53.06**	25.54	51.14	49.10	14.26	36.75	28.40	28.01	26.11	48.00	31.36	43.50	56.22
	Unigram	75.42	52.21	25.52	50.21	48.12	14.34	35.90	**29.40**	25.93	25.56	47.27	24.58	43.05	58.71
8.3M (Impossible Man)	BPE	74.98	52.60	25.66	50.45	48.38	**16.49**	33.33	28.60	27.39	26.67	45.13	26.27	43.75	53.73
	CAMeL	78.05	51.73	25.60	50.84	47.39	**16.82**	34.19	28.85	24.27	26.67	47.55	27.12	42.70	56.72
	CAMeL+Uni	78.19	52.58	25.76	49.22	49.10	15.09	34.19	27.70	28.63	27.78	47.38	28.81	43.40	58.71
	Farasa	78.02	52.46	25.78	50.54	**49.46**	15.58	**45.30**	27.80	25.93	30.56	47.55	27.12	43.05	59.20
	Farasa+Uni	78.90	52.22	25.64	49.57	48.99	14.51	35.90	28.30	26.14	23.89	47.55	27.12	43.45	**61.69**
	Unigram	77.78	52.04	**25.96**	49.82	48.27	14.84	40.17	27.60	27.59	24.44	**48.96**	27.97	44.05	53.23
8.3M (Diverse)	BPE	74.29	51.97	25.73	50.51	48.53	15.25	38.46	27.95	26.14	24.44	46.43	27.97	42.45	48.26
	CAMeL	75.94	52.31	25.68	50.64	47.58	14.51	37.61	28.80	26.97	**31.11**	46.93	25.42	44.30	44.78
	CAMeL+Uni	77.15	52.50	25.63	49.57	48.22	15.09	35.90	27.80	26.97	29.44	47.55	28.81	41.70	60.20
	Farasa	74.54	52.77	25.73	51.34	48.45	13.77	38.46	26.75	25.73	21.11	46.31	27.12	42.95	50.25
	Farasa+Uni	76.49	**53.45**	**25.87**	50.23	48.20	15.66	34.19	28.90	28.01	23.89	**48.96**	28.81	44.30	**63.68**
	Unigram	75.61	52.01	**25.88**	50.10	47.86	14.43	**41.88**	27.80	27.59	26.11	48.11	27.97	43.25	54.23

Table 10: Full results for zero-shot and fine-tuned tasks across multiple datasets and all tokenisers (zero-shot scores are the mean of three seeds).

Repetition Without Exclusivity: Scale Sensitivity of Referential Mechanisms in Child-Scale Language Models

Jon-Paul Cacioli
Independent Researcher
Melbourne, Australia
synthium@hotmail.com

Abstract

We present the first systematic evaluation of mutual exclusivity (ME), the bias to map novel words to novel referents, in text-only language models trained on child-directed speech. We operationalise ME as referential suppression and train 45 GPT-2 models (2.9M–33.5M parameters; 5–20 epochs on AO-CHILDES; 5 seeds) on a pre-registered ME battery. Across all 9 cells of the grid, models show robust anti-ME repetition priming, the opposite of ME, in 85–100% of items (all $p < 2.5 \times 10^{-13}$), attenuating with improved language modelling (Spearman $\rho = -0.533$) but never crossing zero. A positive control on pretrained GPT-2 (124M–1.5B) shows the same priming, attenuating with scale but never reversing into significant ME, confirming the battery can detect suppression when present. The dissociation also holds on a size-matched non-CDS corpus (WikiText). A context-dependence diagnostic further shows that apparent ME-like patterns with nonce tokens reflect embedding similarity, not referential disambiguation. We connect this to the grounded cognition literature, where ME does emerge in visually grounded models, and argue that referential grounding, or another mechanism instantiating competition among candidate referents, may be required for ME-like behaviour to emerge.

1 Introduction

The BabyLM paradigm trains language models on child-scale data and evaluates linguistic competence (Warstadt et al., 2023; Charpentier et al., 2025). Evaluations focus on syntax (BLiMP; Warstadt et al., 2020) and natural language understanding (GLUE; Wang et al., 2018). But referential pragmatics, how children use words to pick out objects in the world, remains untested. We address this gap using mutual exclusivity (ME) as the test case.

ME is the bias to map novel words to novel referents rather than objects that already have a name (Markman and Wachtel, 1988). Intuitively, if a child sees a familiar dog and an unfamiliar object and hears "look at the dax!", ME leads the child to assume "dax" refers to the unfamiliar object, because the dog already has a name. The bias is central to early word learning, developmentally graded and modulated by vocabulary size (Lewis et al., 2020), weaker in bilinguals (Byers-Heinlein and Werker, 2009), and has been studied computationally in grounded models (Gulordava et al., 2020; Ohmer et al., 2022; Nortje et al., 2024; Oneaţă et al., 2025; Thai et al., 2025). A key feature of all computational work showing positive ME results is that the model has referents (images, object representations, or symbolic referent sets) to which labels can be mapped.

BabyLM-class models are text-only. Can ME emerge from distributional learning on child-directed speech alone, without referential grounding? This is not a question about whether text-only models *should* exhibit ME; it is a question about whether distributional statistics are *sufficient* to induce it.

The question is sharpened by recognising that ME and a competing pressure act on the same measurement. When a noun is used as a label in a short discourse, two forces bear on its probability at a later completion position. Repetition priming (or givenness) raises it: recently mentioned words become more likely. Mutual exclusivity lowers it: once an object is named, a novel label should attach elsewhere, suppressing the just-used noun as a candidate for a new referent. These pressures point in opposite directions, so the sign of the net effect is diagnostic. Child-directed speech is dense with repetition (§3.2), which should teach priming; whether text-only learning *also* extracts the referential-contrast signal underlying ME is precisely what is at issue. We therefore measure whether labelling a noun raises its later probability (priming, anti-ME) or lowers it (suppression,

Proceedings of the Second BabyLM Workshop, pages 240–249
October 29, 2026 ©2026 Association for Computational Linguistics

ME-consistent).

We test this across 45 autoregressive models spanning a 12× parameter range and 3.8× perplexity range, plus a masked language model (BabyBERTa; Huebner et al., 2021), and validate the measure with a positive control on pretrained GPT-2 up to 1.5B parameters. We find no evidence of ME at any scale tested; instead, models develop repetition priming that attenuates with improved language modelling but never reverses sign. We provide a mechanistic account grounded in the statistics of child-directed speech and introduce a context-dependence diagnostic that rules out artefactual ME from embedding similarity, a reusable methodological contribution for prompt-based pragmatic evaluation.

2 Background

2.1 ME in Developmental Psychology

ME was first characterised by Markman and Wachtel (1988) as a constraint on word learning: children assume that each object has only one label, so a novel word is mapped to the unnamed object rather than one that already has a name. The bias emerges around 17 months (Halberda, 2003), strengthens with vocabulary growth (Lewis et al., 2020), and is modulated by language experience: bilingual children show weaker ME, consistent with their routine experience of multiple labels for the same referent (Byers-Heinlein and Werker, 2009). An important distinction drawn by McMurray et al. (2012) separates referent selection (an in-the-moment disambiguation) from word learning (retention over time); our evaluation targets the former.

Bayesian accounts formalise ME as probabilistic inference over word–referent mappings (Xu and Tenenbaum, 2007), where the critical ingredient is the referent set: the learner must represent distinct objects, each requiring a label.

2.2 ME in Computational Models

Cross-situational neural models exhibit ME when lexical competition is built into learning or selection (Gulordava et al., 2020); pragmatic agents combining RSA reasoning with gradient learning also develop it (Ohmer et al., 2022), as do meta-learning systems for few-shot word learning (Gandhi and Lake, 2020).

The most relevant recent work comes from visually grounded speech (VGS) models. Nortje et al. (2024) showed that ME emerges from contrastive learning on speech–image pairs, with stronger ME when models have more visual prior knowledge. Extending this to the bilingual case, Oneaţă et al. (2025) found that bilingual VGS models exhibit weaker ME, paralleling human developmental data. Thai et al. (2025) introduced MEBench and reported that even state-of-the-art vision-language models show only weak ME bias.

The key insight from this literature is that every positive ME result involves models with referents. ME has never been evaluated in text-only models trained on naturalistic child-directed speech.

2.3 Child-Scale Language Models

The BabyLM Challenge (Warstadt et al., 2023; Charpentier et al., 2025) trains models on 10M–100M-word corpora, approximating children's input, and evaluates them on syntax (BLiMP) and NLU (GLUE). BabyBERTa (Huebner et al., 2021) trains RoBERTa-base on AO-CHILDES, American-English CDS (∼4M tokens). These evaluations target syntax and NLU; referential pragmatics is untested. We fill this gap.

3 Method

3.1 Pre-Registration and Transparency

We conducted a pilot study on a single autoregressive model and BabyBERTa (described in §3.5) that yielded three mechanistic findings. Based on these exploratory results, we pre-registered a confirmatory scale-sensitivity experiment on the Open Science Framework (OSF; `https://osf.io/zu7af`) before training any additional models. The pre-registration specifies four hypotheses (§3.6), statistical tests, significance thresholds, and the exact training grid. Train and evaluation scripts are publicly available at `https://github.com/synthiumjp/study2-me-eval`. Pilot findings are reported as exploratory with full transparency about the analysis path.

3.2 Corpus

AO-CHILDES (Huebner et al., 2021) contains 893,989 sentences (∼4.07M word tokens, 6.27M BPE tokens) of American-English child-directed speech from the CHILDES database (MacWhinney, 2000). The corpus is overwhelmingly composed of single-referent naming contexts; multi-referent disambiguation constructions (e.g., "the X or the Y") are rare.

A property of CDS that is directly relevant to our findings is its high rate of noun repetition. Among the nine concrete nouns used in our evaluation battery (*ball, book, car, cup, hat, dog, cat, fish, bird*), the probability of a noun reappearing within the next 3 sentences averages 36.4% (range: 30.6% for *dog* to 42.2% for *ball*), and every test noun participates in hundreds to thousands of 10-sentence windows containing 3 or more mentions (e.g., *ball*: 4,236 windows; *cup*: 1,050). This repetition structure, characteristic of how caregivers scaffold naming ("that's a ball . see the ball? . can you get the ball?"), provides a strong distributional signal favouring repetition priming. By contrast, sentences containing two or more of our target nouns account for only 524 of 893,989 sentences (0.06%), and the majority of these are compound expressions (*kitty cat*) rather than contrastive reference. The distributional environment of CDS thus strongly favours label repetition over label contrast.

3.3 Models

Masked LM (pilot only). BabyBERTa (Huebner et al., 2021): RoBERTa-base architecture, 8K vocabulary, trained on AO-CHILDES individual sentences. We use the published checkpoint.

Autoregressive models (confirmatory). GPT-2 architecture (Radford et al., 2019) trained from scratch on AO-CHILDES. We train a 3×3 grid of model sizes and training durations, with 5 random seeds per cell (45 models total):

The three sizes are Small (4 layers, 128 dim, 4 heads; 2.86M params), Medium (6 layers, 256 dim, 8 heads; 8.88M), and Large (8 layers, 512 dim, 8 heads; 33.50M).

All models share a BPE vocabulary of 8,020 tokens (8,000 merges + 20 nonce tokens added with space-prefix encoding). Vocabulary size was fixed at 8,000 learned BPE units as a practical child-scale setting, matching the BabyBERTa convention for AO-CHILDES, keeping the embedding matrix in proportion to the transformer, and ensuring the common concrete nouns used in evaluation are single tokens; it was not selected through vocabulary-size optimisation. Training durations: 5, 10, and 20 epochs. Optimizer: AdamW, learning rate 5×10^{-4}, cosine schedule, batch size 32, context length 128 tokens. Seeds 0–4. Total wall time: 31.4 hours on an AMD RX 7900 GRE (16 GB) via DirectML.

Parameter counts differ from pre-registration estimates (∼2M, ∼7M, ∼30M) due to the embedding matrix dimension with the 8,020-token vocabulary. We report actual counts throughout.

3.4 ME Evaluation Battery

3.4.1 From Referent Selection to Probability Suppression

Our test does not reproduce the perceptual conditions of developmental ME experiments; it asks whether a text-only model exhibits the probability redistribution an exclusivity-like inference would require in a minimally contrastive discourse. The preamble introduces two referents, x and y; the labelling sentence identifies one as x. If a system applies an exclusivity-like inference, that referent becomes less eligible as the target of a subsequent contrasting reference, so probability mass at the completion position should shift away from x toward y. Suppression of x is thus a textual signature of ME-like inference, not a recreation of the looking-time paradigm. Repetition priming predicts the opposite shift, since recently mentioned material becomes more probable. The test therefore pits two opposing pressures on the same completion position: referential exclusion and repetition-based continuation.

3.4.2 Familiar–Familiar Suppression Track (H1)

We test whether labelling one of two introduced referents as a familiar noun increases or decreases the probability of predicting that noun at a subsequent completion position.

Baseline: "there is a [FAM$_1$] and a [FAM$_2$] . this is a _____"

ME condition: "there is a [FAM$_1$] and a [FAM$_2$] . that is a [FAM$_1$] . this is a _____"

ME predicts $P(\text{FAM}_1 \mid \text{ME}) < P(\text{FAM}_1 \mid \text{baseline})$ (suppression) and $P(\text{FAM}_2 \mid \text{ME}) > P(\text{FAM}_2 \mid \text{baseline})$ (boost). Anti-ME predicts the opposite. We test 10 noun pairs in both directions (20 items per model seed). Log-probabilities are extracted via teacher-forced next-token prediction.

3.4.3 Context-Dependence Diagnostic (H2)

To distinguish genuine ME from embedding-level artefacts, we evaluate nonce-token predictions under five conditions:

- **full_context**: "there is a [FAM] and a [NONCE$_1$] . the [NONCE$_2$] is the ____" (standard ME design)
- **swap_context**: Same with FAM and NONCE$_1$ order reversed

- **nonce_only**: "there is a [NONCE$_1$] . the [NONCE$_2$] is the ____" (no familiar noun)
- **fam_only**: "there is a [FAM] . the [NONCE$_2$] is the ____" (no nonce in context)
- **no_preamble**: "the [NONCE$_2$] is the ____" (no context at all)

ME-consistent = $P(\text{NONCE}_2) > P(\text{FAM})$ at the completion position. 8 items $\times$ 10 embedding initialisation seeds per model. Nonce tokens use Strategy B embedding initialisation: each nonce is anchored to a random rare-noun embedding with 10% Gaussian noise. This is the most conservative available strategy: centroid-based approaches produce artificially clustered nonce embeddings that inflate similarity priming, while purely random initialisation places nonces far from the noun region, making probability comparisons uninformative. Strategy B produces nonce embeddings that are plausibly noun-like but not systematically similar to each other.

3.4.4 Dose–Response Track (H3)

We vary the number of labelling repetitions (0, 1, 2, 3) to test whether the priming advantage (the log-probability difference between the labelled and unlabelled noun) increases monotonically with dose. Evaluated on 5 noun pairs $\times$ 5 model seeds per cell (25 item–seed pairs per cell).

3.4.5 Scoring

All scoring uses log-probability of target tokens at the completion position, extracted via teacher-forced next-token prediction. All target tokens are verified to be single BPE tokens in the shared vocabulary.

3.5 Pilot Findings (Exploratory)

Three findings from the single-model pilot motivate the confirmatory design:

Pilot Finding 1: BabyBERTa cross-sentence insensitivity. BabyBERTa's ⟨mask⟩ predictions are invariant to prior discourse context ($< 10^{-6}$ difference across all conditions). This is a consequence of the training regime (individual sentences with masked token prediction), not a limitation of masked LMs as an architecture class; masked LMs trained on concatenated multi-sentence inputs might show different sensitivity. ME evaluation via multi-sentence suppression design is therefore impossible for this specific checkpoint. We report this as a methodological finding.

Pilot Finding 2: AR repetition priming. On the pilot AR model (6L-256d, PPL 36.4), all 6/6 familiar–familiar pairs show anti-ME: labelling FAM$_1$ increases $P(\text{FAM}_1)$ and decreases $P(\text{FAM}_2)$. Priming ranges from +1.3 to +2.9 nats. This is consistent with CDS statistics: parents repeatedly name objects ("that's a ball . see the ball . get the ball").

Pilot Finding 3: Nonce ME is embedding similarity. The context-dependence diagnostic reveals that nonce_only (7/8 ME-consistent) exceeds full_context (2/8). Introducing any nonce in context primes nonce-region tokens; introducing a familiar noun suppresses them via frequency advantage. The disambiguation context contributes nothing.

3.6 Pre-Registered Hypotheses

Based on the pilot, we pre-registered four hypotheses:

H1 (confirmatory): Anti-ME rate $> 50\%$ in all 9 cells of the training grid. *Test:* One-tailed sign test per cell, $\alpha = 0.05$. *Criterion:* All 9 cells must reach significance.

H2 (confirmatory): nonce_only $\geq$ full_context in all 9 cells. *Test:* Wilcoxon signed-rank test (5 paired seeds), $\alpha = 0.05$. *Criterion:* All 9 cells.

H3 (confirmatory): $>$50% of item–seed pairs show strictly monotonic dose–response in all 9 cells. *Test:* Kendall's τ for trend, bootstrap 95% CI on slope, $\alpha = 0.05$. *Criterion:* All 9 cells $>$50% monotonic.

H4 (confirmatory): Conjunction of H1: no cell shows ME-consistent suppression.

Each hypothesis is treated as a separate pre-registered question. No family-wise error rate correction is applied across hypotheses, as each addresses a distinct theoretical prediction with a distinct control comparison.

4 Results

We report confirmatory results followed by exploratory analyses. All confirmatory tests use the pre-registered statistical procedures and significance thresholds.

In summary: anti-ME repetition priming is significant in all 9 cells (H1 confirmed); the context-dependence diagnostic replicates in all 9 cells (H2 confirmed); dose–response priming shows significant positive trends in all 9 cells, with strict monotonicity in 8/9 (H3 partially confirmed); and no cell

Table 1: Perplexity (mean ± SD across 5 seeds).

	5 epochs	10 epochs	20 epochs
Small (2.9M)	57.8 ± 0.1	48.5 ± 0.1	42.0 ± 0.1
Medium (8.9M)	46.3 ± 0.1	36.6 ± 0.1	28.9 ± 0.0
Large (33.5M)	39.3 ± 0.1	27.0 ± 0.1	15.2 ± 0.1

Table 2: H1 results. Anti-ME rate = proportion of item–seed pairs where labelling FAM_1 increases $P(\text{FAM}_1)$. Priming in nats (natural log units; negative = anti-ME direction).

Cell	Anti-ME	Priming (nats)	Sign test p
Small, 5 ep	100/100	-1.14	7.89×10^{-31}
Small, 10 ep	100/100	-1.47	7.89×10^{-31}
Small, 20 ep	99/100	-1.49	7.97×10^{-29}
Medium, 5 ep	100/100	-1.46	7.89×10^{-31}
Medium, 10 ep	100/100	-1.43	7.89×10^{-31}
Medium, 20 ep	98/100	-0.81	3.98×10^{-27}
Large, 5 ep	100/100	-1.20	7.89×10^{-31}
Large, 10 ep	98/100	-1.06	3.98×10^{-27}
Large, 20 ep	85/100	-0.58	2.41×10^{-13}

shows ME-consistent suppression (H4 confirmed). Priming attenuates with improved language modelling ($\rho = -0.533$) but never crosses zero.

4.1 Training Grid

All 45 models converged without exclusions. Table 1 reports perplexity on a held-out AO-CHILDES test set.

PPL spans a 3.8× range (15.2 to 57.8). Seed variance is negligible (all SDs ≤ 0.1).

4.2 H1: Anti-ME at All Scales (Confirmed)

Anti-ME rates range from 85% to 100%, with all sign tests significant at $p < 2.5 \times 10^{-13}$ (Table 2; Figure 1). Not a single cell shows majority ME-consistent suppression.

Priming attenuates with improved language modelling but never crosses zero. **H1 confirmed.**

4.3 H1 Discourse Ablation

To test whether the anti-ME direction depends on the two-referent contrastive structure or reflects simpler repetition, we re-scored the labelled noun under four preamble variants (Medium, 10 epochs, 5 seeds), measuring priming against a no-labelling baseline. Priming was positive in every condition: full contrastive preamble (+1.43 nats), no preamble (+2.22), single referent, i.e. no competitor (+2.34), and non-labelling repetition, "I saw a x" (+1.11). Across the three conditions that retain labelling, *all* 100 item–seed observations show priming and none shows suppression. The

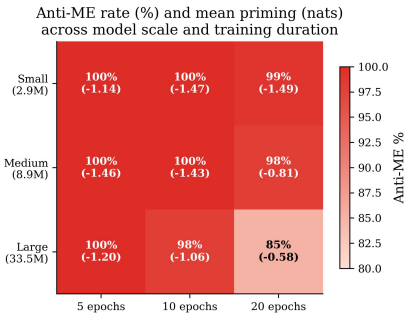


Figure 1: Anti-ME rate (%) and mean priming (nats) across model scale and training duration. All cells show strong anti-ME; priming magnitude decreases toward the bottom-right (larger, more-trained models).

Table 3: H2 results. Mean ME-consistent items out of 8.

Cell	nonce_only	full_context	fam_only
Small, 5 ep	5.0	2.2	∼0
Small, 10 ep	5.7	2.7	∼0
Small, 20 ep	5.3	2.8	∼0
Medium, 5 ep	6.7	2.4	∼0
Medium, 10 ep	6.9	3.3	∼0
Medium, 20 ep	6.4	3.2	∼0
Large, 5 ep	6.0	2.0	∼0
Large, 10 ep	6.2	2.9	∼0
Large, 20 ep	5.7	2.6	∼0

contrastive preamble significantly *reduces* priming relative to the no-preamble and single-referent conditions (Wilcoxon $p < 10^{-17}$), a shift in the ME-predicted direction, but in no labelling condition does priming approach zero. The anti-ME direction is thus not an artefact of the contrastive setup: it survives removing the preamble and the competitor, and is only attenuated, never reversed, when the mention is stripped of its labelling function.

4.4 H2: Context-Dependence Diagnostic Replicates (Confirmed)

nonce_only exceeds full_context in all 9 cells (Table 3; Figure 2). All Wilcoxon tests reach $p = 0.0312$ (the minimum achievable for 5 paired observations all in the same direction).

Aggregated across all 45 models, the five-condition ordering is: nonce_only (6.0/8) > no_preamble (3.1/8) > full_context (2.7/8) > swap_context (2.3/8) > fam_only (0.7/8). This gradient reflects embedding-region priming: nonce context boosts nonce-region predictions; familiar-noun context suppresses them. The disambiguation preamble adds nothing.

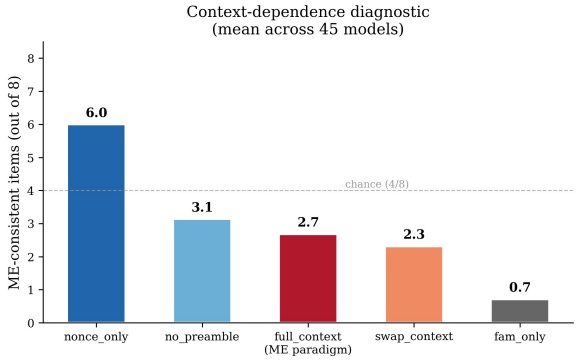


Figure 2: Context-dependence diagnostic (mean across 45 models). The nonce_only condition produces the highest apparent "ME," despite containing no familiar noun and no ME paradigm. The full ME design (full_context) scores below chance. Dashed line: chance (4/8).

Table 4: H3 results.

Cell	Monotonic	Slope	Kendall τ	p
Small, 5 ep	25/25	+0.576	+0.391	2.07×10^{-7}
Small, 10 ep	25/25	+0.847	+0.480	1.98×10^{-10}
Small, 20 ep	22/25	+0.965	+0.421	2.39×10^{-8}
Medium, 5 ep	25/25	+1.080	+0.497	4.19×10^{-11}
Medium, 10 ep	24/25	+1.138	+0.451	2.13×10^{-9}
Medium, 20 ep	15/25	+0.896	+0.241	1.36×10^{-3}
Large, 5 ep	20/25	+1.159	+0.477	2.51×10^{-10}
Large, 10 ep	15/25	+0.976	+0.391	2.14×10^{-7}
Large, 20 ep	12/25	+1.067	+0.317	2.66×10^{-5}

H2 confirmed.

4.5 H3: Dose–Response (Partially Confirmed)

8/9 cells exceed the 50% monotonic threshold (the exception is Large at 20 epochs, 48%). All Kendall's τ are positive and significant (all $p < 0.002$), and all bootstrap 95% CIs on slope exclude zero (Table 4; Figure 3).

Priming reliably increases with dose at all scales (all slopes positive, all trends significant), but strict item-level monotonicity weakens in the best language models. **H3 partially confirmed.**

4.6 H4: No ME at Any Scale (Confirmed)

Conjunction of H1. No cell shows ME-consistent suppression. **H4 confirmed.**

4.7 Exploratory: PPL–Priming Relationship

Across the 45 models, perplexity and priming magnitude are correlated (Spearman $\rho = -0.533$, $p = 0.0002$): better language models show weaker repetition priming. Decomposing this, model size contributes independently ($\rho = +0.434$, $p = 0.003$) as does training duration ($\rho = +0.335$, $p = 0.024$). The relationship is approximately log-linear ($R^2 = 0.44$, vs. linear $R^2 = 0.36$). We do not extrapolate a zero-crossing perplexity from this fit, as any such estimate lies far below the observed range and would not be trustworthy; we note only that within the observed range priming attenuates but remains strictly positive (Figure 4). The GPT-2 positive control (§4.9) probes this trajectory directly at much larger scale and finds attenuation toward zero, not a reversal into ME.

4.8 Exploratory: Per-Item Analysis

Items vary in priming magnitude. The strongest priming pair is cup/book (mean -2.10 nats, 98% anti-ME across all 45 models). The weakest are dog/cat (-0.42 nats, 96% anti-ME) and cat/dog (-0.48 nats, 93% anti-ME). Both weak-priming pairs have natural synonyms in AO-CHILDES (dog/puppy, cat/kitty), potentially diluting the repetition signal. This parallel to bilingual attenuation of ME (Byers-Heinlein and Werker, 2009) is suggestive but requires targeted follow-up.

Per-noun corpus frequency does not predict per-noun priming magnitude (Spearman $\rho = 0.07$, $p = 0.86$, $n = 9$), nor does per-noun repetition rate ($\rho = -0.15$, $p = 0.70$). This suggests that while CDS repetition structure explains the *direction* of the effect (priming rather than suppression), per-item variation in magnitude is driven by other factors, likely properties of the unlabelled competitor noun or pair-specific co-occurrence patterns rather than the labelled noun's frequency alone.

4.9 Positive Control: Pretrained GPT-2

A natural concern is whether the battery could detect ME-consistent suppression if it were present, or whether it is structurally biased toward anti-ME readings. As a positive control, we ran the identical H1 suppression battery on four off-the-shelf pretrained GPT-2 models (Radford et al., 2019) spanning 124M–1.5B parameters, an adult-scale web-text regime 4–45$\times$ larger than our largest child-scale model. These models use the same architecture family but are trained on orders of magnitude more data of a very different kind.

The control reveals a clear scale-dependent trend (Table 5). Mean priming falls monotonically with scale: gpt2 (124M) shows strong anti-ME priming (+2.46 nats, 20/20 items), attenuating through gpt2-medium (+0.97) and gpt2-large (+0.69), until gpt2-xl (1.5B) shows no significant priming (mean -0.31 nats, 7/20 anti-ME, sign test $p = 0.94$). Two

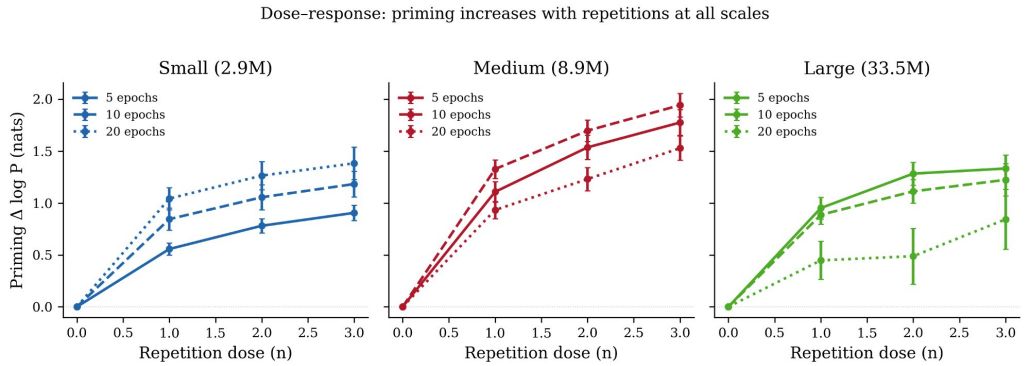


Figure 3: Dose–response curves by model size. Priming advantage (labelled minus unlabelled log-probability) increases with repetition dose at all scales. Error bars: SEM across item–seed pairs. Concavity in the 20-epoch curves (dotted) reflects declining marginal returns on repetition.

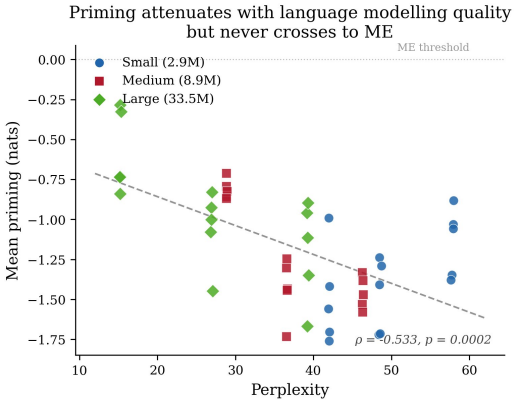


Figure 4: PPL vs. mean priming across 45 models. Each point is one model; shapes indicate size class. Dashed line: linear regression. Dotted line at 0: ME threshold. Priming attenuates with lower perplexity but remains strictly negative (anti-ME).

Model	Params	Anti-ME	Mean (nats)
gpt2	124M	20/20	+2.46
gpt2-medium	355M	18/20	+0.97
gpt2-large	774M	18/20	+0.69
gpt2-xl	1.5B	7/20	−0.31

Table 5: Positive control on pretrained GPT-2. Anti-ME priming attenuates monotonically with scale and is extinguished, but does not reverse into significant ME, by 1.5B parameters. The gpt2-xl mean is dominated by two taxonomic pairs; excluding them yields −0.08 nats (see text).

observations matter. First, the battery is plainly *capable* of moving off the anti-ME regime: the gpt2→gpt2-xl contrast is large and monotonic, so the child-scale null is a genuine measurement rather than a measurement floor. Second, gpt2-xl does not exhibit *ME* so much as an extinguished priming advantage: its slightly negative mean is driven almost entirely by two taxonomic-neighbour pairs (dog/cat −2.03; bird/fish −1.69), and excluding these four ordered pairs returns the mean to −0.08 nats, indistinguishable from zero. The residual suppression is best read as semantic competition between category co-hyponyms that a large model has learned co-occur, not as referential exclusivity. This directly instantiates the discourse-coherence confound that any prompt-based ME measure must contend with, and shows the diagnostic surfaces it rather than masking it.

Taken together with the child-scale grid, the priming advantage attenuates smoothly from +2.5 nats at 2.9M–124M parameters toward zero at 1.5B, but no model at any scale, from child-scale CDS training to adult-scale web text, crosses into significant ME-consistent suppression. This extends the paper's central claim across nearly three orders of magnitude of scale while confirming the instrument's validity.

4.10 Generalisation to a Non-CDS Corpus

To test whether the dissociation is specific to child-directed speech, we trained the Medium architecture (5 seeds, 10 epochs) on a size-matched slice of WikiText-103 (Merity et al., 2017), adult encyclopedic prose, truncated to the same token count as AO-CHILDES. This holds data volume fixed and varies register.

Anti-ME priming persists: 96/100 items across the five WikiText models (mean −0.72 nats), with no model showing ME-consistent suppression. The context-dependence diagnostic also replicates, with the same ordering (nonce_only highest at 5.0/8, fam_only lowest at 2.2/8). The dissociation is there-

fore not an artefact of CDS.

Priming is, however, significantly weaker on WikiText than on the matched CDS cell (-0.72 vs -1.43 nats; Mann–Whitney $U = 0$, $p = 0.008$; every CDS model primes more than every WikiText model). Two points bear on the interpretation. First, this attenuation is *not* a repetition-frequency effect: the nine battery nouns have nearly identical mean 3-sentence repetition rates in the two corpora (35.2% in WikiText vs 36.4% in AO-CHILDES). Rather, in encyclopedic text these nouns occur predominantly in non-object senses and frames ("book" as a published work, "cup" as a trophy, "car" as a railway carriage), weakening the object-naming pattern that drives priming. Second, the WikiText models are substantially weaker language models (PPL ≈198 vs ≈37 for the CDS Medium cell). This runs *opposite* to the within-CDS quality trend, where *better* language models prime *less* ($\rho = -0.533$, §4.7): if model quality drove the difference, the far weaker WikiText models should prime *more*, not less. The attenuation is thus attributable to register and lexical sense rather than to model quality. Critically, the qualitative dissociation (priming, never ME) is genre-invariant.

5 Discussion

5.1 Repetition Priming Without Mutual Exclusivity

The central finding is a dissociation: autoregressive language models trained on child-directed speech develop robust repetition priming but do not develop mutual exclusivity at any scale tested. Better language models attenuate their priming (§4.7), and the GPT-2 control (§4.9) extends this to 1.5B parameters and adult web text: priming is extinguished but never reverses into significant ME, with no sign of an approaching sign change at any scale.

5.2 What Text-Only Learning Lacks

Our findings converge with the grounded models literature. Nortje et al. (2024) demonstrated ME in visually grounded speech models, Oneaţă et al. (2025) showed weaker ME in bilingual VGS models paralleling human data, and Thai et al. (2025) found weak ME bias even in state-of-the-art VLMs. The pattern across these results and ours is consistent: positive ME demonstrations have involved models with referents, whereas over text alone the available mechanisms (embedding similarity and discourse-frequency effects) do not produce lexical exclusivity. This aligns with the Bayesian word-learning framework (Xu and Tenenbaum, 2007), in which the critical ingredient is the referent set, and with evidence that ME in children is graded and experience-modulated (Lewis et al., 2020). The tested text-only models do not represent the competing referent set that standard accounts of ME assume. An alternative reading, however, is that ME follows not from grounding per se but from an explicit lexical-competition objective (Gulordava et al., 2020); we cannot fully separate the two, since grounded models typically also introduce competition. Both accounts locate the missing ingredient outside text-only next-token prediction, the level at which our claim operates, and our cross-study grounding evidence is correlational rather than a direct within-architecture manipulation.

5.3 The Diagnostic and the Scope of the Claim

Beyond the ME-specific findings, the context-dependence diagnostic has broader utility: if an apparent pragmatic effect survives removal of the pragmatically relevant context, it is not pragmatic. Stripping the disambiguation preamble while retaining nonce context *increased* the apparent ME signal, revealing embedding similarity as the driver, a test applicable to any prompt-based claim of pragmatic competence in LMs. The H1 ablation (§4.3) applies the same logic to the suppression track, confirming the anti-ME direction is repetition-driven rather than a response to the referential contrast.

We do not claim that distributional learning is fundamentally inadequate for referential pragmatics, nor that ME requires innate machinery. The claim is narrower: input modality constrains what referential mechanisms can be learned, and under child-scale text-only input the distributional signal favours repetition over exclusivity. Whether different architectures, objectives, or larger scales could overcome this is open; our data neither support nor rule it out, consistent with the BabyLM multimodal track (Warstadt et al., 2023; Charpentier et al., 2025).

6 Conclusion

Child-scale language models trained on CDS develop repetition-based reference tracking rather than mutual exclusivity, a dissociation robust across scale (to 1.5B parameters) and genre. The tested text-only models do not explicitly represent the competing referent set that standard accounts

of ME assume, while prior positive demonstrations involve explicit referents or objectives that impose lexical competition. Whether grounding, referent-set representation, or lexical competition is the operative ingredient is the question these results motivate.

Limitations

Several constraints bound our conclusions. Both corpora are English (AO-CHILDES and the Wiki-Text replication of §4.10); our largest from-scratch model is 33.5M parameters, and though the GPT-2 control reaches 1.5B without reversal, we cannot rule out change at larger scales. We hold training-data volume fixed at the developmentally plausible $\sim$4M-token scale rather than varying it; the GPT-2 control, trained on orders of magnitude more data, shows the dissociation persists far beyond this regime, but we do not isolate data quantity as a factor within our controlled grid. We did not optimise architecture for AO-CHILDES, though the effect's consistency across a $12\times$ parameter range argues against architecture-specificity. Most importantly, a direct manipulation of grounding within one architecture, rather than the cross-study comparison we rely on, remains the key next step.

References

Krista Byers-Heinlein and Janet F. Werker. 2009. Monolingual, bilingual, trilingual: Infants' language experience influences the development of a word-learning heuristic. *Developmental Science*, 12(5):815–823.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420.

Kanishk Gandhi and Brenden M. Lake. 2020. Mutual exclusivity as a challenge for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 33.

Kristina Gulordava, Thomas Brochhagen, and Gemma Boleda. 2020. Deep daxes: Mutual exclusivity arises through both learning biases and pragmatic strategies in neural networks. In *Proceedings of the 42nd Annual Meeting of the Cognitive Science Society*, pages 2089–2095.

Justin Halberda. 2003. The development of a word-learning strategy. *Cognition*, 87(1):B23–B34.

Philip A. Huebner, Elior Sulem, Cynthia Fisher, and Dan Roth. 2021. BabyBERTa: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 624–646.

Molly Lewis, Veronica Cristiano, Brenden M. Lake, Tammy Kwan, and Michael C. Frank. 2020. The role of developmental change and linguistic experience in the mutual exclusivity effect. *Cognition*, 198:104191.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, 3rd edition. Erlbaum.

Ellen M. Markman and Gwyn F. Wachtel. 1988. Children's use of mutual exclusivity to constrain the meanings of words. *Cognitive Psychology*, 20(2):121–157.

Bob McMurray, Jessica S. Horst, and Larissa K. Samuelson. 2012. Word learning emerges from the interaction of online referent selection and slow associative learning. *Psychological Review*, 119(4):831–877.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

Leanne Nortje, Dan Oneaţă, Yevgen Matusevych, and Herman Kamper. 2024. Visually grounded speech models have a mutual exclusivity bias. *Transactions of the Association for Computational Linguistics*, 12:755–770.

Xenia Ohmer, Michael Franke, and Peter König. 2022. Mutual exclusivity in pragmatic agents. *Cognitive Science*, 46(1):e13069.

Dan Oneaţă, Leanne Nortje, Yevgen Matusevych, and Herman Kamper. 2025. The mutual exclusivity bias of bilingual visually grounded speech models. In *Proceedings of Interspeech 2025*, pages 5043–5047.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Anh Thai, Stefan Stojanov, Zixuan Huang, Bikram Boote, and James M. Rehg. 2025. MEBench: A novel benchmark for understanding mutual exclusivity bias in vision-language models. *arXiv preprint arXiv:2505.20122*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the

BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Fei Xu and Joshua B. Tenenbaum. 2007. Word learning as Bayesian inference. *Psychological Review*, 114(2):245–272.

Reading Acquisition Order out of BabyLM Representations: Child-Scale Training Produces Developmentally Aligned Semantic Spaces

Chenyi Ding
University of California, Davis

Zachary N. Houghton
University of Oregon
Vail Systems, Inc

Masoud Jasbi
University of California, Davis

Abstract

The age at which children learn different words, also known as Age of Acquisition (AoA), can be predicted from several word-level properties such as word frequency and concreteness. We ask whether acquisition order information is also implicitly present in the internal representations of language models, and what linguistic factors drive this. We probe 18 public checkpoints that span three architecture families at matched size, data scales from 10 million to roughly 180 billion words, and the winning systems and official baselines of the three previous BabyLM challenges. A linear ridge read-out of frozen hidden states predicts Wordbank age of acquisition for 619 words in the communicative development inventory (CDI), with the probing layer chosen by nested cross-validation. Models trained on child-scale data match or exceed their full-scale counterparts in every family. Masked and hybrid training objectives yield representations from which acquisition order is easier to recover; challenge winners do not best capture children's acquisition order. Geometric measures of semantic structure add predictive value beyond classic predictors only for child-scale models. AoA decodability thus provides a training-free measure of how closely a model's representations track children's real acquisition order.

1 Introduction

Current estimates suggest that by the time they are six year old, children hear somewhere between 7 to 80 million words, depending on what kind of environment they grow up in (Frank, 2023). Yet, they master their native language long before modern language models trained on thousands of times more text (Warstadt et al., 2023). The BabyLM Challenge treats this "data gap" as a research opportunity: models trained on developmentally plausible amounts of text can serve both as more efficient learners and as candidate models of human language acquisition (Charpentier et al., 2025). One concrete way to test the second role is to ask how much of children's actual learning is recoverable from such models. Parent-reported vocabularies collected in the Wordbank repository show that the variance in acquisition order is reliable(Braginsky et al., 2016), and that small set of word-level properties account for a third of it.

Language models have entered this literature from the behavioral side. Chang and Bergen (2022) track the surprisal of individual words across training checkpoints and derive an age of acquisition for the model itself, and Portelance et al. (2023) show that language model surprisal predicts children's age of acquisition across five languages, with the strongest effects for predicates and function words. The third BabyLM Challenge adopted the checkpoint-based method as one of its hidden evaluation tasks (Charpentier et al., 2025). These approaches predict acquisition from model behavior, either over training time or at the output layer. What the trained representation itself encodes about acquisition order has, to our knowledge, been examined only for single models and never compared across training regimes.

In this paper, we treat the decodability of children's age of acquisition (AoA) from frozen representations as a measurable property of a trained model: how much of children's real acquisition order its representations recover. We call this property "developmental alignment", and we measure it across 18 public checkpoints, spanning three architecture families with matched parameter counts (OPT-125M, GPT-2, RoBERTa), training data from 10 million to roughly 180 billion words, the winning systems of the 2023 and 2024 challenges (ELC-BERT and GPT-BERT), and the official baselines of all three challenge years, including the 2025 GPT-BERT baselines that differ only in the ratio of masked to causal training.

Proceedings of the Second BabyLM Workshop, pages 250–259
October 29, 2026 ©2026 Association for Computational Linguistics

For each checkpoint we aggregate hidden states for 619 words in the Communicative Development Inventory (CDI) (Frank et al., 2017) over child-directed sentences from CHILDES, train a ridge read-out against parent-reported age of acquisition (AoA) from the Wordbank repository under cross-validation with nested layer selection, and compare how much acquisition order each checkpoint's representations recover.

Three architecture families tell the same story: checkpoints trained on child-scale corpora match or exceed counterparts trained on three to four orders of magnitude more text in AoA decodability. The effect is largest in the GPT-2 family, where the BabyLM-trained baseline reaches $\rho = .62$ against $.48$ for the full-scale model. Training objective matters as much as data. At identical data and architecture, the masked-focused and mixed GPT-BERT variants reach the highest scores overall ($\rho = .637$ and $.639$), while the purely causal variant trails them. The winners of previous challenges, in contrast, sit in the lower half of the ranking, below several plain baselines. Finally, three geometric measures of semantic neighborhood structure add predictive value over the classic word-level predictors of Braginsky et al. (2016) for child-scale models, and add essentially nothing for the full-scale OPT model.

Our contributions are threefold. First, we introduce AoA decodability as a training-free way to measure how closely a model's representations track children's real acquisition order (requiring a single frozen checkpoint rather than a full trajectory of intermediate checkpoints). Second, we provide the first systematic comparison of this property across data scales, architectures, and training objectives, including the winners and baselines of all previous BabyLM challenges. Third, we show that leaderboard performance and developmental alignment dissociate, which has consequences for how the community evaluates progress toward human-like learning.

2 Background

Acquisition order and lexical structure. Which words children learn first is not arbitrary. Parent-report vocabularies collected in the Wordbank repository show that acquisition order is stable across children, (split-half $r^2 > 0.98$; Braginsky et al. 2016), and that a small set of word-level properties accounts for roughly half of the variance. Beyond these word-level predictors, the structure of the developing vocabulary also predicts acquisition order. Hills et al. (2009) show that early semantic networks grow by preferential acquisition, with new words entering according to their semantic relatedness to words already in the child's vocabulary, and Fourtassi et al. (2020) find that semantic-network connectivity predicts acquisition across languages. Measures defined over a semantic space as a whole, rather than over isolated word properties, therefore have a natural place in accounts of vocabulary growth. This is why in this study, we define three measures (cohesion, distinctiveness, and diversity) over the representation space of language models to test their predictive accuracy against more traditional measures such as frequency.

From behavior to representations. Previous work connecting language models to acquisition order has so far been behavioral. Portelance et al. (2020) predict CDI-based AoA from the average surprisal of an LSTM trained on child-directed speech, with gains over frequency for predicates and function words. Chang and Bergen (2022) define an age of acquisition for a model as the training step at which the surprisal of a word crosses a learning threshold, and the third BabyLM Challenge turned this method into an evaluation task (Charpentier et al., 2025). Portelance et al. (2023) extend surprisal-based prediction to five languages. A parallel line elicits norm-like judgments from instruction-tuned LLMs, including adult AoA ratings (Trott, 2024) and batteries of lexical features tested against child AoA (Hagihara and Miyazawa, 2026). Behavioral measures have a clear advantage: they require nothing beyond output probabilities. They also have two costs. Trajectory-based measures need a full series of intermediate checkpoints, which public models rarely provide, and output probabilities compress the model's complex representation of a given input into a single number per word; Bunzeck and Zarrieß (2025) show that in-context surprisal can hide exactly the lexical deficits it is meant to measure. Probing the representation avoids both. Following Bommasani et al. (2020), we represent each word by its hidden state averaged over occurrences in context, and we ask how much of children's acquisition order a linear read-out recovers from these frozen states. In preliminary experiments with a single child-scale model,

this read-out recovered acquisition order far better than fine-tuning the same model end to end on a corpus-based AoA signal derived from children's first productions in CHILDES (Smolík and Filip, 2022), so we keep every model frozen throughout.

Within the workshop, Capone et al. (2024) argue against age of acquisition as a criterion for ordering training data, because linguistically similar words of comparable difficulty can be acquired at very different ages. Our use is diagnostic rather than curricular: AoA is the quantity we decode from a trained model, not a signal we train on.

3 Methods

We evaluate 18 public checkpoints across three families. For each checkpoint, we extract contextual word representations for 619 CDI words, construct a four-predictor AoA baseline model, and then train a nested cross-validated ridge readout on the frozen hidden states. We then define three geometric measures of semantic neighborhood structure, namely cohesion, distinctiveness, and diversity, capturing how tightly a word clusters with semantically related words (cohesion), how distinct it is from unrelated ones (distinctiveness), and how densely populated its neighborhood is (diversity), and test whether they predict AoA above and beyond the baseline predictors. [1]

Models. Table 1 lists the 18 public checkpoints. The core of the set is three architecture families with matched data-scale pairs: OPT-125M, GPT-2, and RoBERTa. Each family pairs child-scale checkpoints with the original full-scale release. For OPT-125M (Zhang et al., 2022), the child-scale checkpoints are trained on the BabyLM corpus at 10M and 100M words (Warstadt et al., 2023) and on the roughly 150M-token BabyLM v3 corpus (Houghton et al., 2026); the full-scale reference is the original release trained on roughly 180B tokens. For GPT-2 (Radford et al., 2019), the child-scale checkpoints are the 2025 baselines trained on 10M and 100M words (Charpentier et al., 2025); the full-scale counterpart is the original WebText checkpoint. For RoBERTa (Liu et al., 2019), the child-scale checkpoint is the 2023 BabyLM baseline (Warstadt et al., 2023), set against the original release. Beyond these pairs, the set includes the 2023 winning system ELC-BERT (Charpentier and Samuel, 2023), the 2024 winning architecture GPT-BERT (Charpentier and Samuel, 2024), the LTG-BERT baseline (Samuel et al., 2023; Hu et al., 2024), the BabyLlama distillation line (Timiryasov and Tastet, 2023; Hu et al., 2024), and the three 2025 GPT-BERT baselines, which share data and architecture and differ only in the ratio of masked to causal language modeling (Charpentier et al., 2025).

Word representations. For each checkpoint we extract hidden states at every layer. In the contextual condition, each CDI word is read inside up to 50 child-directed sentences drawn from CHILDES (MacWhinney, 2000); the hidden states of its subword tokens are averaged within each sentence and then across sentences (Bommasani et al., 2020). In the bare condition the word is presented alone. Because the 18 models use different tokenizers, we locate each target word by its character position in the original sentence rather than matching tokens directly. All representations are extracted once; no model is updated at any point.

Acquisition target and baseline predictors. The target is production-based age of acquisition for 619 items of the English CDI of the English Words and Sentences CDI, spanning sound effects and early nouns acquired by 16 months (*woof woof*, *dog*, *mommy*) through function words not reliably produced until 30 months (*if*, *but*, *would*), defined as the age in months at which half of children are reported to produce the word (Frank et al., 2017). The reference model is the four-predictor regression of Braginsky et al. (2016): log frequency in child-directed speech, mean length of utterance, concreteness (Brysbaert et al., 2014), and association with babies, which they call babiness (Perry et al., 2015). Under our cross-validation this baseline reaches $\rho = .707$.

Read-out. For every checkpoint, layer, and representation type, we fit a ridge regression ($\lambda = 10$) that uses the frozen hidden states to predict AoA, under five-fold cross-validation, features z-scored within each training fold. We operationalize score as the Spearman correlation between pooled held-out predictions and observed AoA. We use a linear probe because less expressive probes limit the risk that the probe itself, rather than the representation, carries the signal (Hewitt and Liang, 2019; Belinkov, 2022). We select a single layer rather than averaging across all layers because different depths encode different kinds of information, and

[1] Data and analysis code are available at https://github.com/znhoughton/aoa-pipeline.

Checkpoint	Family	Training data	Objective	Source
opt125-babylm150m	OPT-125M	BabyLM v3 150M	causal	Houghton et al. (2026)
opt125-babylm100m	OPT-125M	BabyLM 100M	causal	Warstadt et al. (2023)
opt125-babylm10m	OPT-125M	BabyLM 10M	causal	Warstadt et al. (2023)
opt125-full	OPT-125M	∼180B tokens	causal	Zhang et al. (2022)
gpt2-babylm100m	GPT-2	BabyLM 100M	causal	Charpentier et al. (2025)
gpt2-babylm10m	GPT-2	BabyLM 10M	causal	Charpentier et al. (2025)
gpt2-full	GPT-2	WebText (40 GB)	causal	Radford et al. (2019)
babyllama58-2023	Llama	BabyLM 10M	causal + distillation	Timiryasov and Tastet (2023)
babyllama-100m	Llama	BabyLM 100M	causal + distillation	Hu et al. (2024)
babyllama-10m	Llama	BabyLM 10M	causal + distillation	Hu et al. (2024)
roberta-babylm100m	RoBERTa	BabyLM 100M	masked	Warstadt et al. (2023)
roberta-full	RoBERTa	160 GB text	masked	Liu et al. (2019)
elcbert-100m-2023	LTG-BERT	BabyLM 100M	masked	Charpentier and Samuel (2023)
gptbert-100m-2024	GPT-BERT	BabyLM 100M	hybrid	Charpentier and Samuel (2024)
ltgbert-100m-2024	LTG-BERT	BabyLM 100M	masked	Samuel et al. (2023)
gptbert25-mixed	GPT-BERT	BabyLM 100M	hybrid, 50% causal	Charpentier et al. (2025)
gptbert25-causal	GPT-BERT	BabyLM 100M	hybrid, 93.75% causal	Charpentier et al. (2025)
gptbert25-masked	GPT-BERT	BabyLM 100M	hybrid, 6.25% causal	Charpentier et al. (2025)

Table 1: The 18 checkpoints, grouped by architecture family. HuggingFace identifiers are listed in Appendix A.

averaging would mix a strong signal at one depth with noise from the rest. Because the best layer varies across models, the reported numbers use nested cross-validation: within each outer training fold, an inner five-fold loop selects the layer, which is then evaluated on the held-out outer fold. This keeps the layer choice from ever seeing the words it is scored on, avoiding the optimistic bias of picking the best layer post hoc. We report 95% confidence intervals from 2,000 bootstrap resamples over words, and we compare each checkpoint to the full-data OPT reference on their common valid words.

Semantic structure. From each representation space we compute three measures per word. Let $\mathbf{w}_i$ be the representation of word i, let $s(i, j)$ be the cosine similarity between $\mathbf{w}_i$ and $\mathbf{w}_j$, and let $C(i)$ be the set of other words sharing the CDI category of word i. Cohesion is the mean similarity of a word to its own category,

$$\mathrm{coh}(i) = \frac{1}{|C(i)|} \sum_{j \in C(i)} s(i, j), \qquad (1)$$

distinctiveness is cohesion minus the mean similarity to all words outside the category,

$$\mathrm{dist}(i) = \mathrm{coh}(i) - \frac{1}{|\bar{C}(i)|} \sum_{j \in \bar{C}(i)} s(i, j), \qquad (2)$$

where $\bar{C}(i)$ contains every word outside the category of i, and diversity is the entropy of the full similarity distribution of word i, computed without category labels,

$$\mathrm{div}(i) = -\sum_{b=1}^{B} p_b(i) \log p_b(i), \qquad (3)$$

where $p_b(i)$ is the proportion of the similarities $\{s(i, j) : j \neq i\}$ that fall into bin b of a B-bin histogram over $[-1, 1]$, with $B = 20$. Each set of measures is added to the four baseline predictors in the same cross-validated regression.

Robustness. Hidden layer sizes range from 512 to 768, so we repeat the read-out after projecting every space to 256 principal components; the ranking of checkpoints is essentially unchanged (Spearman $\rho = .922$ with the main analysis). Contextual representations outperform bare ones throughout, consistent with Bommasani et al. (2020), and the main text reports contextual results.

4 Results

We organize the results around four central findings. First, child-scale checkpoints match or exceed full-scale checkpoints within every architecture family. Second, training objective matters as much as data scale for how much of children's acquisition order a model's representations capture, with masked and hybrid objectives ahead of purely causal training. Third, winning previous BabyLM challenges does not predict how well a model captures children's acquisition order. Fourth, a word's semantic neighborhood structure in a model's representation space carries information about the children's acquisition order above

Checkpoint	ρ [95% CI]	Δ vs. ref.
gptbert25-mixed	.639 [.584, .685]	$+.079^*$
gptbert25-masked	.637 [.585, .685]	$+.078^*$
gpt2-babylm100m	.626 [.572, .674]	$+.068^*$
ltgbert-100m-2024	.623 [.568, .677]	$+.064^*$
babyllama-100m	.618 [.563, .667]	$+.058$
gpt2-babylm10m	.604 [.545, .657]	$+.043$
roberta-babylm100m	.583 [.529, .638]	$+.024$
gptbert25-causal	.583 [.523, .640]	$+.023$
opt125-babylm100m	.577 [.518, .630]	$+.017$
opt125-babylm150m	.571 [.509, .627]	$+.012$
babyllama-10m	.571 [.510, .628]	$+.010$
opt125-full (ref.)	.560 [.500, .615]	—
roberta-full	.525 [.455, .592]	$-.035$
elcbert-100m-2023	.523 [.460, .581]	$-.039$
gptbert-100m-2024	.502 [.438, .563]	$-.059$
babyllama58-2023	.489 [.423, .551]	$-.070^*$
gpt2-full	.483 [.415, .545]	$-.077^*$
opt125-babylm10m	.414 [.346, .481]	$-.145^*$

Table 2: Nested read-out scores and pairwise differences against the full-data OPT reference on common valid words. Asterisks mark differences whose 95% bootstrap interval excludes zero.

and beyond what the classic word-level predictors capture, but only for child-scale checkpoints.

Figure 1 and Table 2 report the nested read-out score for every checkpoint. No read-out reaches the $\rho = .707$ of the Braginsky et al. (2016) baseline, which is expected: the baseline uses measured properties of exactly the kind corpora and adult raters provide, while the read-out sees nothing but the representation. The informative comparisons are between checkpoints.

Child-scale data matches or exceeds full-scale data. In every family the child-scale checkpoint at least matches its full-scale counterpart. The contrast is sharpest for GPT-2: the 2025 baseline trained on 100M words reaches $\rho = .626$ [.572, .674], while the WebText checkpoint reaches .483 [.415, .545]. Against the shared reference model, the former is reliably higher ($\Delta\rho = +.068$, 95% CI [.006, .124]) and the latter reliably lower ($-.077$ [$-.141$, $-.016$]). In the OPT family the 150M-token checkpoint reaches .571 against .560 for the 180B-token original, and in the RoBERTa family the BabyLM baseline reaches .583 against .525. Matched or better, in other words, despite three to four orders of magnitude less text. Data volume within the child-scale regime shows no consistent advantage either: the GPT-2 10M baseline (.604) is comparable to the 100M variant, while the OPT 10M baseline falls well below the reference ($-.145$ [$-.219$, $-.073$]), so the effect of very small data interacts with the training recipe.

Objective mix matters as much as data scale. The three 2025 GPT-BERT baselines share data and architecture and differ only in the ratio of causal to masked training. The mixed ($\rho = .639$ [.584, .685]) and masked-focused (.637 [.585, .685]) variants top the entire ranking and are reliably above the reference ($+.079$ [.021, .137] and $+.078$ [.010, .143]); the causal-focused variant reaches .583, with a confidence interval spanning zero ($+.023$ [$-.040$, .087]). The purely masked LTG-BERT baseline is also reliably above the reference ($+.064$ [.001, .128]). Layer profiles separate the two model classes: encoder and hybrid checkpoints reach their peak in the upper third of the network, while decoder profiles are flatter across depth (Figure 2).

Benchmark winners do not best capture children's acquisition order. The winning systems of the first two challenges sit in the lower half of the ranking: ELC-BERT reaches $\rho = .523$ [.460, .581] and the 2024 GPT-BERT submission reaches .502 [.438, .563], below the naive GPT-2 baselines trained on the same corpus. Both systems were selected largely by aggregate accuracy on benchmark suites built around BLiMP (Warstadt et al., 2020). Whatever the challenge leaderboard selects for, it is not the ability to capture children's real acquisition order. The nested protocol also acts as a check on optimistic layer selection here: the distilled babyllama-100m baseline leads the post hoc ranking ($\rho = .671$ at the first layer) but drops to .618 under nested selection, with unstable layer choices across folds, so we treat effects of distillation as unresolved rather than as a finding.

Semantic structure separates the two regimes most sharply. The three neighborhood measures alone predict acquisition order at $\rho = .33$ to .52 from the representation spaces of child-scale checkpoints, but only .19 to .27 from the three full-scale spaces. Added to the four Braginsky et al. predictors, the measures raise the cross-validated correlation from .707 to as high as .756 (roberta-babylm100m) among child-scale checkpoints, while the full-scale OPT space adds nothing (.707 to .708). These analyses use the best layer of each space and are exploratory; the point is their consistency across fifteen child-scale checkpoints, not any single comparison.

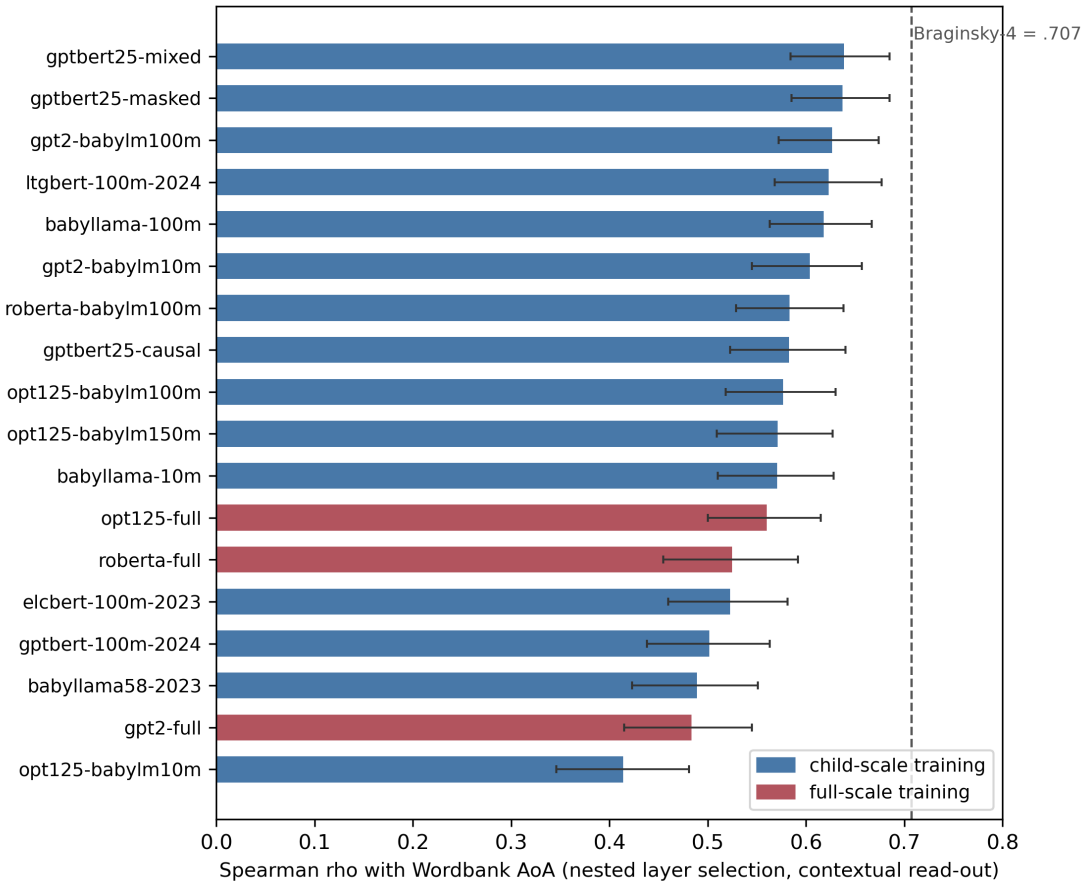


Figure 1: Nested read-out score per checkpoint (contextual representations), with 95% bootstrap confidence intervals. Blue bars are child-scale checkpoints; red bars are full-scale checkpoints. The dashed line marks the four-predictor baseline of Braginsky et al. (2016).

5 Discussion

We asked whether the order in which children learn words is recoverable from the internal representations of trained language models, and what kind of training makes it recoverable. Across 18 checkpoints, a linear read-out of frozen representations predicts children's age of acquisition well above chance, and four results stand out. Child-scale training matched or exceeded full-scale training in all three architecture families, with the sharpest gap in GPT-2 ($\rho = .626$ against .483). Training objective mattered as much as data scale: masked and hybrid objectives produced representations from which acquisition order was easiest to recover, while purely causal training lagged behind. The winning systems of previous challenges, by contrast, did not best capture children's acquisition order. And geometric measures of semantic structure carried acquisition-relevant information beyond the classic predictors only in child-scale representations. We take up each result in turn below.

Why is more of children's acquisition order recoverable from child-scale training? The most direct explanation is register. The BabyLM corpus consists mostly of transcribed or scripted speech, and about 40% of it comes from child-directed or child-appropriate sources (Warstadt et al., 2023), so the distributional statistics a child-scale model compresses are closer to the statistics children learn from. A more skeptical variant of this explanation attributes the effect to our extraction contexts: word representations are aggregated over CHILDES sentences, which are in-domain for child-scale models and out-of-domain for full-scale ones. Two observations speak against a purely skeptical reading. The full-scale models are far from floor (the weakest, GPT-2, still reaches $\rho = .483$), and the semantic-structure result concerns relative geometry within each space rather than fit to the extraction sentences. The decisive test is to repeat the extraction with register-neutral

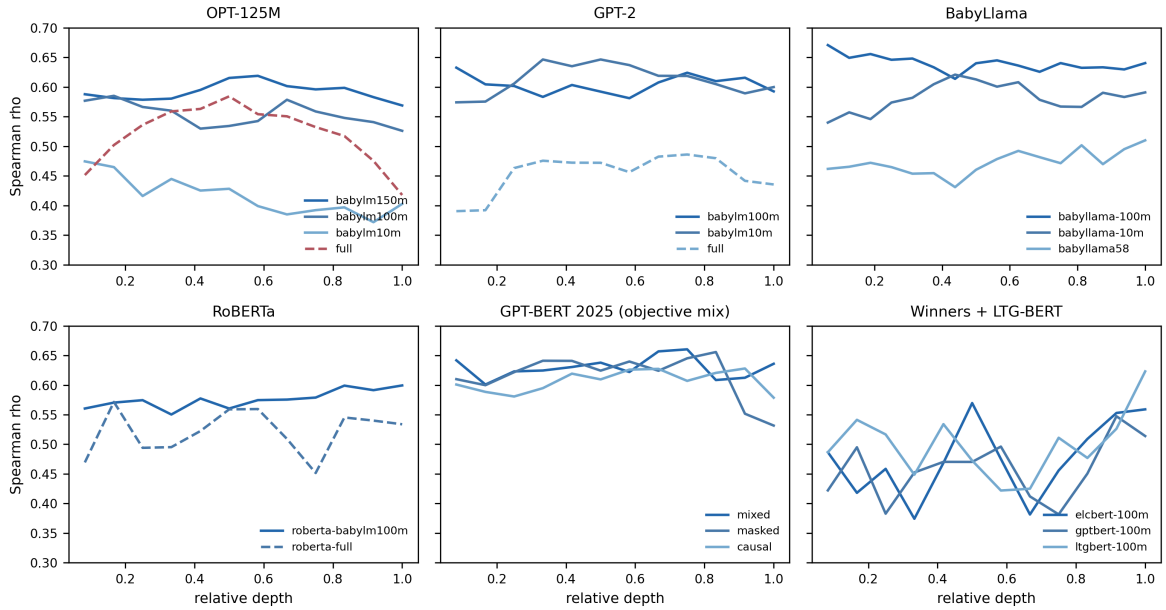


Figure 2: Read-out score by relative depth, per architecture family. Dashed lines are full-scale checkpoints; solid lines are child-scale checkpoints. The first three panels show purely causal models (OPT-125M, GPT-2, Llama); the remaining panels show masked (RoBERTa, LTG-BERT, ELC-BERT) and hybrid (GPT-BERT) objectives.

contexts, and we list it as the first item for future work. The register account also has a precedent with a twist: Feng et al. (2024) find that models trained on child-directed speech alone perform worst among size-matched datasets on syntactic and lexical-semantic benchmarks. Register match does not buy benchmark performance; our results identify what it does buy, namely developmental alignment of the semantic space.

Data quantity provides a second candidate mechanism. In the reading-time literature, surprisal from larger pre-trained transformers fits human reading times strictly worse within a model family (Oh and Schuler, 2023), and GPT-2 yields better psychometric fit than models a thousand times its size (Shain et al., 2024). Oh and Schuler (2023) link this trend to the capacity of larger models to memorize training sequences,[2] which pushes their predictions of open-class words beyond humanlike levels. Child-scale and full-scale checkpoints differ in register and in quantity at once, so our design does not separate the two accounts. Within the child-scale range the data lean toward register, since moving from 10M to 100M words does not consistently change alignment, but the memorization account concerns scales far beyond this range. A scale ladder on a fixed corpus would separate the two accounts, and we list it alongside the register-neutral extraction as future work.

The objective effect has a candidate mechanism. A word's masked-prediction loss depends on how well the model reconstructs it from surrounding words, so masked training organizes the space by substitutability in context. Substitutability is close to the neighborhood structure that predicts children's acquisition (Hills et al., 2009; Fourtassi et al., 2020), which may explain why masked and hybrid objectives yield spaces from which acquisition order is easier to read. Causal training optimizes next-word generation, and what it stores about a word is spread over the positions where that word helps prediction rather than concentrated in the word's own representation.

For the BabyLM community the practical point is that AoA decodability is a cheap evaluation. It needs one frozen checkpoint, public child data, and a ridge regression: no intermediate checkpoints, no fine-tuning, no generation. It complements the official word-learning task (Charpentier et al., 2025), which reads acquisition from surprisal trajectories and therefore requires the full training run. The dissociation between leaderboard rank and how well a model captures children's acquisition order gives the human-likeness

[2]This pattern of models relying on memorization where humans may not extends beyond the reading-time literature as well, such as in ordering preferences of binomials (Houghton et al., 2025; Houghton, 2025).

track a concrete target, since architectures that win on aggregate benchmarks do not automatically produce developmentally aligned representations. The method also transfers directly to the 2026 multilingual track: CDI instruments have been adapted well beyond English and Wordbank hosts norms for Dutch and Mandarin (Frank et al., 2017), the two new languages of that track (Choshen et al., 2026), and every step of the pipeline is language-agnostic.

6 Conclusion

This study measured how much of children's word-learning order can be read out of the frozen representations of 18 public checkpoints. Child-scale training matched or exceeded full-scale training in all three architecture families, masked and hybrid objectives produced representations from which acquisition order was easiest to recover, and the winners of previous challenges did not best capture children's acquisition order. Children's acquisition order is already present in the representations of well-trained BabyLM systems; whether models can also be made to acquire words in a child-like order over training is the natural next question, and the intermediate checkpoints the community already releases make both questions answerable with the same data.

Limitations

First, all results concern English; the multilingual extension is planned but not yet run. Second, the acquisition target is parent-reported production AoA for 619 CDI items, which covers early vocabulary only and inherits the biases of parental report. Third, contextual representations are extracted from CHILDES sentences, which may favor models trained on similar registers; a replication with register-neutral extraction contexts is the clearest next check. Fourth, each checkpoint represents a single training run, so seed variance is unquantified, and the checkpoints differ in tokenizer and training details beyond the axes we analyze. Finally, decodability is correlational: a linear probe recovering acquisition order from a model's representation does not show that children exploit the same structure.

References

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781.

Mika Braginsky, Daniel Yurovsky, Virginia A. Marchman, and Michael C. Frank. 2016. From uh-oh to tomorrow: Predicting age of acquisition for early words across languages. In *Proceedings of the 38th Annual Conference of the Cognitive Science Society*.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, 46(3):904–911.

Bastian Bunzeck and Sina Zarrieß. 2025. Subword models struggle with word learning, but surprisal hides it. *arXiv preprint arXiv:2502.12835*.

Luca Capone, Alessandro Bondielli, and Alessandro Lenci. 2024. ConcreteGPT: A baby GPT-2 based on lexical concreteness and curriculum learning. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 189–196.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every Layer Counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Steven Y. Feng, Noah D. Goodman, and Michael C. Frank. 2024. Is child-directed speech effective training data for language models? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Abdellah Fourtassi, Yuan Bian, and Michael C. Frank. 2020. The growth of children's semantic and phonological networks: Insight from 10 languages. *Cognitive Science*, 44(7):e12847.

Michael C. Frank. 2023. Bridging the data gap between children and large language models. *Trends in Cognitive Sciences*, 27(11):990–992.

Michael C. Frank, Mika Braginsky, Daniel Yurovsky, and Virginia A. Marchman. 2017. Wordbank: An open repository for developmental vocabulary data. *Journal of Child Language*, 44(3):677–694.

Hiromichi Hagihara and Kazuki Miyazawa. 2026. How well do large language models mirror human cognition of word concepts?: A comparison of psychological ratings for early-acquired English words. *Behavior Research Methods*, 58:58.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 2733–2743.

Thomas T. Hills, Mounir Maouene, Josita Maouene, Adam Sheya, and Linda B. Smith. 2009. Longitudinal analysis of early semantic networks: Preferential attachment or preferential acquisition? *Psychological Science*, 20(6):729–739.

Zachary N. Houghton, Yuhan Zhou, Devin Pluth, and Vijay K. Gurbani. 2026. The holistic storage of verb+up phrases in text-based and audio-based language models. *arXiv preprint arXiv:2606.13993*.

Zachary Nicholas Houghton. 2025. *Multi-Word Representations in Minds and Models: Investigating the Storage of Multi-Word Phrases in Humans and Large Language Models*. University of California, Davis.

Zachary Nicholas Houghton, Kenji Sagae, and Emily Morgan. 2025. The role of abstract representations and observed preferences in the ordering of binomials in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 695–702.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, 3rd edition. Lawrence Erlbaum Associates.

Byung-Doh Oh and William Schuler. 2023. Why does surprisal from larger transformer-based language models provide a poorer fit to human reading times? *Transactions of the Association for Computational Linguistics*, 11:336–350.

Lynn K. Perry, Marcus Perlman, and Gary Lupyan. 2015. Iconicity in English and Spanish and its relation to lexical category and age of acquisition. *PLOS ONE*, 10(9):e0137147.

Eva Portelance, Judith Degen, and Michael C. Frank. 2020. Predicting age of acquisition in early word learning using recurrent neural networks. In *Proceedings of the 42nd Annual Conference of the Cognitive Science Society*, pages 1057–1063.

Eva Portelance, Yuguang Duan, Michael C. Frank, and Gary Lupyan. 2023. Predicting age of acquisition for children's early vocabulary in five languages using language model surprisal. *Cognitive Science*, 47(9):e13334.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

David Samuel, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2023. Trained on 100 million words and still in shape: BERT meets British National Corpus. In *Findings of the Association for Computational Linguistics: EACL 2023*.

Cory Shain, Clara Meister, Tiago Pimentel, Ryan Cotterell, and Roger Levy. 2024. Large-scale evidence for logarithmic effects of word predictability on reading time. *Proceedings of the National Academy of Sciences*, 121(10):e2307876121.

Filip Smolík and Maroš Filip. 2022. Corpus-based age of word acquisition: Does it support the validity of adult age-of-acquisition ratings? *PLOS ONE*, 17(5):e0268504.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Sean Trott. 2024. Can large language models help augment English psycholinguistic datasets? *Behavior Research Methods*, 56:6082–6100.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

A Checkpoint identifiers

HuggingFace identifiers, in the order of Table 1: znhoughton/opt-babylm-125m-20eps-seed964; babylm/opt-125m-strict-2023; babylm/opt-125m-strict-small-2023; facebook/opt-125m; BabyLM-community/babylm-baseline-100m-gpt2; BabyLM-community/babylm-baseline-10m-gpt2; openai-community/gpt2; timinar/baby-llama-58m; babylm/babyllama-100m-2024; babylm/babyllama-10m-2024; babylm/roberta-base-strict-2023; FacebookAI/roberta-base; lgcharpe/ELC_BERT_baby_100M; ltg/gpt-bert-babylm-base; babylm/ltgbert-100m-2024; BabyLM-community/babylm-baseline-100m-gpt-bert-mixed; BabyLM-community/babylm-baseline-100m-gpt-bert-causal-focus; BabyLM-community/babylm-baseline-100m-gpt-bert-masked-focus.

Relational Attention for Data-Efficient Language Modeling

Adrian Brasoveanu
UC Santa Cruz,
abrsvn@gmail.com

Ece Takmaz
Utrecht University
e.k.takmaz@uu.nl

Jakub Dotlačil
Utrecht University
j.dotlacil@uu.nl

Abstract

We present Relational BabyLM, a system submission to the BabyLM 2026 challenge that combines two cognitively motivated inductive biases in a single decoder-only Transformer. Architecturally, we replace standard self-attention with a Dual Attention Transformer (DAT), which separates the routing of object-level ("sensory") lexical features from structural/relational information (Altabaa and Lafferty, 2025; Altabaa et al., 2024; Webb et al., 2024; Kerg et al., 2022; Webb et al., 2021). Relational attention (RA) disentangled from self-attention greatly increases data efficiency and out-of-training-sample generalization on purely relational tasks, but language modeling requires object-level and relational information to be integrated as well as disentangled, and RA-based LMs have remained largely unexplored. BabyLM's data-constrained training and comprehensive evaluation is an ideal testing ground for whether that data efficiency transfers. As a training intervention, we add a Next-Latent Prediction (NextLat; Teoh et al. 2026) objective that encourages hidden states to compress history incrementally into a dense belief state. Architecture is the dominant factor for structural linguistic generalization; the objective is secondary but still significant. DAT's three relational attention types (full RA vs. the simpler RCA and DisRCA variants) are largely interchangeable at 10M words; full RA pulls ahead at 100M. We also introduce a novel symbol-retrieval mechanism (RoPE-based, as opposed to learned, relative symbols) that matches learned symbol libraries while adding no parameters. On the strict (100M-word) track, our best model ranks 6th of 55 overall and 3rd of 55 on the leaderboard's NLP-task subset at the time of writing; our two strongest models outperform the GPT-2 baseline on most benchmarks, with one attaining the highest EWoK score among strict-track entries.

1 Introduction

Standard Transformer language models (Vaswani et al., 2017; Radford et al., 2019) face two structural limitations in the low-data BabyLM regime. First, ordinary self-attention entangles relational information with high-dimensional object-level features: the query–key dot product is derived from the inputs / sources / objects, and can be understood as encoding the relation between the objects and controlling which source / object is selected (via the softmax) based on that relational information, but the routed value is a feature vector derived from that same source / object. This entanglement is at odds with the division of labor in, for example, formal semantics between lexical semantics, which makes word-specific, object-like contributions to sentential meaning (truth conditions), and compositional semantics, which makes structure-based, relation-like contributions to meaning (Montague, 1973; Dowty et al., 1981; Partee, 1995).

The ability to separate relational structures from sensory features is evident in infants as young as seven months, who can readily generalize algebraic rules (relational patterns: ABA vs. ABB) to novel vocabularies (objects) (Marcus et al., 1999). Infants generalize this form of relational (symbolic) reasoning quickly, from few 'training' examples, and apply it systematically to novel out-of-training-sample (OOTS) objects. In contrast, self-attention-based transformers require a lot more data, and still struggle with OOTS generalization (Webb et al., 2024; Kerg et al., 2022; Webb et al., 2021).

A second issue with self-attention is that, unlike RNNs, it allows a model to look back at any past token, so the architecture lacks an inherent pressure to compress sequence history into a dense, unified summary that can support OOTS generalization. This contrasts with human language processing, which is subject to a "Now-or-Never" bottleneck (Christiansen and Chater, 2016): linguistic input

Proceedings of the Second BabyLM Workshop, pages 260–278
October 29, 2026 ©2026 Association for Computational Linguistics

must be rapidly chunked and compressed into hierarchical representations before fleeting sensory memory decays. This forces eager incremental processing that drives expectations in rational parsing (Hale, 2011) and cue-based memory retrieval (Lewis and Vasishth, 2005).

In this paper, we present Relational BabyLM, a system submission to the BabyLM 2026 challenge (Choshen et al., 2026) that combines two cognitively motivated inductive biases in a single decoder-only Transformer. The primary architectural contribution is (our own reimplementation of) the Dual Attention Transformer (DAT; Altabaa and Lafferty, 2025), which augments sensory self-attention with a parallel *relational attention* mechanism: source selection works as usual, but what is retrieved is an explicit pairwise relation vector together with an abstract symbol identifying the source (Section 2.1). DAT's data-efficiency advantages have been largely established on purely relational tasks, and the evidence for its usefulness in language modeling is limited to modest perplexity gains (Altabaa and Lafferty, 2025). Whether relational attention helps a language model under a broad linguistic evaluation has remained an open question investigated by our present submission.

The complementary training intervention is Next-Latent Prediction (NextLat; Teoh et al., 2026), which trains a small auxiliary dynamics model to predict the next hidden state from the current one and the actual next token (RNN-style), pushing the hidden states toward a *belief state*, a sufficient statistic of the past for predicting the future similar to dynamic semantics info states (Kamp, 1981; Heim, 1982; Groenendijk and Stokhof, 1991). The dynamics model is discarded after training, leaving the base architecture and autoregressive inference unmodified (Section 2.4).

We submit models to the strict (100M-word) and strict-small (10M-word) tracks, and run a controlled experimental matrix on the strict-small track to separate the two inductive biases. The evaluation reveals an asymmetry between them.

1. **Architecture is the dominant factor for structural generalization** (Sections 4.2 and 4.6). DAT configurations outperform matched standard Transformers on BLiMP, with the largest gains on structural domains such as island effects and subject–verb agreement, though a head-ratio sweep shows that enlarging the relational stream, rather than merely having one, does not help further.
2. **The training objective is a secondary but significant factor for cognitive alignment** (Section 4.3). NextLat models explain more variance in human reading measures than next-token-prediction models, and improve 5 of 7 fine-tuned (Super)GLUE task accuracies.
3. **The symbol-retrieval mechanism matters** (Section 4.4). In a fully powered five-seed comparison across seven mechanisms, relational-symbolic variants underperform and the rest are indistinguishable, so our new RoPE-based relative symbols match a learned relative symbol library at no parameter cost.

On the strict (100M-word) track, our two strongest models rank among the top ten entries on the official leaderboard at the time of writing, with our best model 6th of 55 overall and 3rd of 55 on the NLP-task subset, and both outperform the official GPT-2 baseline on most benchmarks; on the strict-small (10M-word) track, our NextLat submission ranks 8th of 127 on BLiMP and 6th on EWoK (Section 4.1).

2 Model

The Dual Attention Transformer (DAT) (Altabaa and Lafferty, 2025) separates two kinds of information that are entangled in ordinary self-attention. Standard self-attention routes *sensory information*: it selects source objects by comparing queries and keys, then routes their feature vectors. DAT adds *relational attention*, whose routed values explicitly encode relations between a receiving object and source objects. The resulting layer contains both sensory heads and relational heads, allowing the model to retrieve object-level features and relation-level information in parallel.

2.1 Sensory and Relational Attention

Let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times d}$ be a sequence of object representations, where n is the sequence length and d is the model (embedding) dimension, so that each $x_i \in \mathbb{R}^d$. Standard sensory attention from receiver x_i to context $\mathbf{x}$ is

$$\mathrm{Attn}(x_i, \mathbf{x}) = \textstyle\sum_{j=1}^{n} \alpha_{ij}\, x_j W_v, \tag{1}$$

$$\alpha_i = \mathrm{Softmax}\big(\langle x_i W_q,\, x_j W_k \rangle\big)_{j=1}^{n}, \tag{2}$$

where W_q, W_k, W_v are learned query, key, and value projection matrices and α_{ij} is the j-th component of α_i. The attention weights α_{ij} encode a

selection criterion, but the retrieved values $x_j W_v$ remain sensory representations of source objects. Relational attention keeps the selection mechanism but changes what is retrieved. Instead of routing only $x_j W_v$, the receiver retrieves a relation vector between x_i and x_j, tagged by a symbol s_{ij} identifying the source:

$$\mathrm{RelAttn}(x_i, \mathbf{x}) = \sum_{j=1}^{n} \alpha_{ij} \big(r(x_i, x_j)\, W_r + s_{ij}\, W_s \big), \tag{3}$$

where the pairwise relation vector $r(x_i, x_j) \in \mathbb{R}^{d_r}$ stacks d_r learned comparison channels,

$$r(x_i, x_j) = \sigma_{\mathrm{rel}}\Big(\big(\langle x_i W_q^{\mathrm{rel},\ell},\, x_j W_k^{\mathrm{rel},\ell} \rangle \big)_{\ell=1}^{d_r} \Big), \tag{4}$$

each channel ℓ using its own relational projections $W_q^{\mathrm{rel},\ell}, W_k^{\mathrm{rel},\ell}$, and σ_{rel} being a relation activation function (e.g., identity, sigmoid, tanh, or softmax). The symbol $s_{ij} \in \mathbb{R}^{d_s}$ identifies the source as an abstract symbol (Section 2.2). The projections $W_r \in \mathbb{R}^{d_r \times d_v}$ and $W_s \in \mathbb{R}^{d_s \times d_v}$ map the relation and symbol into a common output space $\mathbb{R}^{d_v}$, so their contributions can be summed. Thus, the "message" from source j to receiver i contains a source-receiver relation $r(x_i, x_j)$ and a symbol s_{ij} tagging the sender.

In principle, Transformers with only self-attention or only relational attention could be used for language modeling. The Dual Attention Transformer combines the two. For n_h^{sa} sensory heads and n_h^{ra} relational heads, the combined output for position i concatenates the two groups and applies an output projection. The SA/RA head split controls the allocation to sensory versus relational retrieval: a 6SA/6RA split gives half to each stream, while 9SA/3RA gives three quarters to sensory attention. We instantiate DAT with pre-normalized causal decoder blocks, applying the same decoder mask to sensory and relational heads before the softmax over source positions.

2.2 Symbol Assignment

Relational attention tags each source j with a symbol that is receiver-conditioned, written s_{ij}. Some of the symbol "libraries" and associated retrieval mechanisms can depend on the source alone ($s_{ij} = s_j$), as with a symbol tied to the absolute position j, whereas other symbol libraries make it depend on both, for example, relative position symbols for the receiver–source offset $j - i$. The original DAT design (Altabaa and Lafferty, 2025) offers three mechanisms—symbols indexed by absolute position, symbols indexed by the receiver–source offset, and a learned symbol library retrieved via symbolic attention (or a relational-symbolic variant)—and uses symbolic attention for its language models. We add a fourth, RoPE-based relative symbols: in the original work (Su et al., 2023) rotary encodings act on the queries and keys as a positional encoding, whereas here they generate the relative symbol (the "value") itself. The symbol mechanisms we study in a controlled five-seed comparison are:

- **Learned relative symbols** (`relative`): a learned symbol library indexed by the relative offset $j - i$, clipped to a maximum distance Δ. Each source is tagged by a receiver-conditioned relative symbol.
- **RoPE-based relative symbols** (`relative_rope`, new): instead of a learned offset-indexed library, the relative symbol at offset $j - i$ is generated by applying rotary position encoding (RoPE) to the clipped relative distance. This gives a deterministic relative symbol that does not require learning a separate symbol library.
- **Learned positional symbols** (`positional`): a learned symbol library indexed by absolute source position j, so the symbol s_j depends only on the source position in the sequence.
- **Sinusoidal positional symbols** (`positional_sinusoidal`): the same as learned positional symbols, but the position-indexed symbol library is replaced by a fixed sinusoidal positional encoding, as in one of the Abstractor experiments (Altabaa et al., 2024).
- **Symbolic attention** (`symbolic`): maps each object to a convex combination of abstract symbols from a learned library $\mathcal{S} = (s_1, \ldots, s_{n_s})$ via learned feature templates.
- **Relational-symbolic** (`relsymbolic(_n4)`): first computes a local relation profile for each position over a causal neighborhood of size k (default 2, or 4), then applies symbolic attention to that profile.

2.3 Rel. Attn. Type: RA, DisRCA, and RCA

We compare three relational attention types, which differ in (i) whether the attention weight itself serves as the relation, and (ii) what is routed. **RCA**

(Altabaa et al., 2024) is self-attention where the values are replaced by symbols: $x'_i = \sum_j \alpha_{ij}\, s_{ij}$, with $\alpha_{ij} = \sigma_{\text{rel}}(\langle x_i W_q, x_j W_k \rangle)$. The attention weight itself is the relation, and the routed values are symbols rather than sensory features. The Abstractor treats σ_{rel} as a configurable hyperparameter, softmax in most of its experiments but element-wise activations in others, since softmax can mask relevant information. Our runs use identity (architecture comparison, head-ratio sweep, submitted models) or sigmoid (symbol-retrieval comparison; Appendix B). **DisRCA** (Altabaa and Lafferty, 2025) is an intermediate design between RCA and RA. A softmax attention weight α_{ij} selects sources, which are gated by a separately computed per-head relational compatibility r_{ij}—the construction of Eq. 3 taken as one scalar channel per relational head rather than as a d_r-dimensional vector: $x'_i = \sum_j \alpha_{ij}\, r_{ij}\, s_{ij}$. The elementwise product creates an AND-like condition—a source is routed only if it is both selected and relationally compatible—and only the symbol is routed. Full **RA** (Altabaa and Lafferty 2025; Section 2.1 above) separates selection from relation. Two independent sets of query/key maps compute (i) a softmax attention weight α_{ij} for selecting sources and (ii) a separate relation vector $r(x_i, x_j)$. The output routes both: $x'_i = \sum_j \alpha_{ij} \big(r(x_i, x_j)\, W_r + s_{ij}\, W_s\big)$.

2.4 Next-Latent Prediction and Belief States

Self-attention's unrestricted look-back gives a Transformer no inherent pressure to compress sequence history into a dense, unified summary like RNNs do. To add this pressure with the goal of OOTS generalization, we augment the training objective with Next-Latent Prediction (NextLat; Teoh et al., 2026). Let h_t be the final-layer hidden state at position t and $p_\theta(\cdot \mid h_t)$ the LM head that maps it to a distribution over the vocabulary. A small latent dynamics model predicts the next hidden state from the current one and the next token, $\hat{h}_{t+1} = h_t + \delta_\psi([x_{t+1}; h_t])$, where the update δ_ψ is a LayerNorm over $[x_{t+1}; h_t]$ followed by a three-layer GELU MLP of width $2d$; the skip in the equation is the only residual connection. Dropout (0.1, as elsewhere in the model) is applied to h_t entering δ_ψ, but not to the skip path. Alongside the usual next-token cross-entropy $\mathcal{L}_{\text{NTP}}$, two auxiliary losses compare that prediction against the true h_{t+1}: a Smooth L1 loss $\mathcal{L}_h$ on the hidden state itself, and a token-space divergence $\mathcal{L}_{\text{KL}} = D_{\text{KL}}\big(p_\theta^{\text{sg}}(\cdot \mid \text{sg}[h_{t+1}]) \,\|\, p_\theta^{\text{sg}}(\cdot \mid \hat{h}_{t+1})\big)$. Targets are detached ($\text{sg}[\cdot]$) to prevent representational collapse, and the LM head is frozen inside the KL term, so the only remaining gradient path runs through the predicted state $\hat{h}_{t+1}$, shaping the backbone's representations rather than the LM head. The total objective is

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \lambda_h\, \mathcal{L}_h + \lambda_{\text{KL}}\, \mathcal{L}_{\text{KL}}, \tag{5}$$

with $\lambda_h = 1.0$ and $\lambda_{\text{KL}} = 0.5$ throughout; an auxiliary cross-entropy on the predicted state's emissions is available but carries weight zero in every reported run. We use the one-step setting, which is what the belief-state result requires; the original NextLat paper additionally studies multi-step rollouts in service of speculative decoding. Unlike the token-level losses, $\mathcal{L}_h$ applies at every position whose prediction does not cross a document boundary, so belief-state structure is shaped even while processing context. These losses encourage the hidden states to converge toward a *belief state*, a sufficient statistic of the past for predicting the future. The auxiliary dynamics model is discarded after training, preserving the unmodified base architecture and standard autoregressive inference.

3 Pretraining and Evaluation

3.1 Challenge Setting and Corpora

We train on the BabyLM strict-small and strict tracks, which provide 10 million words and 100 million words respectively. Both tracks consist of developmentally plausible English text drawn from six sources: CHILDES, OpenSubtitles, Simple Wikipedia, Gutenberg, BNC spoken, and Switchboard (Choshen et al., 2026; Warstadt et al., 2023). The strict-small regime is the most data-limited BabyLM setting and is therefore the most diagnostic of inductive-bias effects.

We tokenize with byte-pair-encoding (BPE) tokenizers in the GPT-BERT recipe (Charpentier and Samuel, 2024), each with a vocabulary of 16,384. The strict-small track and the 18-layer strict model use the tokenizers distributed with the official BabyLM GPT-BERT baselines (the 10M-word and 100M-word releases, respectively); the 16-layer wide (1024 hidden dim) strict model uses a 2026 retraining of the 100M-word tokenizer on the 2026 strict corpus. Training sequences are packed to a length of 512 tokens (264 for the symbol-retrieval comparison; Appendix B). To preserve the short-utterance structure of child-directed input, we segment sequences at end-of-sequence (EOS)

boundaries: tokens after an EOS boundary cannot attend to tokens before it, and the auxiliary dynamics model is never penalized for predicting latent states across EOS: causal next-token prediction (NTP) training with utterance-level segmentation.

3.2 Model Configurations and Pretraining

All strict-small models share a 12-layer, 768-dimensional decoder-only backbone with a BPE vocabulary of 16,384, rotary positional encodings, pre-LayerNorm blocks, and GELU or SwiGLU feed-forward activations depending on configuration. The standard Transformer baseline uses 12 self-attention heads; DAT replaces these with n_h^{sa} sensory and n_h^{ra} relational heads, with the split controlling the sensory/relational allocation. LM-head tying varies independently of the objective: NextLat runs use an untied head, matching the source NextLat setup, while NTP runs appear with both tied and untied heads. Tied models have ≈123.3M parameters and untied models ≈135.9M, the untied head adding ≈12.6M parameters (the auxiliary NextLat dynamics model adds ≈5.9M parameters and is discarded after training). All models train for 10 epochs with gradient clipping at 1.0. Batch sizes, learning rates, and head splits vary across the three analysis families and are summarized in Appendix B. The strict-track submissions use the Muon/LambW recipe of Section 3.3, and their full specifications are in Appendix C.

3.3 Optimization

Our experiments use two optimization recipes. The strict-small analysis families use AdamW (Loshchilov and Hutter, 2019) with no weight decay, β_1=0.9, β_2=0.999, and the family-specific learning rates of Appendix B. The strict-track submissions use Muon (Jordan et al., 2024; Liu et al., 2025) for matrix-valued hidden-layer parameters and LambW (our own decoupled-weight-decay variant of LAMB (You et al., 2020), which relates to LAMB as AdamW relates to Adam) for embeddings, the LM head, symbol tables, normalization parameters, and biases. Muon applies a Newton–Schulz approximation to orthogonalize momentum updates; we use a Muon learning rate of 0.02, momentum of 0.95, and 5 Newton–Schulz steps. LambW retains LAMB's layerwise trust ratio while applying weight decay independently of the adaptive update, following the decoupling principle of AdamW (Kingma and Ba, 2015; Loshchilov and Hutter, 2019). We use a LambW learning rate of 10^{-3} and weight decay of 0.01. We compare the two recipes head-to-head in one controlled strict-track experiment, with AdamW at a learning rate of 5×10^{-4} (Table 7, Appendix E); because it jointly changes optimizer, learning rate, and weight decay, the comparison reflects the whole recipe rather than the optimizer alone.

3.4 Experimental Comparisons

Our experimental matrix comprises 140 training/evaluation rows across 47 experiment groups, organized into three analysis families: (i) an architecture comparison (standard Transformer vs. DAT) with 77 rows, (ii) a complete RCA/SwiGLU symbol-retrieval comparison at 264-token context window with 35 rows (7 symbol conditions × 5 seeds), and (iii) an RCA SA/RA head-ratio sweep with 28 rows (7 ratios at 512-token context with 3 seeds, plus one 264-token run per ratio). All models use the strict-small (10M-word) track. SwiGLU was chosen early in an unreported comparison with GELU, and was held fixed for all the submitted models, as well as the symbol-retrieval and head-ratio comparison families; the architecture comparison includes both GELU and SwiGLU runs, with activation partly coupled with symbol mechanism.

The architecture comparison varies four factors in a partially crossed design: **Base architecture**—standard self-attention with 12 heads vs. DAT with 9 sensory + 3 relational, or 6 + 6 heads; **Training objective**—next-token prediction (NTP) vs. Next-Latent Prediction (NextLat); **LM head**—tied vs. untied; and **Relational attention type**—full relational attention (RA), relational cross-attention (RCA), or disentangled RCA (DisRCA).

3.5 Evaluation

We evaluate on the BabyLM zero-shot benchmark suite, which probes distinct linguistic and cognitive dimensions without task-specific fine-tuning: grammatical phenomena (**BLiMP**, **BLiMP supplement**, Warstadt et al. 2020), conceptual property knowledge (**COMPS**, Misra et al. 2023), discourse tracking and world knowledge (**Entity Tracking**, Kim and Schuster 2023, **EWoK**, Ivanova et al. 2025, **GlobalPIQA**, Chang et al. 2025), and human-likeness measures (fit to human reading-time and EEG data, and age of acquisition). We also evaluate selected models on the **(Super)GLUE** fine-tuning suite. Detailed benchmark definitions are provided in Appendix D.

Our claims rest on two statistical models, each fit on the strict-small (10M-word) track.

Binomial GLMMs on BLiMP. We fit binomial generalized linear mixed-effects models (GLMMs) on BLiMP pseudo-items (successes out of items per pseudo-item). A baseline-vs-DAT model compares the standard Transformer against the 9SA/3RA RCA/SwiGLU DAT configuration, with fixed effects for base architecture, LM-head tying, and training objective; an internal-DAT model adds fixed effects for relational attention type, head split, feed-forward activation and symbol-position setting, and context length. All models use Type III Wald χ^2 tests with sum-to-zero contrast coding, and crossed random intercepts for experiment (which identifies the seed and configuration), BLiMP subtest, and pseudo-item ID. Post-hoc contrasts use Tukey-adjusted pairwise comparisons. Throughout, the architecture comparison's factors are only partially crossed (tied/NextLat is unobserved), so contrasts involving a missing cell are model-based extrapolations.

Reading regression. A cross-experiment linear mixed-effects model regresses the incremental R^2 (ΔR^2) gained by adding model surprisal to baseline reading-time predictors, with crossed random intercepts for experiment and reading measure, and fixed effects for training objective, model type, LM-head tying, context length, and reading condition. The last of these is an evaluation-condition control: each experiment contributes two ΔR^2 rows per measure, one from the base reading regression and one from a spillover regression that adds previous-word predictors.

4 Experiments

4.1 BabyLM Submission

Table 1 summarizes our strict-track models and the official GPT-2 baseline under the official leaderboard evaluation. Our two strongest entries are DAT models trained with NextLat, the Muon/LambW recipe, and RoPE-based relative symbols, each on a two-phase curriculum that starts with line (utterance) EOS segmentation and switches to document EOS boundaries: an 18-layer model (768-dim, 9SA/3RA, RCA) trained with a curriculum of 6 line-EOS / 4 document-EOS epochs, and a 16-layer wide model (1,024-dim, 12SA/4RA, RA) trained with a curriculum of 2 line-EOS / 8 document-EOS epochs. Both submitted checkpoints are exponential moving averages (EMA) of the training weights; Appendix C gives their full specifications. At the time of writing our two strongest entries rank 6th and 7th of 55 overall; our best, the 16-layer wide (1024-dimensional) model, also ranks 3rd of 55 on the NLP-task subset, which comprises all tasks except age of acquisition and fit to reading data. The wide model beats the baseline on eight of the nine benchmarks (all but GlobalPIQA) and holds the highest EWoK score on the strict-track leaderboard at the time of writing (59.54); the 18-layer model beats the baseline on seven of nine, and shows no reliable age-of-acquisition correlation (0.00; non-significant correlations are scored as zero, Appendix D) whereas the baseline's correlation is significantly negative (-11.58). Its curriculum trades a little BLiMP accuracy for a large gain on the BLiMP supplement ($+8.5$ points over the non-curriculum 18L model). Note that these are not controlled parameter or training-recipe matched comparisons: the GPT-2 baseline has 98M parameters trained with AdamW, while our entries range from 123M to 304M parameters trained with Muon/LambW. A parameter-matched architecture comparison is provided in Section 4.2 below with models trained on the strict-small track.

On the strict-small (10M-word) track, we submit two 12-layer DAT models with the same 9SA/3RA RCA architecture, SwiGLU activations, RoPE-based relative symbols, and Muon/LambW recipe: a base NTP model with a tied head (123M parameters) and a NextLat model with an untied head (136M parameters), both trained for 10 epochs at 512-token context with seed 1. At the time of writing they rank 60th and 53rd of 127 entries by Overall Average (37.91 and 38.16, versus 37.38 for the GPT-2 baseline), but place near the top on the structural and knowledge-oriented benchmarks: the NextLat model ranks 8th of 127 on BLiMP (72.34, versus 65.23 for the baseline) and 6th on EWoK (52.93). Relative to the base model, NextLat improves BLiMP ($+1.9$), EWoK ($+2.7$), reading (7.12 vs. 6.56), and (Super)GLUE (63.97 vs. 62.96); the two models differ in LM-head tying and therefore parameter count. Table 6 in Appendix E reports the full strict-small results.

4.2 Does Dual Attention Help?

For structural syntactic reasoning (BLiMP), architecture choice is the dominant factor. A binomial GLMM with fixed effects for base architecture

Model	BLiMP	Supp.	EWoK	Ent. trk.	COMPS	PIQA	GLUE*	Reading	AoA	Overall	NLP
GPT-2 baseline (12L)	74.73	65.00	54.37	16.91	55.85	36.62	67.75	6.93	−11.58	40.73	53.03
DAT 12L (NTP)	79.81	60.03	56.66	20.84	57.84	**37.65**	65.92	5.76	−12.02	41.39	54.11
DAT 18L (NextLat)	**80.62**	62.01	56.99	**20.93**	58.36	34.72	68.90	6.49	0.00	43.23	54.65
DAT 18L (NextLat, curric.)	78.94	70.51	57.25	20.47	58.69	32.72	69.95	6.22	0.00	43.86	55.50
DAT 16L wide (NextLat, curric.)	79.49	**70.96**	**59.54**	20.89	**59.22**	36.17	**71.43**	**7.10**	−9.48	**43.92**	**56.81**

Table 1: Strict-track (100M-word) results from the official BabyLM leaderboard at the time of writing (55 entries). All DAT models use relative symbol retrieval with RoPE-based relative symbols (no learned symbol library; Section 2.2), SwiGLU activations, and the Muon/LambW recipe: the 12L model has 123M parameters (9SA/3RA, RCA, 768-dim, tied head, seed 0), the 18L models have 191M parameters (9SA/3RA, RCA, 768-dim, seed 1), and the 16L wide model has 304M parameters (12SA/4RA, RA with four relation channels, 1,024-dim, seed 1). Full specifications of the two top entries are in Table 4. PIQA is GlobalPIQA; GLUE* is the leaderboard's (Super)GLUE fine-tuning average; Reading is the leaderboard's composite human-likeness score, which aggregates eleven eye-tracking, self-paced-reading, and ERP measures (Appendix D); AoA is the age-of-acquisition score (Appendix D); Overall is the leaderboard Overall Average (equal weight across the nine benchmarks); NLP is the Overall Average restricted to the seven NLP-task benchmarks, excluding the two human-likeness measures (reading and age of acquisition, which form the leaderboard's Human-like Average). The GPT-2 baseline is the official challenge baseline (98M parameters, AdamW). Bold marks the best value per column. The comparison with the GPT-2 baseline is not parameter or training-recipe matched.

(standard Transformer vs. DAT), LM-head tying, and training objective reveals a large and significant main effect of architecture ($\chi^2(1) = 139.43$, $p < 0.001$; Table 11, Appendix E). Estimated contrasts show that DAT scores higher than the standard Transformer baseline across all four head-tying $\times$ objective conditions, with advantages ranging from $+0.083$ to $+0.288$ on the log-odds scale (all $p < 0.001$; Table 2). Note that contrasts at factor combinations not present in the training matrix (such as tied/NextLat) are model-based extrapolations (Section 3.5). The best DAT configuration reaches 70.66% BLiMP against 68.16% for the best standard Transformer (Table 10, Appendix E).

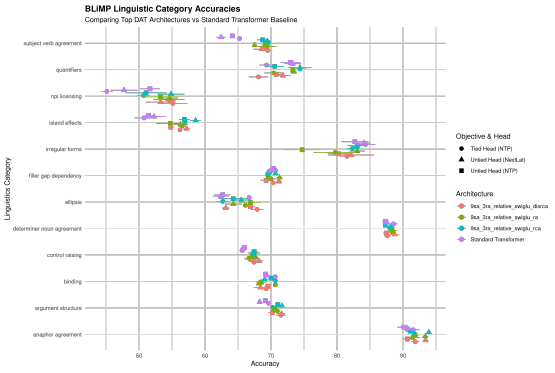


Figure 1: BLiMP accuracy by linguistic category, comparing DAT and standard Transformer configurations. Gains are largest on structural domains (e.g., island effects, subject–verb agreement).

The advantage is not uniform across linguistic phenomena. A focused architecture $\times$ linguistic-domain interaction model shows a significant interaction ($\chi^2(12) = 573.07$, $p < 0.001$): DAT yields larger gains on structural domains such as island effects, subject–verb agreement, and quantifiers than on lexical or semantic ones. Figure 1 shows BLiMP accuracy by linguistic category for DAT and standard Transformer configurations; Figure 2 in Appendix E gives mean BLiMP accuracy across configurations.

4.3 Does NextLat Complement Dual Attention?

While architecture governs structural accuracy, the training objective governs alignment with human cognitive processing. A cross-experiment reading regression models the incremental R^2 (ΔR^2) gained by adding model surprisal to baseline reading-time predictors. The training objective is significant ($t = 5.22$, $p < 0.001$): NextLat models explain more variance in human reading measures than NTP models (Table 9, Appendix E). Model type (DAT vs. standard) is also significant ($t = -2.61$, $p = 0.010$), with standard Transform-

Contrast	LM head	Objective	Estimate	SE	z	p
baseline − DAT	tied	NextLat	−0.288	0.038	−7.49	< 0.001
baseline − DAT	untied	NextLat	−0.186	0.022	−8.37	< 0.001
baseline − DAT	tied	NTP	−0.185	0.022	−8.33	< 0.001
baseline − DAT	untied	NTP	−0.083	0.022	−3.72	< 0.001

Table 2: Estimated contrasts (baseline − DAT) from the BLiMP GLMM, by LM-head tying and training objective. Negative estimates indicate that DAT scores higher than the standard Transformer baseline. All four contrasts are significant at $p < 0.001$ (Tukey-adjusted).

ers explaining slightly more reading-time variance than DAT models, and context length is significant ($t = -7.08$, $p < 0.001$). The matrix separates the objective from LM-head tying: its untied/NTP and untied/NextLat cells are matched on both, and NextLat gains 0.83 BLiMP points across them (Appendix E).

Fine-tuned task performance. On the strict-track (Super)GLUE benchmarks, NextLat improves 5 of 7 task accuracies, with the largest gains on MultiRC (+5.7 percentage points) and WSC (+3.8; Table 8, Appendix E). These two strict-track models differ in LM-head tying and parameter count ($\approx$ 12.6M).

Architecture×objective interaction. The full baseline-vs-DAT model reveals significant interactions between base architecture and both head tying ($\chi^2(1) = 10.62$, $p = 0.001$) and training objective ($\chi^2(1) = 10.81$, $p = 0.001$): the size of the DAT advantage depends on the objective and on whether the LM head is tied, so the two interventions are not simply additive.

4.4 Which Relational Mechanisms Matter?

The RCA/SwiGLU 264-token symbol-retrieval comparison is the only fully-powered complete subset in our matrix: seven symbol conditions × five seeds (Table 15, Appendix E). A binomial GLMM on BLiMP subtests shows a large, significant effect of the symbol-retrieval mechanism ($\chi^2(6) = 134.29$, $p < 0.001$).

Contrasts against the learned-relative-symbol baseline (Table 12) show that the two relational-symbolic conditions are significantly worse on BLiMP than the learned-relative baseline's 69.34%: `relsymbolic_n4`, which combines the larger size-4 relational-symbol neighborhood with a learned symbol library, at 66.32% ($p < 0.001$), and `relsymbolic` at 68.33% ($p = 0.023$). The remaining five conditions (learned-relative, RoPE-relative, learned-positional, sinusoidal-positional, and symbolic-attention) are not significantly different from one another (all pairwise $p > 0.5$, Tukey-adjusted); beyond the relational-symbolic disadvantage, the comparison does not support a finer ranking among symbol sources. That indifference is itself useful: RoPE-based relative symbols match the learned library they replace (69.43% vs. 69.34%) while eliminating its parameter cost—a learned relative-symbol library holds $(2\Delta{+}1)\,d$ parameters, 1.05M for the submitted 1,024-dimensional models with $\Delta{=}512$—and its maximum-offset table.

4.5 Which Relational Attention Type?

The controlled RA-type comparison varies only attention type, objective, and LM-head tying across 25 runs, holding architecture, symbols, context, and recipe fixed (Appendix E). On BLiMP, the three types are indistinguishable: the internal-DAT GLMM finds no significant effect of attention type ($\chi^2(2) = 3.50$, $p = 0.174$), and the largest between-type spread in any condition is 0.76 points against a pooled within-cell seed standard deviation of 0.54. RCA is numerically highest in all three conditions. This converges with the DAT paper's relational-games ablation, where RA and RCA perform similarly (Altabaa and Lafferty, 2025), and contradicts its language-modeling ablation, in which RCA-head DAT performs "no better than a standard Transformer with a matching total number of heads": ours beats exactly that baseline by 2.5 BLiMP points (Section 4.2). At 10M words the simpler RCA, which routes only symbols and needs no relation-projection parameters, is the equal of full RA.

At 100M words the ordering changes. In a matched 18-layer pair differing only in attention type, RA beats RCA on six of seven benchmarks (BLiMP 79.30 vs. 78.44, EWoK +1.7, entity tracking +3.7). RA is also the least stable: in a controlled 12-layer, 1,024-dimensional pair its training loss diverged and never recovered below its epoch-0 value, while RCA trained monotonically to 79.72 BLiMP and DisRCA to 79.38 (Appendix E). The extra capacity RA carries—separate relation projections and four relation channels—appears to need both data and a stable recipe.

4.6 SA/RA Head-Ratio Sweep

We sweep the sensory-to-relational head ratio at 512-token context, with 3 seeds per ratio (Table 18, Appendix E). A mixed-effects model with

seed as a random intercept shows that higher relational-attention (RA) fraction significantly decreases BLiMP accuracy (slope = -1.827, $t = -5.887$, $p < 0.001$) and EWoK accuracy (slope = -1.039, $p = 0.021$). With 12 heads in total the two fractions are complements: a larger relational stream is also a smaller sensory one. The RA fraction has no significant effect on COMPS ($p = 0.201$) or reading measures ($p = 0.488$), and COMPS and supplement scores stay flat across ratios, so the head-ratio allocation acts mainly on syntactic structures. Mean BLiMP peaks at the balanced 6SA/6RA split (70.33%), but 9SA/3RA — the allocation used in all submitted models — trades 0.6 BLiMP points for a 1.4-point supplement gain and the best COMPS score.

5 Related Work

BabyLM and data-efficient language models. The BabyLM challenge (Warstadt et al., 2023; Choshen et al., 2026) provides a shared evaluation for sample-efficient pretraining on developmentally plausible corpora. Successful entries have mostly intervened on the data or the training objective rather than on the attention mechanism: curricula and corpus filtering, distillation, and hybrid objectives such as GPT-BERT (Charpentier and Samuel, 2024), which merges causal and masked prediction in one Transformer stack. Architecture-level interventions like our submission evaluated under the full suite are comparatively rare.

Relational bottleneck and dot-product relations. The closest point of comparison to our work is the Abstractor architecture and relational cross-attention (RCA) (Altabaa et al., 2024) and DAT (Altabaa and Lafferty, 2025). While DAT greatly improves data efficiency on purely relational tasks, the evidence for its usefulness in language modeling is a modest perplexity gain on web text (16.94 to 16.09 at the 350M scale, and less at larger scales), with no evaluation of what the relational stream does to linguistic generalization. Our work extends the original DAT design with new symbol-retrieval mechanisms and studies them in a developmentally plausible training regime with controlled comparisons under the BabyLM evaluation suite.

The motivation for RCA and DAT connects to older work on compositionality and variable binding (Smolensky, 1990; Holyoak and Hummel, 2000). The ESBN (Webb et al., 2021) shows that a recurrent network augmented with an external memory can implement variable binding and indirection, allowing symbol-like representations to emerge and enabling near-perfect generalization of abstract rules to novel entities; CoRelNet (Kerg et al., 2022) simplifies this mechanism, showing that a matrix of dot-product similarities can itself serve as a relational representation that supports out-of-distribution generalization. A broader cognitive framework for all of these works is provided in the relational bottleneck proposal (Webb et al., 2024), which argues that architectures can benefit from restricting part of the computation to relations rather than object-specific sensory content.

Latent-state objectives and cognitive/psycholinguistic evaluation. Next-Latent Prediction (Teoh et al., 2026) was introduced and evaluated as a world-modeling objective, on world-model benchmarks and on the latent rollouts that support speculative decoding. Whether the compression pressure it imposes also makes a model more human-like is a separate question, and the one our reading regression asks (Section 4.3). It is the same pressure the Now-or-Never bottleneck attributes to human comprehension (Christiansen and Chater, 2016), and the link between a model's word-by-word predictions and human reading effort is the standard currency for testing it (Hale, 2001; Lewis and Vasishth, 2005).

6 Conclusion

We set out to ask whether the data efficiency shown by relational attention on purely relational tasks transfers to language modeling. Our answer is a qualified yes. Under the strict-small regime, architecture is the dominant factor for structural linguistic generalization, worth roughly 2.5 BLiMP points between the best configurations of each at matched depth, width, and head count; the gain comes from adding a relational stream rather than from enlarging it, and at this scale the three relational mechanisms are interchangeable. The NextLat objective is secondary but significant, and interacts with architecture: NextLat models track human reading measures more closely and improve 5 of 7 fine-tuned (Super)GLUE accuracies. Among symbol sources only the relational-symbolic variants underperform, and our novel parameter-free RoPE-based relative symbols match the learned library they replace. At 100M words full relational attention pulls ahead, and our two strongest models rank among the top strict-track entries at the time of writing.

Limitations

Statistical caveats. The binomial GLMMs model aggregated per-item counts without observation-level random effects, so overdispersion is not addressed and exact p-values may be anti-conservative.

Incomplete replication. Most experiment groups contain only 3 seeds; only the RCA/SwiGLU 264-token symbol-retrieval comparison has the full 5-seed design.

NextLat effect size. While the training objective is statistically significant for cognitive alignment ($t = 5.22$, $p < 0.001$ in the reading regression), the effect is secondary to the architecture effect on structural generalization. The architecture$\times$objective interaction ($\chi^2(1) = 10.81$, $p = 0.001$) indicates that the NextLat benefit is not uniform across architectures. The evidence for the NextLat inductive bias driving structural generalization, while statistically significant, does not have the same weight as architecture in our experiments.

Scope. All replicated results use the strict-small (10M-word) track. Our submitted strict-track entries are single-seed; the recipe-comparison models have two (NextLat) and three (NTP) seeds. At strict scale the base and NextLat runs also differ in LM-head tying and parameter count; the strict-small matrix separates the two, since its untied/NTP and untied/NextLat cells are matched on both. Seed spread is non-trivial: the Muon/LambW NTP trio spans 79.81/79.20/78.78 BLiMP (range 1.03) and 56.66/55.23/55.35 EWoK (range 1.43); the NextLat pair spans 79.75/79.03 BLiMP (range 0.72). The seed ranges exceed the NTP-vs-NextLat gaps the recipe table reports (0.06 BLiMP, 0.51 EWoK on seed 0). Entity tracking has one extreme outlier: a third NTP seed scores 33.63 against 20.84 and 20.80 for the other two. The comparison with the official GPT-2 baseline is likewise not parameter- or recipe-matched. We do not evaluate on the strict (100M-word) track with full replication, leaving open whether the DAT architecture and NextLat objective effects scale with more data. The standard Transformer baseline uses 12 self-attention heads; we do not report a parameter-matched baseline that scales head count independently of the sensory/relational split. The architecture comparison is also not matched on every other axis: the DAT and standard configurations differ in feed-forward activation, initialization, and symbol mechanism as well as in attention.

Dataset and task coverage. We evaluate on the BabyLM strict-small zero-shot suite (BLiMP, BLiMP supplement, COMPS, Entity Tracking, EWoK, and the human-likeness measures) and the strict-track (Super)GLUE fine-tuning suite. We do not report generation quality. The human-likeness measures derive from a single resource of 205 sentences (de Varda et al., 2023) covering eye-tracking, self-paced reading, and EEG responses.

Implementation faithfulness. The DAT implementation in our experiments differs from the original DAT formulation in several respects: the use of SwiGLU vs. GELU feed-forward activations, tied vs. untied LM heads, the symbol-retrieval mechanism (RoPE-based relative symbols rather than a learned symbol library), and the weight initialization scheme. The relation activation used in RCA (identity or sigmoid rather than softmax) is a hyperparameter setting the original design provides for (in principle), but has remained previously underexplored.

Optimizer-recipe comparison. The Muon/LambW recipe improves all five accuracy-based zero-shot measures (BLiMP $+1.73$pp, BLiMP supplement $+1.15$pp, EWoK $+1.77$pp, entity tracking $+1.99$pp, COMPS $+1.02$pp; Table 7), while reading-time measures decrease slightly. This comparison jointly changes optimizer family, learning rate, and weight decay, so the improvement is a property of the recipe and (potentially) not of specific components in isolation.

Acknowledgments

AB gratefully acknowledges the compute support from the National Research Platform (NRP) at the University of California, San Diego. NRP has been developed, and is supported in part, by funding from National Science Foundation, from awards 1730158, 1540112, 1541349, 1826967, 2112167, 2100237, and 2120019, as well as additional funding from community partners (Weitzel et al., 2025). ET and JD were supported by the European Research Council (ERC), grant 101088098 - MEMLANG. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of any funding agency. Part of this work also used resources available through the Dutch

national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-18356.

LLM assistants were used in circumscribed, closely supervised ways during code development and the editing process of the manuscript. On the spectrum between local, narrow autocomplete vs. global, large-scale LLM-based generation of code or text, this work stays decidedly close to the local / narrow end of the spectrum. We retain full responsibility for all content, and all errors are our own.

References

Awni Altabaa and John Lafferty. 2025. Disentangling and integrating relational and sensory information in transformer architectures. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 1271–1297. PMLR.

Awni Altabaa, Taylor Whittington Webb, Jonathan D. Cohen, and John Lafferty. 2024. Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in transformers. In *The Twelfth International Conference on Learning Representations*.

Tyler A. Chang, Catherine Arnett, Abdelrahman Sadallah, Abdelrahman Eldesokey, Abeer Kashar, Abolade Daud, Abosede Grace Olanihun, Adamu Labaran Mohammed, Adeyemi Praise, Adhikarimayum Meerajita Sharma, and 1 others. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *Preprint*, arXiv:2510.24081.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Morten H. Christiansen and Nick Chater. 2016. The now-or-never bottleneck: A fundamental constraint on language. *Behavioral and Brain Sciences*, 39:e62.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2023. Cloze probability, predictability ratings, and computational estimates for 205 English sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*, 56(5):5190–5213.

David R. Dowty, Robert E. Wall, and Stanley Peters. 1981. *Introduction to Montague Semantics*. Studies in Linguistics and Philosophy. D. Reidel Publishing Company, Dordrecht.

Jeroen Groenendijk and Martin Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy*, 14(1):39–100.

John Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

John T. Hale. 2011. What a rational parser would do. *Cognitive Science*, 35(3):399–443.

Irene Heim. 1982. *The Semantics of Definite and Indefinite Noun Phrases*. Ph.D. thesis, University of Massachusetts Amherst, Amherst, MA.

Keith J. Holyoak and John E. Hummel. 2000. The proper treatment of symbols in a connectionist architecture. In Eric Dietrich and Arthur B. Markman, editors, *Cognitive Dynamics: Conceptual and Representational Change in Humans and Machines*, pages 229–263. Lawrence Erlbaum Associates, Mahwah, NJ.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks.

Hans Kamp. 1981. A theory of truth and semantic representation. In Jeroen Groenendijk, Theo Janssen, and Martin Stokhof, editors, *Formal Methods in the Study of Language*, pages 277–322. Mathematical Centre Tracts, Amsterdam.

Giancarlo Kerg, Sarthak Mittal, David Rolnick, Yoshua Bengio, Blake Richards, and Guillaume Lajoie. 2022. On neural architecture inductive biases for relational tasks. *Preprint*, arXiv:2206.05056.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Richard L. Lewis and Shravan Vasishth. 2005. An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, 29(3):375–419.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, and 9 others. 2025. Muon is scalable for LLM training. *Preprint*, arXiv:2502.16982.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Gary F. Marcus, Sujith Vijayan, Shoba Bandi Rao, and Peter M. Vishton. 1999. Rule learning by seven-month-old infants. *Science*, 283(5398):77–80.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Richard Montague. 1973. The proper treatment of quantification in ordinary english. In *Approaches to Natural Language*, pages 221–242. D. Reidel Publishing Company, Dordrecht.

Barbara H. Partee. 1995. Lexical semantics and compositionality. In Lila R. Gleitman and Mark Liberman, editors, *An Invitation to Cognitive Science, Volume 1: Language*, 2 edition, pages 311–360. MIT Press, Cambridge, MA.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Paul Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1–2):159–216.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864. Originally published in 2021.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S. Hu, Tim Pearce, Pratyusha Sharma, Akshay Krishnamurthy, Riashat Islam, Alex Lamb, and John Langford. 2026. Next-latent prediction transformers learn compact world models. *Preprint*, arXiv:2511.05963. Version 3, May 2026; v1 posted November 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Taylor W. Webb, Steven M. Frankland, Awni Altabaa, Simon Segert, Kamesh Krishnamurthy, Declan Campbell, Jacob Russin, Tyler Giallanza, Randall O'Reilly, John Lafferty, and Jonathan D. Cohen. 2024. The relational bottleneck as an inductive bias for efficient abstraction. *Trends in Cognitive Sciences*, 28(9):829–843.

Taylor Whittington Webb, Ishan Sinha, and Jonathan D. Cohen. 2021. Emergent symbols through binding in external memory. In *The Ninth International Conference on Learning Representations*.

Derek Weitzel, Ashton Graves, Sam Albin, Huijun Zhu, Frank Wuerthwein, Mahidhar Tatineni, Dmitry Mishin, Elham Khoda, Mohammad Sada, Larry Smarr, Thomas DeFanti, and John Graham. 2025. The national research platform: Stretched, multi-tenant, scientific Kubernetes cluster. In *Practice and Experience in Advanced Research Computing 2025: The Power of Collaboration*, PEARC '25, New York, NY, USA. Association for Computing Machinery.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations*.

A Further Details

Our code—model implementation, training and evaluation scripts, configurations, and analysis code—is available at `https://github.com/abrsvn/babylm_dat_2026`. Our submitted checkpoints are also publicly available on the HuggingFace Hub and are linked from the official leaderboard entries.

	Architecture	Symbols	Head ratio
Context	512	264	512
Batch size	12 or 16	16	12
LR (AdamW)	5×10^{-5}	10^{-3}	5×10^{-5}
SA/RA split	6/6, 9/3	9/3	1/11 to 11/1
Attn type	RA, RCA, DisRCA	RCA	RCA
Rel. activation	identity	sigmoid	identity
Activation	GELU/SwiGLU	SwiGLU	SwiGLU
LM head	tied/untied	tied	tied
Seeds	3	5	3

Table 3: Training configurations of the three strict-small analysis families (Section 3.4). SA/RA = sensory/relational head split. All runs use AdamW, 10 epochs, gradient clipping at 1.0, and 12-layer, 768-dimensional backbones. The architecture comparison also spans training objective (NTP vs. NextLat) and includes single auxiliary runs at 264- and 1,280-token context (the context-length factor in the internal-DAT analysis of Appendix E); LM-head tying varies independently of the objective: NextLat runs are untied, NTP runs appear both ways.

B Strict-Small Training Configurations

Table 3 summarizes the training configurations of the three strict-small analysis families (Section 3.4), referenced from Section 3.2.

C Top Strict-Track Model Specifications

Table 4 gives the full model and training specifications of our two strongest strict-track entries (Section 4.1). Parameter counts are computed directly from the published checkpoints; the remaining settings are taken from the training configurations. Both models use the RoPE-based relative symbols introduced in Section 2.2, which add no learned parameters, and both submitted checkpoints are exponential moving averages of the training weights.

Table 5 reports the official per-task fine-tuned (Super)GLUE scores for the two top entries and the GPT-2 baseline.

D Benchmark Definitions

BLiMP. The Benchmark of Linguistic Minimal Pairs (Warstadt et al., 2020) consists of 67 minimal-pair paradigms testing grammatical phenomena such as subject–verb agreement, island effects, and filler–gap dependencies. We use aggregate BLiMP accuracy and a decomposition by linguistic category.

BLiMP supplement. The BLiMP supplement contains additional minimal-pair paradigms beyond the core 67, probing further syntactic and semantic phenomena.

COMPS. The Conceptual Minimal Pair Sentences benchmark (Misra et al., 2023) tests property attribution and property inheritance. Each pair attributes a property to two concepts, one that has it and one that does not (*A robin/#penguin can fly*); the harder subsets restate the property of a novel concept introduced as a subordinate of the positive or negative concept (*A wug is a robin/penguin. Therefore, a wug can fly*), which controls for memorization of the literal test phrase, and then add the negative concept back as a distractor.

Entity Tracking. This benchmark (Kim and Schuster, 2023) evaluates whether a model can maintain and update the changing states and locations of entities across a discourse, testing a core cognitive prerequisite for narrative comprehension.

EWoK. The Elements of World Knowledge benchmark (Ivanova et al., 2025) evaluates basic conceptual world knowledge across domains such as spatial relations, physical states, and social interactions. Each example consists of minimal-pair sentences testing whether a model understands concepts that help model the world.

Reading and eye-tracking. The BabyLM human-likeness benchmark regresses a model's per-word surprisal against eleven psycholinguistic dependent variables drawn from de Varda et al. (2023): four eye-tracking measures (first-fixation, first-pass, go-past, and right-bounded reading time), self-paced reading time, and six event-related potential components (ELAN, LAN, N400, P600, EPNP, PNP). The incremental R^2 (ΔR^2) reported in Table 9 is the additional variance in these measures explained by adding model surprisal to baseline predictors, averaged over the eleven variables.

GlobalPIQA. Global PIQA (Chang et al., 2025) is a physical commonsense-reasoning benchmark spanning over 100 language varieties, with a culture-specific non-parallel split and a shared, culture-agnostic parallel split. The Strict and Strict-Small tracks use the English subset, evaluated cloze-style: completions are chosen by length-normalized conditional log-probabilities; the leaderboard score is the mean of the parallel and non-parallel splits (Choshen et al., 2026).

Setting	18L curriculum model	16L wide model
Leaderboard name	DAT Strict Curriculum	DAT Strict NextLat Final
Parameters	191.2M	304.3M
Layers / hidden dim.	18 / 768	16 / 1,024
Attention heads (SA/RA)	9 / 3	12 / 4
Attention type	RCA	RA (4 relation channels, identity activation)
Symbol mechanism	RoPE-based relative symbols, shared across layers; no learned symbol library (parameter-free)	
Feed-forward	SwiGLU, hidden dim. 3,072	SwiGLU, hidden dim. 4,096
Normalization	pre-LayerNorm	
Positional encoding	RoPE (θ=10,000); max. relative position 512	
LM head	untied	
Vocabulary / sequence length	16,384 / 512 packed tokens (514 model positions)	
Tokenizer	GPT-BERT 100M (2024 release)	GPT-BERT 100M (2026 release)
Batch	48 sequences of 512 tokens	36 sequences of 512 tokens
Random seed	1	1
Curriculum	Phase A: 6 epochs line-EOS; Phase B: 4 epochs document-boundary	Phase A: 2 epochs line-EOS; Phase B: 8 epochs document-boundary
Muon lr (Phase A / Phase B)	0.02 / 0.008	0.02 / 0.005
LambW lr (Phase A / Phase B)	10^{-3} / 4×10^{-4}	10^{-3} / 2.5×10^{-4}
Weight decay	0.01	0.05
LR schedule	1% warmup, cosine decay to 10% of the initial rate	
NextLat	horizon 1; MSE weight 1.0, KL weight 0.5, CE weight 0.0	
EMA	decay 0.999; submitted weights are the EMA checkpoints	
Initialization / dropout	scaled-normal (0.02) projections / 0.1	

Table 4: Full specifications of our two strongest strict-track models, verified against the published checkpoints and training configurations. Both are decoder-only DAT models trained from scratch on the BabyLM strict (100M-word) corpus for 10 total epochs: Muon (momentum 0.95, 5 Newton–Schulz steps) optimizes hidden weight matrices and LambW (β_1=0.9, β_2=0.999, ϵ=10^{-8}) optimizes embeddings, heads, symbol tables, norms, and biases. Phase B is warm-started from the Phase-A EMA weights. The two models use different releases of the GPT-BERT 100M-word tokenizer, both with a vocabulary of 16,384: the 2024 release is byte-identical to the tokenizer distributed with the official GPT-BERT baseline, and the 2026 release is a retraining of the same tokenizer recipe on the 2026 strict corpus (Section 3.1). Training sequences are packed to 512 tokens as in Section 3.1; the models allocate 514 positions: the 512 packed tokens, a leading BOS slot, and one appended final-target token that the one-step NextLat rollout conditions on (nextlat-free models allocate 513). Parameter counts are checkpoint (backbone) counts; the training-time NextLat dynamics module adds a further 10.5M parameters and is discarded after training. No teacher model, synthetic data, or multimodal input is used.

Age of acquisition (AoA). Following Chang and Bergen (2022), the AoA evaluation tracks word surprisal across training checkpoints, derives a model age of acquisition for each word, and compares it with children's acquisition ages from CDI norms. The reported score is the Pearson correlation between model and child acquisition ages (scaled by 100); correlations that are not significant at $p \leq 0.1$ are scored as zero.

E Further Results

Strict-small leaderboard results. Table 6 reports the official strict-small (10M-word) leaderboard results for our two submissions and the GPT-2 baseline, complementing the strict-track results of Table 1 and discussed in Section 4.1.

Optimizer-recipe comparison. Table 7 reports the strict-track optimizer-recipe comparison between AdamW and Muon/LambW at 12 layers, referenced in Sections 3.3 and 4.1.

Strict-track fine-tuned task performance. Table 8 reports the fine-tuned (Super)GLUE comparison between the same two 12-layer Muon/LambW models as Table 7, discussed in Section 4.3.

Reading regression. Table 9 gives the fixed-effect coefficients of the cross-experiment reading regression of Sections 3.5 and 4.3.

Architecture-comparison group means. Table 10 gives the per-configuration means behind Section 4.2.

Symbol ablation at submission scale (not controlled). A strict-small run with a learned relative-symbol library in place of RoPE-based symbols scores 71.42 on BLiMP against 72.33 for the submitted model, but is 14 layers deep rather than 12 and was trained with an exponential mov-

Task	GPT-2 base	18L curric.	16L wide
BoolQ	69.66	69.48	**70.52**
MNLI	60.76	63.00	**66.26**
MRPC	85.34	88.00	**88.74**
MultiRC	65.92	65.68	**66.30**
QQP	71.56	73.14	**74.26**
RTE	57.55	66.91	**70.50**
WSC	**63.46**	**63.46**	**63.46**
(Super)GLUE avg.	67.75	69.95	**71.43**

Table 5: Official fine-tuned (Super)GLUE task scores for the GPT-2 baseline and our two top strict-track entries (Table 1), from the official leaderboard evaluation. Metrics follow the leaderboard's definitions: F1 for MRPC and QQP, accuracy for the rest—including MultiRC, which the BabyLM pipeline scores by accuracy rather than by SuperGLUE's F1a and exact match. Bold marks the best value per row.

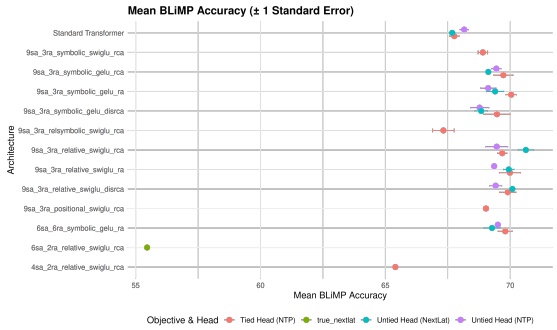


Figure 2: Mean BLiMP accuracy across model configurations. With few exceptions, DAT architectures outperform the standard Transformer baseline (top). Error bars are ± 1 standard error across seeds.

ing average, so the difference is not attributable to the symbol mechanism. A depth-matched 12-layer run completed training but was not evaluated.

Attention type at 100M words (18 layers). Two 18-layer curriculum runs differ only in attention type and in batch size (48 for RCA, 44 for RA; the RA model's relation projections raise its memory cost): same 2026 tokenizer, same phase-A switch point, same learning rates, weight decay, and epoch counts, each warm-started from its own phase A. RA scores higher on six of seven benchmarks — BLiMP 79.30 vs. 78.44, EWoK 57.93 vs. 56.20, entity tracking 22.77 vs. 19.07, COMPS 58.62 vs. 57.23, and both GlobalPIQA splits — and lower on the BLiMP supplement (68.56 vs. 69.00). Each arm is a single seed, and RA is sensitive to the surrounding hyperparameters: two further 18-layer RA runs under different learning-rate, switch-point, and epoch settings score 78.18 and 76.49, a range that spans the RCA result. All figures in this paragraph are raw-checkpoint scores, not exponential moving averages.

Wide-12L training stability (controlled for RA vs. RCA). At 1,024 dimensions and 12 layers, a pair differing only in attention type shows RA diverging during training (loss peaking at 5.25 in epoch 3, never recovering below its epoch-0 value; BLiMP 58.21, EWoK 50.15, COMPS 49.65, all at or near chance) while RCA trains monotonically (BLiMP 79.72). A DisRCA run also trains cleanly (BLiMP 79.38) but additionally differs in weight decay (0.01 vs. 0.05). The submitted 16-layer wide RA model trained normally under a two-phase curriculum with a reduced Phase-B learning rate. Per-epoch training loss: RA 3.44, 3.69, 4.55, 5.25, 5.14, 4.97, 4.49, 3.98, 3.75, 3.61; RCA 3.34, 2.81, 2.67, 2.40, 2.36, 2.33, declining monotonically; DisRCA 3.36, 2.95, 2.90, 2.87, 2.84, 2.82, 2.78, 2.74, 2.70, 2.68.

Tokenizer and curriculum observations (not controlled). Two strict-track observations from the 18-layer curriculum experiments are worth recording, though neither is a controlled comparison. First, the 2024-tokenizer run scores slightly higher than the 2026 retraining on the same 18-layer architecture (BLiMP EMA 79.17 vs. 78.44; supplement 70.45 vs. 69.00), but the runs also differ in weight decay (0.01 vs. 0.05), phase-A switch point, and Phase-B epoch count, so the tokenizer effect cannot be isolated. Second, two 2026-tokenizer RA curriculum runs with different hyperparameter settings (learning rate 4e-4 vs. 1e-3, Muon lr 0.008 vs. 0.02, phase-A switch at checkpoint 3 vs. 1, Phase-B epochs 6 vs. 8) score BLiMP 79.30 and 76.49; the lower-learning-rate, later-switch setting wins. A curriculum-versus-document-boundary pair is confounded by warm-start, learning rate, Muon lr, and epoch count (weight decay is identical).

Baseline-vs-DAT omnibus tests. Table 11 reports the Type III Wald χ^2 tests for the baseline-vs-DAT GLMM of Section 4.2, and Table 12 the full pairwise contrasts for the symbol-retrieval comparison of Section 4.4.

Internal-DAT analysis. The internal-DAT ANOVA (Table 13) decomposes BLiMP performance across the DAT design factors. The activation/symbol-position term ($\chi^2(2) = 26.41$, $p < 0.001$) covers the early GELU/symbolic runs that preceded the fixed-SwiGLU design

Model	BLiMP	Supp.	EWoK	Ent. trk.	COMPS	PIQA	GLUE*	Reading	AoA	Overall
GPT-2 baseline (12L)	65.23	57.25	50.63	**19.10**	51.81	35.09	63.80	5.63	**−12.15**	37.38
DAT 12L (NTP)	70.40	**57.94**	50.25	18.62	**52.79**	**35.64**	62.96	6.56	−14.00	37.91
DAT 12L (NextLat)	**72.34**	56.40	**52.93**	18.45	52.69	33.65	**63.97**	**7.12**	−14.15	**38.16**

Table 6: Strict-small (10M-word) results from the official BabyLM leaderboard at the time of writing (127 entries). Both DAT models use 9SA/3RA RCA, SwiGLU activations, RoPE-based relative symbols, and the Muon/LambW recipe (Muon lr 0.02, effective batch of 16 sequences of 512 tokens, seed 1, 10 epochs; the NextLat run reaches that batch as 8 sequences with 2 gradient-accumulation steps); the NTP model has a tied LM head (123M parameters) and the NextLat model an untied head (136M parameters). Column definitions as in Table 1. Bold marks the best value per column. The comparison with the GPT-2 baseline (98M parameters, AdamW) is not parameter- or recipe-matched.

Optimizer	Objective	BLiMP	Supp.	EWoK	Ent. trk.	COMPS
AdamW	NTP	78.08	58.88	54.89	18.85	56.82
AdamW	NextLat	77.95	59.92	55.07	19.63	57.69
Muon/LambW	NTP	79.81	60.03	56.66	20.84	57.84
Muon/LambW	NextLat	79.75	61.01	57.17	20.27	57.90

Table 7: Optimizer-recipe comparison on the strict track: 12-layer, 768-dimensional DAT models (9SA/3RA, SwiGLU, relative symbols), batch of 64 sequences. AdamW: lr 5×10^{-4}, no weight decay. Muon/LambW: Muon lr 0.02 (momentum 0.95, 5 Newton–Schulz steps) for hidden matrix parameters; LambW lr 10^{-3} (weight decay 0.01) for embeddings, heads, norms, and biases. All rows are seed 0. NTP models use tied LM heads; NextLat models use untied LM heads. The Muon/LambW recipe improves all five measures, but the comparison jointly changes optimizer, learning rate, and weight decay.

(Section 3.4); LM-head tying ($p = 0.006$) and context length ($p = 0.001$) are also significant. Relational attention type ($p = 0.174$) and layer allocation ($p = 0.437$) are not (Section 4.5). The training objective is also not significant within the DAT family on BLiMP ($p = 0.216$), consistent with architecture being the primary lever for structural generalization while NextLat provides a complementary cognitive-alignment benefit.

Objective at matched LM-head tying. The architecture comparison observes three of the four head-tying $\times$ objective combinations: tied/NTP, untied/NTP, and untied/NextLat. The last two are matched on head tying and on parameter count, since the auxiliary dynamics model is discarded after training, so their difference isolates the objective. Within the controlled cell of Table 14, NextLat gains 0.83 BLiMP points over NTP (RCA $+1.17$, DisRCA $+0.67$, RA $+0.58$); the internal-DAT model does not find the objective significant on BLiMP ($p = 0.216$), while it is highly significant in the reading regression ($t = 5.22$, $p < 0.001$). The first two cells isolate LM-head tying at NTP: 70.18 tied against 69.42 untied.

RA-type per-cell means. Table 14 reports BLiMP means for the controlled RA-type comparison of Section 4.5: 25 runs holding SwiGLU, learned relative symbols, 512-token context, 9SA/3RA, identity relation activation, 12 layers, 768 dimensions, and all training hyperparameters fixed. Three of the seven RA runs used a batch of 16; the other 22 used 12. Pooled within-cell seed SD is 0.54 (df=16).

Further symbol-retrieval results. Table 15 gives the per-condition means behind Section 4.4, and Table 16 the same comparison on composite reading scores.

SA/RA head-ratio sweep. Figure 3 shows mean BLiMP accuracy across SA/RA head-ratio splits, and Table 19 reports the mixed-effects model estimates for the SA/RA head-ratio sweep at 512-token context (3 seeds per ratio, seed as random intercept). The RA head fraction has a significant negative effect on BLiMP and EWoK, but no significant effect on COMPS or reading measures.

Task	Muon base	Muon NL	Δ
BoolQ	0.684	0.684	0.000
MNLI	0.596	0.594	−0.002
MRPC	0.725	0.740	+0.015
MultiRC	0.598	0.655	+0.057
QQP	0.761	0.772	+0.011
RTE	0.547	0.576	+0.029
WSC	0.654	0.692	+0.038

Table 8: Fine-tuned (Super)GLUE task accuracies: Muon/LambW base (no NextLat, tied head) vs. Muon/LambW NextLat (untied head). All seven tasks are scored by accuracy (MultiRC by per-answer accuracy, rather than SuperGLUE's F1a and exact match); the leaderboard itself scores MRPC and QQP by F1, so these values differ from the corresponding leaderboard entries for the same checkpoints. NextLat improves 5 of 7 task accuracies; a two-sided paired t-test across the seven task accuracies gives $t(6) = 2.61$, $p = 0.040$. These two strict-track models differ in LM-head tying and parameter count (≈ 12.6M); the strict-small matrix separates the objective from tying, since its untied/NTP and untied/NextLat cells are matched on both.

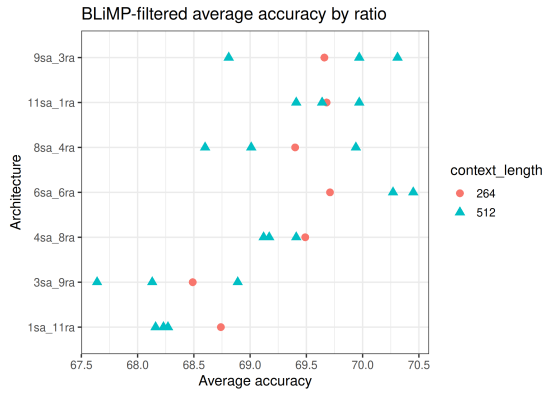


Figure 3: Mean BLiMP accuracy across SA/RA head-ratio splits. Error bars are ±1 standard error across 3 seeds.

Term	Estimate	SE	t	p
(Intercept)	0.048	0.011	4.56	0.001
model_type	−0.001	0.000	−2.61	0.010
objective	0.002	0.000	5.22	< 0.001
tie_lm_head	0.000	0.000	0.35	0.728
datapoint_length1	−0.005	0.001	−7.08	< 0.001
datapoint_length2	−0.004	0.001	−4.85	< 0.001
model_variant	−0.003	0.000	−28.48	< 0.001

Table 9: Cross-experiment reading regression: coefficients for the incremental R^2 (ΔR^2) gained by adding model surprisal to baseline reading-time predictors. The response is the per-experiment, per-dependent-variable ΔR^2. The training objective is significant ($t = 5.22$, $p < 0.001$): Next-Latent Prediction models explain more variance in human reading measures than NTP models. Model type (DAT vs. standard) is also significant ($p = 0.010$). The model-variant effect ($p < 0.001$) reflects the large difference between base and spillover reading conditions.

Architecture	Objective	LM head	N	BLiMP	Reading	Entity	COMPS
Standard Transformer	NTP	tied	3	67.76	7.06	14.27	52.10
Standard Transformer	NTP	untied	3	68.16	7.16	19.14	51.86
Standard Transformer	NextLat	untied	3	67.68	7.36	16.64	52.03
DAT (RCA, SwiGLU)	NTP	tied	3	**70.66**	6.66	19.71	52.42
DAT (RCA, SwiGLU)	NextLat	untied	3	70.62	7.27	**21.92**	52.36
DAT (RA, GELU)	NTP	tied	3	70.04	6.89	22.23	**52.51**
DAT (RA, GELU)	NextLat	untied	3	69.39	**7.37**	20.46	52.28

Table 10: Main results: BLiMP accuracy and downstream benchmarks by architecture and training objective. Standard Transformer baselines and DAT models share the same 12-layer, 768-dimensional, 12-head configuration, but also differ in feedforward activation (GELU vs. SwiGLU), initialization, and symbol mechanism. N is the number of random seeds. Reading is the composite reading score (mean of eye-tracking and self-paced reading).

Effect	χ^2	df	p
(Intercept)	39.76	1	< 0.001
base_arch	139.43	1	< 0.001
is_tied	2.64	1	0.105
objective	2.26	1	0.132
base_arch:is_tied	10.62	1	0.001
base_arch:objective	10.81	1	0.001

Table 11: Type III ANOVA (Wald χ^2 tests) for the baseline-vs-DAT BLiMP model (binomial GLMM). The model includes fixed effects for base architecture (standard Transformer vs. DAT), LM-head tying, and training objective, with crossed random intercepts for experiment, BLiMP subtest, and pseudo-item. The base-architecture effect is large and significant; the architecture also interacts significantly with both head tying and objective.

Contrast	Estimate	SE	z	p
relative − relative_rope	−0.006	0.019	−0.30	0.999
relative − positional	0.012	0.019	0.61	0.997
relative − positional_sinusoidal	0.030	0.019	1.54	0.722
relative − symbolic	0.028	0.019	1.42	0.792
relative − relsymbolic	0.062	0.019	3.21	0.023
relative − relsymbolic_n4	0.182	0.019	9.37	< 0.001

Table 12: RCA/SwiGLU symbol-retrieval contrasts from the BLiMP subtest binomial GLMM. Contrasts are relative to the `relative` (learned relative symbol) baseline; p-values are Tukey-adjusted.

Effect	χ^2	df	p
(Intercept)	56.999	1	< 0.001
attn_type	3.496	2	0.174
layers	0.604	1	0.437
activation_position	26.406	2	< 0.001
is_tied	7.564	1	0.006
objective	1.531	1	0.216
context_length	10.187	1	0.001

Table 13: Type III ANOVA (Wald χ^2 tests) for internal DAT mechanics on BLiMP (binomial GLMM). Fixed effects: relational attention type (RA/RCA/DisRCA), layer allocation, activation/position setting, LM-head tying, training objective, and context length. Activation/position, head tying, and context length are significant; relational attention type and layer allocation are not.

Condition	RA	RCA	DisRCA	spread
tied / NTP	69.99 (3)	70.66 (3)	69.90 (3)	0.76
untied / NTP	69.36 (2)	69.46 (3)	69.41 (3)	0.10
untied / NextLat	69.94 (2)	70.62 (3)	70.08 (3)	0.68
marginal	69.79 (7)	70.25 (9)	69.80 (9)	0.45

Table 14: BLiMP cell means for the controlled RA-type comparison. RCA is numerically highest in all three conditions; the omnibus GLMM finds no significant effect of attention type ($\chi^2(2) = 3.50$, $p = 0.174$).

Symbol condition	BLiMP	Entity	COMPS
relative (learned)	69.34	22.13	52.56
relative_rope	**69.43**	20.54	52.54
positional	69.17	16.69	52.56
positional_sinusoidal	68.88	**23.12**	52.14
symbolic	68.90	17.70	**52.60**
relsymbolic	68.33	21.17	52.09
relsymbolic_n4	66.32	19.88	51.63

Table 15: RCA/SwiGLU symbol-retrieval comparison, using 5 seeds on BLiMP, COMPS and Entity accuracy scores. All models use the same 9SA/3RA RCA architecture with RoPE positional encodings and SwiGLU activations. Further comparisons are in the appendix.

Symbol condition	Eye-tracking	SPR
relative (learned)	9.94	3.64
relative_rope	9.41	3.66
positional	9.78	3.81
positional_sinusoidal	9.84	3.70
symbolic	9.64	3.58
relsymbolic	9.93	3.85
relsymbolic_n4	10.06	3.90

Table 16: RCA/SwiGLU symbol-retrieval comparison (complete design: 7 symbol conditions $\times$ 5 seeds). All models use the same 9SA/3RA RCA architecture with RoPE positional encodings and SwiGLU activations. The `relative` condition is the reference baseline for contrasts. Shown measures are the eye-tracking and self-paced reading composite scores.

Effect	χ^2	df	p
(Intercept)	38.28	1	< 0.001
symbol_retrieval	134.29	6	< 0.001

Table 17: Type III ANOVA (Wald χ^2 tests) for the RCA/SwiGLU symbol-retrieval BLiMP subtest model (binomial GLMM). The symbol-retrieval mechanism has a large, significant effect on BLiMP subtest accuracy ($\chi^2(6) = 134.29, p < 0.001$).

SA	RA	BLiMP	Supp.	EWoK	COMPS
1	11	68.22	56.92	50.01	51.67
3	9	68.22	57.38	50.37	51.70
4	8	69.23	56.92	50.44	51.77†
6	6	**70.33**	56.32	50.55	51.64
8	4	69.18	57.06	50.75	51.70
9	3	69.70	**57.75**	50.60	**52.04**
11	1	69.67	56.87	**51.13**	51.81

Table 18: RCA SA/RA head-ratio sweep at 512-token context, mean of 3 seeds. All models use RCA with SwiGLU and relative symbols. The 6SA/6RA split achieves the best BLiMP accuracy (70.33%). †One 4SA/8RA COMPS evaluation is missing; its COMPS mean is over two seeds.

Metric	Estimate (Slope)	Std. Error	t	p
BLiMP	-1.827	0.310	-5.887	<0.001***
Reading	0.170	0.240	0.710	0.488
COMPS	-0.245	0.190	-1.287	0.201
EWoK	-1.039	0.447	-2.323	0.021*

Table 19: Mixed-effects models for the SA/RA head-ratio sweep at 512-token context (3 seeds per ratio, seed as random intercept). The RA head fraction has a significant negative effect on BLiMP and EWoK, but no significant effect on COMPS or reading measures. The seed random effect collapsed to zero variance in the BLiMP model.

Can Translation Bridge the Data Gap for Low-Resource Languages? A Case Study of Hindi and Telugu

Venkata Naga Sai Vishnu Rohit Pulipaka and Libby Barak
Montclair State University
New Jersey, USA
{pulipakav1,barakl}@montclair.edu

Abstract

Low-resource languages often lack sufficient training data for developing small language models, limiting the application of recent language acquisition benchmarks beyond English. Moreover, comparison across languages is made harder given differences in training data composition and quality. We investigate whether machine translation can be used to create BabyLM-scale training corpora for low-resource languages by translating the English BabyLM corpus into Hindi and Telugu using IndicTrans2. We train GPT-BERT and GPT-Wee models and evaluate them using perplexity and linguistic benchmarks. GPT-BERT consistently outperforms GPT-Wee across both monolingual and bilingual settings. Although perplexity is lower on in-domain test data, this pattern is consistent across English and the translated languages, suggesting that the effect reflects domain similarity rather than artifacts introduced by translation. Furthermore, the models achieve competitive performance on linguistic evaluations relative to pretrained language models. These findings suggest that translation warrants further analysis to obtain data for developing and evaluating small language models in low-resource settings.

1 Introduction

The study of language acquisition in India has drawn attention from researchers across disciplines such as psychology and education, but has been less explored from a computational perspective. Indeed, the linguistic diversity of native speakers in India presents a complex challenge, with hundreds of languages and thousands of dialects spanning five language families (Mohanty, 2006). These languages also differ substantially in their available linguistic and computational resources, and many are considered low-resource from a computational standpoint. For example, Hindi has considerably larger resources compared to Telugu, though both are official state languages in certain regions of India (Mohanty, 2010; Philip et al., 2021). Interestingly, English remains a constant across this landscape, given its widespread use in educational settings (Mohanty, 2006).

Recently, there has been growing interest in computational modeling for Indian languages, including the development of Large Language Models (LLMs) optimized for these language families (e.g., IndicBERT, MuRIL) (Dabre et al., 2022; Khanuja et al., 2021), and workshops dedicated to the development of models and resources for these languages (Jha et al., 2024). While most work on small language models has focused on English (see (Charpentier et al., 2025; Hu et al., 2024)), several studies have focused on languages other than English, including Chinese, Japanese, Slovak, and Hebrew (Gao et al., 2025; Kriš and Šuppa, 2025; Gelboim and Sulem, 2025).

Choshen et al. (2026) added non-English datasets to the BabyLM challenge for Chinese and Dutch. These datasets rely on the availability of sufficient data in these two languages to create a complementary resource to the English data, i.e., 100M words spanning across data from diverse genres. We ask whether training data can be created artificially for low-resource languages directly from the English data.

We use IndicTrans2 (Gala et al., 2023) to translate the 100M English data provided by the BabyLM workshop into Hindi and Telugu (Choshen et al., 2026). As a preliminary analysis, we focus on Hindi and Telugu out of the many Indic languages. Hindi, an Indo-Aryan language written in Devanagari, is one of the most widely spoken languages in India, while Telugu, a Dravidian language with a distinct grammatical structure and writing system, provides a complementary typological perspective and remains significantly underresourced in computational studies. Although we do not use additional resources to construct the

Proceedings of the Second BabyLM Workshop, pages 279–288
October 29, 2026 ©2026 Association for Computational Linguistics

translated data itself, the low-resource status of Telugu relative to Hindi may still affect translation quality, which we account for in our evaluation. A native speaker of Hindi and Telugu evaluated a sample of the translated sentences and verified translation quality.

We train both monolingual and bilingual models for each language and each language-English pairing. The use of data in three languages that stems from the same semantic and pragmatic coverage allows us for a more direct comparison of the results across all models. We evaluate the models on their ability to represent the translated language, the original data in each language, and the developmentally grounded measures provided by BabyLM (Choshen et al., 2026). Our results show that the translation results in similar byte ratio to a small parallel corpus data in line with suggested guidelines for data creation (Choshen et al., 2026; Arnett et al., 2024). GPT-BERT achieves superior performance to GPT-Wee for both monolingual and bilingual models on perplexity and linguistic performance measures. Across all three languages, perplexity was consistently lower on the BabyLM test data drawn from the same distribution as the training corpus compared with independent evaluation data. Because this pattern was observed for English as well as the translated Hindi and Telugu corpora, it suggests that the difference is primarily driven by domain similarity rather than noise introduced by the translation process. Despite being trained on translated corpora, the models performed well on the linguistic evaluations compared with pre-trained larger language models.

The consistency of these results across the three languages suggests that the LLM-based translation might provide a sufficiently stable training data to support further exploration of using this method to procure limited training data for small language models. We conclude by outlining suggestions for further work to validate this approach across additional languages and training methods.

2 Related Work

Language acquisition in India presents a complex challenge in its diverse native languages and differences across language exposure. In addition to the numerous official languages that stem from several language families, access to English widely differs depending on social-economic status and geographical location (Parshad et al., 2016). Computational modeling can enable analysis of language acquisition through manipulation of degree of language exposure, input properties, and language combinations. While models of English language acquisition rely on data sets of child-directed speech and story books (MacWhinney, 2000; Gerlach and Font-Clos, 2020), CHILDES only covers several Indic languages and on a much smaller scale.

While most of the work on small language models focused on models of English, several studies explored the modeling non-English learning, developing data sets for other languages, and creating standard evaluation measures (Choshen et al., 2026). One approach to data creation has been to collect data from similar sources across languages (e.g., child-directed speech, newspaper data, subtitles, etc.). Previous work suggested synthetic data generation as a way to overcome the limited data availability and evaluated the approach on English as preliminary analysis (Ulm et al., 2025; Tampier et al., 2025). Yet, synthetic data generation showed mixed results as a source for training even in high-resource languages (Poh et al., 2025; Feng et al., 2024; Haga et al., 2024). Beyond data availability, models of multilingual learners offer an additional challenge of quantifying the effect of typological differences on the training process.

In this study, we propose to use the carefully curated English BabyLM corpus as the source for generating training data in low-resource languages through machine translation, following recent advances in multilingual translation models(Gala et al., 2023; Goyal et al., 2022). Using a shared source corpus provides comparable semantic and pragmatic coverage across languages, facilitating controlled comparisons between monolingual and bilingual models, similar to multilingual evaluation benchmarks (Liang et al., 2023; Jumelet et al., 2025). This approach offers the advantage of consistent thematic and register coverage across all datasets given the shared source, while reducing the need to locate diverse content types in multiple languages independently. However, translation may introduce noise and systematic biases while reducing language-specific cultural and topical variation (Artetxe et al., 2020; Bender et al., 2021), and translation quality may vary across target languages, potentially placing lower-resource languages such as Telugu at a disadvantage relative to Hindi. Our evaluation framework provides an initial approach to evaluating these trade-offs.

3 Methods

We use the English 100M words data (BabyLM-community/BabyLM-2026-Strict) provided by the BabyLM workshop (Choshen et al., 2026). We translate each of the sub-sections of the data: CHILDES (MacWhinney, 2000), British National Corpus (BNC) (BNC Consortium, 2007), dialogue portion, Project Gutenberg (children's stories) (Gerlach and Font-Clos, 2020), OpenSubtitles (Lison and Tiedemann, 2016), Simple English Wikipedia (Wikimedia, 2023), and Switchboard Dialog Act Corpus (Stolcke et al., 2000). We translate the data to Hindi and Telugu, independently, using IndicTrans2 (Gala et al., 2023) (the en-indic-1B checkpoint), with source language code eng_Latn and target language codes hin_Deva and tel_Telu respectively. Sentences are pre- and post-processed using the IndicTrans Toolkit's IndicProcessor. We translate with a batch size of 64 for Hindi and 32 for Telugu, beam search with beam size 5, and a maximum sequence length of 256 tokens. Since the training data is a translation of the original data release, the models in all languages theoretically represent equivalent semantic input that varies in other linguistic properties such as syntactic and morphological structure. We note that the resulting Hindi translation contains 118,309,059 words and 1,381,881,024 bytes, a word-to-byte ratio of 1:11.68. The Telugu translation contains 79,179,423 words and 1,471,312,814 bytes, a word-to-byte ratio of 1:18.58. For comparison, a parallel Hindi and Telugu corpus extracted from OpenSubtitles (Lison and Tiedemann, 2016) had word-to-byte ratios of 1:11.97 and 1:17.18, respectively.

We train monolingual and bilingual models using the code provided by the BabyLM workshop (Choshen et al., 2026), including GPT-Wee (based on GPT-2) and GPT-BERT (Bunzeck and Zarrieß, 2023; Charpentier and Samuel, 2024a). The monolingual models are trained on data from a single language. The English model is trained in the full 100M word data, and the Hindi and Telugu models are trained on the full translation. To analyze the effect of language learning setting on the resulting model, we create bilingual models for each language pair: we combine translated data from CHILDES and Gutenberg with the original English data for the remaining subsets (BNC, OpenSubtitles, Simple Wikipedia, and Switchboard). This selection is intended to approximate naturalistic Indic-English bilingual exposure, in which the native/Indic language predominates in the home setting (CHILDES for child-directed language and Gutenberg for book reading) and English predominates in more formal or educational contexts (e.g., Wikipedia) (Mohanty, 2006).

We train each GPT-Wee model 2 times for each setup with different seeds, for a total of 10 models. The models are trained for 10 epochs with a learning rate of 5e-5 and a warmup ratio of 0.01. Batch size is 4 for monolingual runs and 16 for bilingual runs. Total training steps are determined by dataset size, and no fixed gradient accumulation.

We train each model 2 times for each setup with different seeds, for a total of 10 GPT-BERT models. The parameters for training are a batch size of 32768 (global; 128 per GPU), max steps of 15625, 250 warmup steps, and a learning rate of 1e-2. Gradient accumulation is dynamic rather than fixed. Validation is every 1000 steps for bilingual runs, while monolingual runs disable validation.

We evaluate the models with two complementary goals: (1) their ability to model natural language as reflected in corpus data and (2) their linguistic knowledge. For language modeling, we measure perplexity on two evaluation sets. The first is the BabyLM test set, which follows the same data distribution as the training corpus and therefore provides an in-domain evaluation of how well the models learn each language despite their typological differences. The second consists of parallel sentence pairs extracted from OpenSubtitles that do not overlap with any of the BabyLM subtitle data. Although this dataset is drawn from the same subtitle domain as 31% of the BabyLM training data, it represents an independent sample with no overlap with BabyLM data and therefore evaluates the models' ability to generalize beyond the specific benchmark data. Moreover, the Hindi and Telugu models trained and tested on BabyLM data are both using the translation method, while the OpenSubtitles data measures their ability to generalize to data that is not obtained through the same translation methodology. Using parallel data ensures that all languages are evaluated on equivalent content, allowing differences in perplexity to reflect language modeling performance rather than differences in sentence meaning. Together, these two evaluations assess both in-domain language modeling and generalization to unseen subtitle data.

In addition, we analyze the linguistic abilities of the models using the measures made available by BabyLM that include benchmark for either Hindi,

Telugu or both. As such, we evaluate the monolingual and bilingual models using MultiBLiMP (Hindi only), which measures grammatical and syntactic knowledge through minimal sentence pairs; SIB200 (Hindi and Telugu), which evaluates document classification and semantic understanding across languages; and MuBench (Hindi and Telugu), which assesses multilingual language understanding on a range of downstream NLP tasks.

Our training and evaluation are primarily based on the BabyLM code and datasets. The additional code used for translation[1], and the translated data[2] are publicly available in the corresponding repositories.

4 Results

We evaluate the monolingual and bilingual language models using complementary intrinsic and extrinsic benchmarks designed to assess both language modeling performance and cross-lingual generalization. We first present perplexity results over all models and data sources, followed by an evaluation on linguistic abilities using three benchmarks following the BabyLM evaluation guidelines.

4.1 Perplexity

We evaluate monolingual language models trained on BabyLM data translated into Hindi and Telugu using IndicTrans2, comparing two model architectures: GPT-Wee and GPT-BERT (Bunzeck and Zarrieß, 2023; Charpentier and Samuel, 2024a). We evaluate on two data sources: (1) held-out BabyLM test data translated into the target language using the same IndicTrans2 procedure as the training data, providing an in-distribution evaluation; and (2) OpenSubtitles (OS), a naturally occurring parallel corpus used to assess generalization beyond the training distribution. To assess the value of translated child-directed speech relative to native-language data, we additionally train GPT-BERT models on non-translated Hindi and Telugu data drawn from CC-100 (Wenzek et al., 2020), a web crawl corpus. For each language, we sample CC-100 data to match the byte size of the corresponding translated training corpus, ensuring a controlled comparison between translated and non-translated conditions.

Table 1 presents perplexity results across all models and evaluation sets. Importantly, all models are trained and tested on parallel data making the comparison more streamlined for semantic content. GPT-BERT outperforms GPT-Wee across all three languages on both evaluation sets, consistent with previous finding (Charpentier and Samuel, 2024b). For GPT-Wee, Hindi perplexity is lower than Telugu on both sets, while GPT-BERT shows the reverse pattern, achieving lower perplexity for Telugu than Hindi, though on a smaller scale. This crossover suggests that the two architectures interact differently with the typological properties of each language, or with the translation quality of each language pair, and warrants further investigation.

The Hindi and Telugu GPT-BERT models achieve markedly low perplexity scores (4.60 and 3.68 on BabyLM test data, respectively), substantially lower than the English GPT-BERT model (119.98). While this may partly reflect that the translated test data shares structural and lexical patterns with the training data more tightly than the English test set does, the magnitude of the difference suggests that tokenization may also play a role: subword segmentation for morphologically rich languages like Hindi and Telugu can produce longer token sequences that reduce per-token uncertainty independently of linguistic knowledge. We return to this point in the Discussion and flag it as a confound in interpreting absolute perplexity values across languages.

The increase in perplexity from BabyLM test data to the OpenSubtitles corpus is substantially larger for GPT-Wee in the translated languages, and particularly dramatic for Telugu (134.25 to 1343.92), indicating that the smaller model learns representations that do not generalize beyond the training distribution. GPT-BERT shows more stable generalization: for English, it achieves lower perplexity on OpenSubtitles than on the BabyLM test set (114.69 vs. 119.98). Hindi and Telugu GPT-BERT models show a larger increase from test to OS-data than English, but remain far more stable than GPT-Wee, suggesting that the translated corpora provide a reasonably stable training signal despite originating from machine translation.

Comparing translated and non-translated training conditions, the translated GPT-BERT models outperform their non-translated counterparts on BabyLM test data, as expected given the in-distribution nature of that evaluation. For both

[1] https://github.com/NLPlabMSU/babylm-translation-scripts

[2] https://osf.io/973gp/?view_only=b255c4a0243341f498b6b243ddf528f6

Model	Test	OS-data
EN-GPT-Wee	140.5	292.35
HI-GPT-Wee	85.41	707.13
TE-GPT-Wee	134.25	1343.92
EN-GPT-BERT	119.98	114.69
HI-GPT-BERT	4.6	20.79
TE-GPT-BERT	3.68	17.62
NT-HI-GPT-BERT	11.17	8.36
NT-TE-GPT-BERT	8.53	12.13

Table 1: Results for the models trained on a single language (showing the mean across runs), GPT-Wee using BabyLM data (top panel), GPT-BERT using BabyLM data (middle panel), and GPT-BERT on non-translated data (bottom panel). Each column corresponds to testing on the data of the language used to train the model.

Hindi and Telugu, naturally occurring web data generalizes better to naturally occurring evaluation text. NT-HI-GPT-BERT outperforms HI-GPT-BERT on the OpenSubtitles data (8.36 vs. 20.79) and NT-TE-GPT-BERT outperforms the translated model on OpenSubtitles (12.13 vs. 17.62). This pattern may reflect translation quality, typological distance from English, or the adequacy of the CC-100 data as a representative corpus for each language, and represents an important direction for future work. Nevertheless, the difference in out-of-domain generalization between translated and non-translated models is modest, particularly for Telugu, suggesting that translated child-directed speech remains a competitive training source even beyond its native distribution — an encouraging result given the significantly lower cost of data creation via translation compared with sourcing naturally occurring corpora for low-resource languages.

Overall, these results suggest that IndicTrans2-based translation of child-directed speech offers a promising path toward cognitively plausible training data for low-resource languages, with translated data providing advantages in Hindi that do not fully extend to Telugu, underscoring the importance of evaluating this approach across multiple language pairs before drawing general conclusions.

We next train bilingual models on either English and Hindi or English and Telugu. Since English is commonly used in education settings in India, we use the data from CHILDES and Gutenberg in the translated form, and the original English for the other sources, which is roughly an even split between the two languages for each model. Table 2 presents the perplexity results for the bilingual models with the addition of two multilingual models: Llama 3.2 1B, and Sarvam-2b (Dubey et al., 2024; Sarvam AI, 2024). The pre-trained models provide a reference point for model performance using large data, which is non-translated and covers a wider linguistic scope, and training protocols that have been optimized for Indic languages. We run the same evaluation methods using the pre-trained models to allow more direct comparison.

The bilingual experiments further demonstrate the utility of translated data for multilingual language modeling. Because the bilingual models were trained on a fixed input size and similar semantic scope, the observed performance can be more directly interpreted as model sensitivity to language composition rather than properties of the input in each language. GPT-BERT maintained low perplexity on both English and the translated language, while exhibiting remarkably similar behavior for English–Hindi and English–Telugu training. Despite the substantially different resource availability of Hindi and Telugu, the bilingual models achieved comparable performance, suggesting that the translated corpora provide a stable basis for cross-lingual learning.

Though the translated data offers similar semantic content across all languages and models, we recognize that the results from monolingual and bilingual models may be influenced by the tokenizer training. We return to this issue in the Discussion when interpreting the cross-lingual results and their implications.

Finally, compared with larger multilingual models, the bilingual models show a mixed pattern. Llama 3.2 1B achieves lower perplexity on English, consistent with the benefits of large-scale pretraining. However, the gap is smaller for Hindi and Telugu, where the bilingual GPT-BERT models achieve comparable or lower perplexity than Sarvam-2B despite being substantially smaller. These results suggest that carefully constructed translated training data can enable small models to learn effective representations for targeted languages, although larger models maintain an advantage in broader language modeling settings.

4.2 Linguistic Knowledge

We next evaluate whether the representations learned from translated training data extend beyond perplexity-based evaluation. Specifically, we consider the multilingual benchmarks available

Model	English		Hindi		Telugu	
	Test	OS-data	Test	OS-data	Test	OS-data
EN-HI-GPT-Wee	233.69	448.11	93.05	879.17	–	–
EN-TE-GPT-Wee	199.26	365.27	–	–	270.56	3937.21
EN-HI-GPT-BERT	82.55	188.63	8.37	30.67	–	–
EN-TE-GPT-BERT	84.49	187.90	–	–	4.25	19.10
Llama 3.2 1B	60.45	46.98	24.97	21.07	3.70	4.41
Sarvam-2B	186.58	76.14	210.51	229.16	205.81	2350.40

Table 2: Results for the models trained on two languages, and general-purpose baselines (showing the mean across runs). Each column group corresponds to testing on English, Hindi, or Telugu data, regardless of which language pair the model was trained on.

Model	English			Hindi			Telugu	
	MBLIMP	SIB200	MuBench	MBLIMP	SIB200	MuBench	SIB200	MuBench
EN-GPT-Wee	0.7469	0.3284	0.2762	-	-	-	-	-
HI-GPT-Wee	-	-	-	0.9261	0.2353	0.2750	-	-
TE-GPT-Wee	-	-	-	-	-	-	0.2108	0.2832
EN-GPT-BERT	0.7978	0.2451	0.2823	-	-	-	-	-
HI-GPT-BERT	-	-	-	0.9399	0.1961	0.2809	-	-
TE-GPT-BERT	-	-	-	-	-	-	0.1471	0.2822
EN-HI-GPT-Wee	0.7070	0.2647	0.2827	0.9150	0.1324	0.2729	-	-
EN-TE-GPT-Wee	0.7136	0.2794	0.2803	-	-	-	0.2304	0.2765
EN-HI-GPT-BERT	0.7868	0.2500	0.2768	0.9136	0.2304	0.282	-	-
EN-TE-GPT-BERT	0.7991	0.2500	0.2709	-	-	-	0.1961	0.2813
Sarvam-2B	0.8006	0.4706	0.2833	0.9772	0.4510	0.2854	0.3824	0.2768
Llama 3.2 1B	0.8188	0.6275	0.2786	0.9772	0.6078	0.2796	0.43	0.277

Table 3: Linguistic knowledge results (BLiMP/MultiBLiMP, SIB-200, MuBench) across English, Hindi, and Telugu (showing the mean across runs).

for Hindi and Telugu: MultiBLiMP (Hindi only), SIB-200 (Hindi and Telugu), and MuBench (Hindi and Telugu).As for the bilingual models, we run the same evaluation methods using the pre-trained models to allow more direct comparison.

Table 3 presents the results for all monolingual and bilingual models across the three languages. Note that each model is only evaluated on languages included in its training data, e.g., the bilingual English-Hindi model is not tested in Telugu. In addition, Telugu is not tested on MultiBLiMP since the benchmark does not include this language. Unlike the results for perplexity measures, the two models, GPT-Wee and GPT-BERT, score mostly similarly across the languages and measures without one model being consistently superior to the other across the board.

Across linguistic evaluations, the effect of model scale depends strongly on the type of capability being measured. Small models trained on translated BabyLM-scale corpora achieve strong performance on MultiBLiMP, indicating that translated data can support learning of syntactic patterns in both Hindi and English. Unlike the perplexity results, Telugu performance on SIB-200 is considerably lower, suggesting that lower language modeling perplexity does not necessarily translate into stronger syntactic generalization. The larger multilingual models substantially outperform small models on SIB-200, suggesting that broader semantic classification benefits from additional parameters and training resources. Interestingly, the gap between the small models and larger multilingual models is considerably smaller for Telugu than for English and Hindi, suggesting that model scale interacts differently with language-specific factors, including training data availability and typological differences. In contrast to MultiBLiMP and SIB-200, MuBench showed little variation across models regardless of architecture, language, or model scale. This may suggest that the benchmark is less sensitive to differences introduced by translated training data or multilingual modeling in our study. While this uniformity could reflect a genuine insensitivity of the benchmark to differences in training data composition, it may alternatively indicate that all models — regardless of size or training language — converge on a similar performance floor for the task

types included in MuBench, which would itself be a finding worth investigating in future work.

Overall, the bilingual models show only modest degradation relative to monolingual models, indicating that multilingual training with translated corpora preserves linguistic competence while enabling cross-lingual modeling. Some English–Hindi and English–Telugu combinations outperform their monolingual counterparts and others showing slight degradations. The absence of a consistent trend suggests that the interaction between the paired languages, rather than the translated data itself, plays a larger role in shaping the performance.

5 Discussion

This work explores the use of LLM-based translation to overcome limitations in training data availability for low-resource language contexts. Using the English BabyLM corpus as the source for all translated datasets provides consistent semantic content and comparable distributions across data types and modalities, enabling controlled cross-lingual comparisons. At the same time, this design may bias the learned representations toward English and reduce language-specific cultural and linguistic variation. Furthermore, translation quality may differ across target languages, potentially placing lower-resource languages such as Telugu at a disadvantage relative to Hindi. Nevertheless, the potential benefits of generating training data for languages with limited existing resources motivate this approach. Our evaluation framework is designed to account for these trade-offs by examining language modeling, linguistic competence, and multilingual transfer across both monolingual and bilingual settings.

Overall, our results suggest that the translated Hindi and Telugu corpora preserve sufficient linguistic information to support the training of small language models. Both GPT-Wee and GPT-BERT successfully learned the translated corpora, with GPT-BERT consistently achieving lower perplexity and stronger linguistic performance across monolingual and bilingual settings. While perplexity was substantially lower on the held-out BabyLM test sets than on independent evaluation data, this pattern was observed consistently across English, Hindi, and Telugu, suggesting that it primarily reflects domain similarity rather than artifacts introduced by translation.

One result that warrants further attention is the crossover between GPT-Wee and GPT-BERT in their relative performance on Hindi versus Telugu. GPT-Wee achieves lower perplexity on Hindi than Telugu, while GPT-BERT shows the reverse pattern. We hypothesize that this interaction reflects a combination of translation quality differences. Telugu, being lower-resource may result in noisier translations that benefit more from the greater capacity of GPT-BERT and tokenization effects, whereby the more agglutinative morphology of Telugu produces longer subword sequences that interact differently with each model's training objective. Disentangling these factors would require controlled experiments varying translation quality and tokenizer design independently, which we leave for future work.

The downstream evaluations reveal a more nuanced picture. Models trained on translated corpora achieved strong performance on MultiBLiMP, indicating that translated data preserve the syntactic regularities required for grammatical generalization. In contrast, larger multilingual models maintained a substantial advantage on SIB-200, particularly for English and Hindi. Interestingly, the performance gap was considerably smaller for Telugu. This pattern suggests that balanced language representation during training may be particularly beneficial for modeling low-resource languages, although further work is needed to disentangle the effects of language balance, model scale, and training data composition.

This work also opens several directions for future research. While the translated corpora provide a controlled framework for comparing language learning across English, Hindi, and Telugu, the approach should be validated on a broader range of evaluation data, including additional domains, registers, and English varieties spoken by Hindi speakers. Such evaluations would provide a more comprehensive assessment of out-of-domain generalization and help disentangle the effects of translation quality, domain similarity, and language contact. Future work should also examine the role of tokenization, as byte-pair encoding (BPE) may be less suitable for morphologically rich or agglutinative languages, potentially affecting both language modeling and downstream performance.

An important consideration when training models on translated data is the possible presence of systematic linguistic properties that distinguish translated language from naturally produced language also known as translationese. Previous work

has demonstrated that translationese can be readily identified and can introduce systematic differences in lexical, syntactic, and stylistic properties (Kunz, 2026; Vanmassenhove et al., 2021; Rabinovich et al., 2016; Rabinovich and Wintner, 2015; Koppel and Ordan, 2011). While our evaluation mostly focused on linguistic abilities, which are of interest in the context of language acquisition and learning, further work will evaluate the use of translation as training data as reflected in more culture-dependent benchmarks and downstream tasks.

While additional evaluation is needed to validate this approach across languages and benchmarks, the translated BabyLM framework also provides a controlled setting for investigating more nuanced questions of bilingual language acquisition. Unlike naturally occurring corpora, the translated data allow systematic manipulation of the linguistic environment, including the proportion of English and translated input (e.g., child-directed vs. educational content in each language), the degree of code-switching, and the balance of language exposure during training. Such controlled rearing conditions make it possible to examine how different patterns of bilingual experience influence language learning and to compare the resulting developmental trajectories with those observed in native and heritage bilingual speakers. More broadly, this work demonstrates that machine-translated developmental corpora can serve not only as a practical resource for low-resource language modeling, but also as an experimental framework for studying multilingual language acquisition under carefully controlled conditions.

Limitations

We acknowledge several limitations of our study. First, due to computational resource constraints, we trained and evaluated a limited number of model configurations. Additional runs would allow for a more robust estimation of variance and a better understanding of the stability of the observed differences across languages and model architectures. Similarly, exploring a broader range of model choice for translation and target languages would provide a more complete picture of how translated corpora influence multilingual learning. Additionally, our evaluation is limited by the availability of linguistic benchmarks for the target languages. While MultiBLiMP provides a valuable measure of syntactic knowledge, it is not currently available for Telugu.

Although translation provides a controlled approach for constructing comparable training corpora across languages, it may also introduce biases toward the source language and reduce naturally occurring linguistic and cultural variation. Future work should examine additional translation methods, native-language resources, and alternative evaluation settings to better characterize the trade-offs between controlled multilingual training data and naturally occurring language input.

Finally, as noted in Section 4, subword tokenization using byte-pair encoding (BPE) may disproportionately affect perplexity comparisons across languages. Because BPE segments morphologically rich languages such as Hindi and Telugu into more subword units per word than English, the effective prediction task differs in granularity across languages, potentially lowering per-token perplexity independently of linguistic knowledge. Models such as Sarvam-2B are trained with tokenizers specifically optimized for Indic scripts, which may account for some of the differences observed in cross-model comparisons. For this reason, perplexity values should be interpreted within the same language and model family rather than across them, and future work should evaluate tokenization-controlled perplexity measures or character-level evaluation alternatives.

References

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

BNC Consortium. 2007. The british national corpus, xml edition.

Bastian Bunzeck and Sina Zarrieß. 2023. GPT-wee: How small can a small language model really get?

In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 35–46, Singapore. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, and 1 others. 2025. Babylm turns 3: Call for papers for the 2025 babylm workshop. *arXiv preprint arXiv:2502.10645*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024a. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024b. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *arXiv preprint arXiv:2602.20092*.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M Khapra, and Pratyush Kumar. 2022. Indicbart: A pre-trained model for indic natural language generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1849–1863.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Steven Y. Feng, Noah D. Goodman, and Michael C. Frank. 2024. Is child-directed speech effective training data for language models? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22055–22071, Miami, Florida, USA. Association for Computational Linguistics.

Jay Gala, Pranjal A Chitale, A K Raghavan, Varun Gumma, Sumanth Doddapaneni, Aswanth Kumar M, Janki Atul Nawale, Anupama Sujatha, Ratish Puduppully, Vivek Raghavan, Pratyush Kumar, Mitesh M Khapra, Raj Dabre, and Anoop Kunchukuttan. 2023. Indictrans2: Towards high-quality and accessible machine translation models for all 22 scheduled indian languages. *Transactions on Machine Learning Research*.

Yuan Gao, Suchir Salhan, Andrew Caines, Paula Buttery, and Weiwei Sun. 2025. Bliss: Evaluating bilingual learner competence in second language small language models. In *Proceedings of the First BabyLM Workshop*, pages 160–174.

Anita Gelboim and Elior Sulem. 2025. Tafberta: Learning grammatical rules from small-scale language acquisition data in hebrew. In *Proceedings of the First BabyLM Workshop*, pages 76–90.

Martin Gerlach and Francesc Font-Clos. 2020. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1):126.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM challenge: Exploring the effect of variation sets on language model training efficiency. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 252–261, Miami, FL, USA. Association for Computational Linguistics.

Michael Y Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21.

Girish Nath Jha, Sobha L., Kalika Bali, and Atul Kr. Ojha, editors. 2024. *Proceedings of the 7th Workshop on Indian Language Data: Resources and Evaluation*. ELRA and ICCL, Torino, Italia.

Jaap Jumelet, Leonie Weissweiler, and Arianna Bisazza. 2025. Multiblimp 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Preprint*, arXiv:2504.02768.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, and 1 others. 2021. Muril: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

Moshe Koppel and Noam Ordan. 2011. Translationese and its dialects. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 1318–1326.

L'uboš Kriš and Marek Šuppa. 2025. Slovakbabylm: Replication of the babylm and sample-efficient pretraining for a low-resource language. In *Proceedings of the First BabyLM Workshop*, pages 301–312.

Jenny Kunz. 2026. Preferences for idiomatic language are acquired slowly — and forgotten quickly: A case study on swedish. *Transactions of the Association for Computational Linguistics*, 14:1266–1285.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. 2023. XLM-V: Overcoming the vocabulary bottleneck in multilingual masked language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152, Singapore. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Brian MacWhinney. 2000. *The CHILDES project: The database*, volume 2. Psychology Press.

Ajit K Mohanty. 2006. Multilingualism of the unequals and predicaments of education in india: Mother tongue or other tongue. *Imagining multilingual schools*, 262:283.

Ajit K Mohanty. 2010. Languages, inequality and marginalization: implications of the double divide in indian multilingualism. *International Journal of the Sociology of Language*, 2010(205).

Rana D Parshad, Suman Bhowmick, Vineeta Chand, Nitu Kumari, and Neha Sinha. 2016. What is india speaking? exploring the "hinglish" invasion. *Physica A: Statistical Mechanics and its Applications*, 449:375–389.

Jerin Philip, Shashank Siripragada, Vinay P Namboodiri, and CV Jawahar. 2021. Revisiting low resource status of indian languages in machine translation. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, pages 178–187.

Whitney Poh, Michael Tombolini, and Libby Barak. 2025. What did you say? generating child-directed speech questions to train llms. In *Proceedings of the First BabyLM Workshop*, pages 237–245.

Ella Rabinovich, Sergiu Nisioi, Noam Ordan, and Shuly Wintner. 2016. On the similarities between native, non-native and translated texts. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1881.

Ella Rabinovich and Shuly Wintner. 2015. Unsupervised identification of translationese. *Transactions of the Association for Computational Linguistics*, 3:419–432.

Sarvam AI. 2024. Sarvam 2b: A small language model for indian languages. https://huggingface.co/sarvamai/sarvam-2b-v0.5.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

Alexander Tampier, Lukas Thoma, Loris Schoenegger, and Benjamin Roth. 2025. Recombitext: Compositional data augmentation for enhancing llm pretraining datasets in low-resource scenarios. In *Proceedings of the First BabyLM Workshop*, pages 548–565.

Jannek Ulm, Kevin Du, and Vésteinn Snæbjarnarson. 2025. Contrastive decoding for synthetic data generation in low-resource language modeling. In *Proceedings of the First BabyLM Workshop*, pages 29–41.

Eva Vanmassenhove, Dimitar Shterionov, and Matthew Gwilliam. 2021. Machine translationese: Effects of algorithmic bias on linguistic complexity in machine translation. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main volume*, pages 2203–2213.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. Ccnet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the twelfth language resources and evaluation conference*, pages 4003–4012.

Wikimedia. 2023. Simple english wikipedia dump. https://dumps.wikimedia.org/simplewiki/20230301/. Accessed: 2023-07-31.

Causal, Hybrid, and Ternary Transformers under a Restricted Data Budget

Arnav Srivastav
University of California, Santa Cruz
Santa Cruz, CA 95064, USA
asriva31@ucsc.edu

Abstract

We evaluate five decoder-only Transformers trained in the BabyLM 2026 Strict-Small setting. In a closely matched training comparison, Causal-30M and Hybrid-30M share architecture, initialization, data, tokenization, optimization settings, seed, and training exposure. Causal-30M reaches lower held-out causal next-token loss. Their benchmark results come from different evaluation pipelines—server-held-out evaluation for Causal-30M and preliminary local evaluation for Hybrid-30M—so we report them descriptively and do not treat their benchmark differences as an objective-level ranking. A larger 64.57M-parameter hybrid configuration improves the preliminary NLP Average and substantially improves Entity Tracking, but its gains are not uniform across evaluation categories.

Separately, we compare architecture- and exposure-matched dense and fake-quantized ternary models. In this single-seed comparison, Ternary-30M averages 42.90 across the eight categories measured for both models, against 43.45 for Dense-30M, retaining 98.73% of the dense model's measured aggregate. The two models use different peak learning rates, so this result describes the tested training recipes, not the effect of ternarization alone. Overall, the experiments show benchmark-specific trade-offs across objectives, model configurations, and weight representations. The ternary experiment measures capability retention under fake quantization and does not assess packed storage or specialized low-bit inference.

1 Introduction

Progress in language modeling has been driven by scale: larger models, larger datasets, and more compute. Transformers made large-scale sequence modeling more parallelizable (Vaswani et al., 2017). Scaling studies described predictable relationships among language-modeling loss, parameter count, training data, and compute (Kaplan et al., 2020; Hoffmann et al., 2022). Many contemporary modeling choices were developed and tested with billions or trillions of training tokens. When data is scarce, however, those choices may not transfer cleanly: additional capacity may be underused, alternative objectives may redistribute the available learning signal, and low-precision weight constraints may impose a greater optimization cost.

The BabyLM Challenge is a shared setting for studying these questions under a restricted pretraining budget. Its evaluation suite spans grammatical, conceptual, downstream, and developmentally motivated behavior, picking up differences that language-modeling loss alone can miss (Warstadt et al., 2025; Hu et al., 2024; Choshen et al., 2026). Previous BabyLM systems and related work on data-efficient pretraining have improved performance through architectural changes, knowledge distillation, shorter training sequences, and modified objectives (Samuel et al., 2023; Charpentier and Samuel, 2023; Timiryasov and Tastet, 2023; Press et al., 2021; Charpentier and Samuel, 2024).

GPT-BERT is especially relevant: it combines causal and masked language modeling in a single Transformer and outperformed its causal-only and masked-only variants in the 2024 BabyLM setting (Charpentier and Samuel, 2024). Whether hybrid training helps across other small-model configurations and evaluation dimensions is not yet settled.

We study five GPT-style, decoder-only Transformers in the BabyLM 2026 Strict-Small setting. The experiments fall into two groups. The first examines pretraining objective and model configuration. Causal-30M and Hybrid-30M each have 29.22M parameters and are matched in architecture, initialization, corpus, tokenizer, context length, optimization settings, seed, and training exposure. Causal-30M uses next-token prediction; Hybrid-30M alternates between causal and bidirectional masked-language-modeling updates. This comparison evaluates the complete hybrid recipe: attention

Proceedings of the Second BabyLM Workshop, pages 289–298
October 29, 2026 ©2026 Association for Computational Linguistics

visibility, input corruption, prediction targets, and loss-bearing positions all differ, not just the loss function. We also evaluate Hybrid-65M, a larger hybrid configuration. It is wider and shallower than Hybrid-30M and uses different attention-head geometry, so it tests a different configuration rather than isolating parameter count.

The second group examines fake-quantized ternary training. Ternary networks represent forward-pass weights using three scaled levels (Li et al., 2016; Ma et al., 2024), while related BitNet work applies low-bit linear layers to Transformer language models (Wang et al., 2023). These methods are motivated by the potential to reduce storage and arithmetic costs.

We compare Dense-30M and Ternary-30M, two 29.66M-parameter causal Transformers matched in architecture, corpus, tokenizer, context length, seed, and training exposure. Ternary-30M ternarizes its attention and feed-forward projections during the forward pass but keeps full-precision latent weights for optimization. This experiment measures capability retention under a ternary constraint. It does not evaluate bit-packed storage or specialized low-bit inference.

1. In the most closely matched training comparison, Causal-30M reaches lower held-out causal next-token loss than Hybrid-30M. The models' benchmark outputs came from different evaluation pipelines, so we treat their benchmark rows as provisional observations, not a controlled ranking of objectives.

2. Hybrid-65M improves on Hybrid-30M in validation loss, preliminary NLP Average, Entity Tracking, and several other evaluations, but the gains are uneven. Hybrid-65M also differs in width, depth, and attention-head geometry. This comparison describes a larger configuration, not an isolated effect of parameter count.

3. In a single architecture- and exposure-matched comparison, Ternary-30M retains 98.73% of Dense-30M's mean across the eight categories measured for both models. The runs use different peak learning rates, so the result describes the tested dense and ternary training recipes rather than the effect of ternarization alone.

Together, these experiments show that training objective, model configuration, and weight representation lead to different trade-offs across evaluation dimensions under a restricted data budget.

2 Related Work

Data-efficient pretraining. BabyLM provides a shared setting for comparing language models trained on developmentally plausible, restricted corpora (Warstadt et al., 2025; Hu et al., 2024). Previous results show that performance under these constraints depends on architecture and training procedure as well as parameter count (Warstadt et al., 2025; Hu et al., 2024). LTG-BERT optimized the Transformer architecture and pretraining recipe for a 100M-word corpus (Samuel et al., 2023), while ELC-BERT introduced learned connections to earlier layer outputs and won the original Strict and Strict-Small tracks (Charpentier and Samuel, 2023). Baby Llama instead used knowledge distillation from an ensemble trained on the same small corpus (Timiryasov and Tastet, 2023). Our work complements these system-level contributions by using matched model pairs to examine particular training and weight-representation choices rather than optimizing a single leaderboard submission.

Causal and hybrid objectives. Causal language modeling predicts the next token from left context. BERT-style masked modeling reconstructs corrupted inputs using bidirectional context (Devlin et al., 2018). GPT-BERT showed that causal and masked language-modeling objectives can share a single Transformer stack and beat its causal-only and masked-only variants in the 2024 BabyLM setting (Charpentier and Samuel, 2024). Our causal–hybrid comparison revisits this question with models matched in architecture, parameter count, corpus, tokenizer, context length, optimization settings, seed, and training exposure. We evaluate the complete hybrid recipe, not one component of masked training in isolation. Our procedure is an independent implementation, not an exact replication of GPT-BERT, so differing outcomes suggest that hybrid performance depends on the specific recipe and model configuration.

Low-bit and ternary models. Ternary Weight Networks introduced scaled three-level weights as an alternative to full-precision and binary networks, with potential reductions in storage and arithmetic cost (Li et al., 2016). More recent work has applied low-bit training to Transformers. BitNet introduced 1-bit BitLinear projections for language

Source	Objective/scale	Dense/ternary
SimpleWiki	3,622,739	3,624,923
Gutenberg	3,424,991	3,424,824
BNC Spoken	1,299,294	1,299,971
OpenSubtitles	1,199,672	1,199,986
Switchboard	217,758	217,758
Total	9,764,454	9,767,462

Table 1: Whitespace-delimited word counts for the two corpus generations, before end-of-text markers are inserted. The objective/scale and dense–ternary groups are compared only within their own corpus generations.

models (Wang et al., 2023), while BitNet b1.58 extended this approach to ternary weights and reported performance comparable to full-precision Transformers with the same model size and training exposure (Ma et al., 2024). These studies motivate our dense–ternary comparison, but our scope is narrower: we measure capability retention under fake-quantized ternary forward passes while retaining full-precision latent weights. We do not claim the storage, latency, or energy benefits of a packed low-bit deployment.

3 Data and Tokenization

3.1 Training corpora

We train all models on restricted natural English corpora assembled from five sources distributed with BabyLM 2026: Simple English Wikipedia, Project Gutenberg, BNC Spoken, OpenSubtitles, and Switchboard (Choshen et al., 2026). We exclude CHILDES, synthetic or teacher-generated text, and all BabyLM benchmark data from pretraining. This corpus-construction choice is not a claim that child-directed speech is unimportant. The objective and scale experiments—Causal-30M, Hybrid-30M, and Hybrid-65M—use the final corpus generation. The Dense-30M and Ternary-30M comparison uses a slightly earlier generation drawn from the same source mixture. Table 1 gives the composition of both corpora.

We normalize text with Unicode NFKC, then normalize common typographic punctuation, remove control characters and source metadata, and collapse whitespace. We extract text using source-specific processing rules, merge short units into medium-length blocks, and deduplicate at the block level after normalization. Blocks are shuffled with seed 1337 and split roughly 98/1/1 into training, validation, and test sets. No normalized block appears in more than one split. Only the training split goes to the optimizer; the validation split measures pretraining loss, and the test split is held out from both training and model selection.

The final corpus has 9,799,922 whitespace-delimited words after end-of-text markers are inserted. The optimizer split gets 9,607,048 words, with 99,498 and 93,376 words set aside for validation and testing. The earlier dense–ternary corpus has 9,789,053 words total, 9,592,958 of them in the optimizer split. The two generations differ in corpus construction details, so we do not interpret performance differences between the two experimental groups as controlled effects.

3.2 Tokenization

We train byte-level variants of byte-pair encoding tokenizers from scratch on the training split of each corpus (Sennrich et al., 2015). This avoids relying on a tokenizer pretrained on additional text. In our tokenizer reports, no unknown tokens occur in the training, validation, or test splits. Both tokenizers preserve capitalization and punctuation and encode each end-of-text marker as a single token. They do not automatically insert beginning- or end-of-sequence tokens.

The objective and scale experiments use a vocabulary of 8,192 tokens. Their training split contains 14,421,277 BPE tokens, corresponding to 1.501 BPE tokens per whitespace-delimited word. The tokenizer reserves special tokens for padding, unknown inputs, end of text, masking, and sequence separation. The mask token is used during the masked-language-modeling steps of the hybrid objective.

The dense–ternary comparison uses a 16,384-token byte-level BPE tokenizer trained on its corresponding corpus generation. Its training split contains 13,336,022 BPE tokens, corresponding to 1.390 BPE tokens per whitespace-delimited word. This tokenizer reserves special tokens for padding, unknown inputs, and end of text. It does not reserve masking or separator tokens because both models in this comparison use causal language modeling.

The two tokenizer settings do not constitute a controlled vocabulary-size experiment. In addition to vocabulary size, the experimental groups differ in corpus generation, context length, dropout, batching, and model configuration. We therefore restrict causal comparisons to models that share the same corpus and tokenizer.

4 Models and Training

We train five GPT-style, single-stack Transformers with pre-LayerNorm residual blocks. Each model uses learned token and absolute positional embeddings, scaled dot-product self-attention, exact GELU activations, and feed-forward layers four times wider than the residual stream. The token embeddings and language-modeling head share their weights. Linear projections and the output head have no bias. LayerNorm has a learned scale but no bias.

We initialize embedding and linear weights from a normal distribution with standard deviation 0.02. For the output projections of the attention and feed-forward sublayers, we reduce the initialization standard deviation to $0.02/\sqrt{2L}$, where L is the number of Transformer layers. The five model configurations are listed in Table 2.

4.1 Training objectives

Causal-30M, Dense-30M, and Ternary-30M are trained with standard autoregressive language modeling. For a token sequence $x_{1:T}$, the objective is

$$\mathcal{L}_{\text{causal}} = -\frac{1}{T-1}\sum_{t=1}^{T-1}\log p(x_{t+1} \mid x_{\le t}).$$

A causal attention mask prevents tokens from attending to later positions, and the loss is averaged over all valid next-token predictions.

Hybrid-30M and Hybrid-65M split their training between causal language modeling and bidirectional masked language modeling. The assignment is deterministic. Update s uses the masked objective when

$$(7919s \bmod 10000) < 5000.$$

Over 36,648 optimizer updates, this gives 18,322 masked updates and 18,326 causal updates.

On a masked update, each token other than padding, end-of-text, and mask tokens is selected independently with probability 0.15. If no token is chosen in a sequence that has eligible tokens, one eligible position is forced so that the sequence still contributes to the loss. Of the selected tokens, 80% are replaced by the mask token, 10% by a token sampled from the non-special vocabulary, and 10% are left unchanged (Devlin et al., 2018). Attention is bidirectional on these updates, and only the selected positions contribute to the loss.

This experiment compares causal training against the complete hybrid recipe. The hybrid condition changes the attention pattern, input corruption, prediction targets, and number of loss-bearing positions; it does not just swap the loss function. Causal-30M and Hybrid-30M otherwise share the same architecture, initial parameters, corpus, tokenizer, context length, optimizer, learning rate, seed, and total exposure.

The runs share a seed but are not paired minibatch-for-minibatch. Generating masked inputs draws extra random numbers, so the sequence of sampled training windows eventually diverges. We call the comparison seed- and exposure-matched, not example-order-matched.

4.2 Fake-quantized ternary training

Dense-30M and Ternary-30M share the same architecture, corpus, tokenizer, context length, batching configuration, seed, and training exposure. Architecturally, Ternary-30M differs by applying fake-quantized ternary weights to its attention and feed-forward projections (Li et al., 2016). The runs also use different peak learning rates, as reported below.

For each output channel i, we first compute the mean absolute magnitude of its full-precision latent weights:

$$m_i = \frac{1}{N_i}\sum_j |w_{ij}|.$$

The ternarization threshold is then set to

$$\Delta_i = 0.7 m_i.$$

Weights at or below Δ_i in magnitude are mapped to zero. The remaining weights are represented by their sign and a channel-specific scale,

$$\alpha_i = \frac{\sum_j |w_{ij}| \mathbb{I}[|w_{ij}| > \Delta_i]}{\sum_j \mathbb{I}[|w_{ij}| > \Delta_i]},$$

with the denominator clamped to at least one. The forward-pass weight is

$$\widetilde{w}_{ij} = \alpha_i \operatorname{sign}(w_{ij}) \mathbb{I}[|w_{ij}| > \Delta_i],$$

which takes one of three values: $\{-\alpha_i, 0, +\alpha_i\}$. During backpropagation, a straight-through estimator passes gradients to the underlying full-precision latent weights (Bengio et al., 2013).

We apply this transformation to the combined query, key, and value projection, the attention output projection, and both feed-forward projections in

Model	Params	Layers	Width	Heads	MLP width	Context	Dropout	Objective
Causal-30M	29.22M	16	368	8	1,472	512	0.1	Causal LM
Hybrid-30M	29.22M	16	368	8	1,472	512	0.1	Causal + MLM
Hybrid-65M	64.57M	12	640	10	2,560	512	0.1	Causal + MLM
Dense-30M	29.66M	16	352	4	1,408	256	0.0	Causal LM
Ternary-30M	29.66M	16	352	4	1,408	256	0.0	Causal LM + ternary proj.

Table 2: Model configurations. Causal-30M and Hybrid-30M share the same architecture and parameter count. Dense-30M and Ternary-30M form a separate architecture-matched pair. Hybrid-65M is both wider and shallower than Hybrid-30M; it is a larger configuration, not a controlled change in parameter count alone.

every Transformer layer. Token and positional embeddings, LayerNorm parameters, activations, softmax operations, and the tied language-modeling head stay dense. Ternary-30M has 29,658,464 trainable parameters: 23,789,568 in ternarized projection matrices and 5,868,896 in dense components.

The ternary experiment measures how much model quality survives the forward-pass weight constraint. It does not assess a complete low-bit deployment. Training still uses full-precision latent weights and optimizer state, checkpoints are not bit-packed, and inference uses no specialized ternary kernels. We make no claims about deployment-level memory, latency, throughput, or energy savings.

4.3 Optimization

All models are trained with AdamW (Loshchilov and Hutter, 2019), using $\beta_1 = 0.9$, $\beta_2 = 0.95$, and gradient clipping at a global norm of 1.0. Weight decay of 0.1 applies to linear weight matrices. Embeddings, LayerNorm parameters, biases, and the tied output head do not receive weight decay.

The learning rate increases linearly during the first 2% of optimizer updates and then follows a cosine decay schedule, ending at 10% of its peak value. Training uses bfloat16 automatic mixed precision with full-precision parameters and optimizer state. We do not use gradient accumulation. Every run uses seed 1337. Table 3 summarizes the model-specific optimization and training-exposure settings.

Every 250 updates, we estimate validation loss on 20 batches sampled from the held-out validation split. Each configuration reuses the same validation windows at every evaluation. We use causal next-token loss as the common validation metric for all models; masked-language-modeling loss is excluded. This tests the same prediction task across configurations, but it is not objective-neutral: Causal-30M optimizes causal prediction at every update, while the hybrid models do so on roughly half of theirs.

4.4 Training exposure

Training examples are contiguous token windows with starting positions sampled with replacement from the optimizer split. Table 3 summarizes the resulting training exposure. Causal-30M, Hybrid-30M, and Hybrid-65M use exact exposure counts reconstructed from the sampled token windows. Dense-30M and Ternary-30M use estimates based on the corpus-level words-per-token ratio.

All runs targeted a ceiling of 100M whitespace-word exposures. The optimizer splits contain fewer than 10M words, so this budget amounts to more than ten exposure-equivalent passes. Batches are also sampled with replacement rather than drawn as ordered sweeps through the corpus. We report both word exposure and exposure-equivalent passes and do not call the runs ten literal epochs.

Except for AoA, all benchmark results use the final checkpoints at roughly 100M exposures; we do not select checkpoints by minimum validation loss. Causal-30M, Hybrid-30M, and Hybrid-65M were trained on an NVIDIA H100 NVL. The metadata for the earlier Dense-30M and Ternary-30M runs did not record the exact GPU. The two experimental generations also differ in context length, tokenizer, dropout, and batching, so we do not compare their training times or memory consumption directly.

5 Evaluation

We use the BabyLM 2026 evaluation suite (Choshen et al., 2026). BLiMP and its Supplement test linguistic knowledge through minimal-pair judgments (Warstadt et al., 2019). EWoK tests physical and social world knowledge (Ivanova et al., 2024). COMPS tests conceptual properties and property inheritance (Misra et al., 2022). Entity

Model	Peak LR	Batch	Tok./step	Steps	BPE positions	Word exposures	Equiv. passes
Causal-30M	3×10^{-4}	8	4,096	36,648	150,110,208	99,984,450	10.407
Hybrid-30M	3×10^{-4}	8	4,096	36,648	150,110,208	99,987,918	10.408
Hybrid-65M	3×10^{-4}	8	4,096	36,648	150,110,208	99,987,918	10.408
Dense-30M	3×10^{-4}	32	8,192	16,970	139,018,240	99,999,545†	10.424†
Ternary-30M	2×10^{-4}	32	8,192	16,970	139,018,240	99,999,545†	10.424†

Table 3: Optimization and training-exposure settings. Word-exposure counts for Causal-30M, Hybrid-30M, and Hybrid-65M are reconstructed exactly from the sampled token windows. Values marked with † are estimates based on the corpus-level words-per-token ratio. Dense-30M and Ternary-30M use different peak learning rates, so their comparison does not isolate weight representation alone.

Tracking evaluates whether models track changing information about entities across discourse, while GlobalPIQA tests cultural commonsense reasoning. The (Super)GLUE category summarizes seven fine-tuning tasks. Reading measures how much model surprisal improves the prediction of human reading times. Age of Acquisition (AoA) compares word-learning trajectories across training with human acquisition data.

Aggregation. Let $\mathcal{N}$ be the seven conventional NLP categories: BLiMP, BLiMP Supplement, EWoK, Entity Tracking, COMPS, GlobalPIQA, and (Super)GLUE. The reported aggregates are

$$\mathrm{NLP} = \frac{1}{7}\sum_{c\in\mathcal{N}} s_c,$$

$$\mathrm{HumanLike} = \frac{s_{\mathrm{Reading}} + s_{\mathrm{AoA}}}{2},$$

$$\mathrm{Overall} = \frac{\sum_{c\in\mathcal{N}} s_c + s_{\mathrm{Reading}} + s_{\mathrm{AoA}}}{9}.$$

The aggregation script assigns zero to missing categories. AoA is unavailable for Dense-30M and Ternary-30M, so their Overall scores are incomplete and cannot be compared directly with complete nine-category scores. For this pair, we use the mean over the eight available categories as the primary aggregate and report the zero-filled Overall score only to match the leaderboard calculation.

Checkpoints and provenance. Final-checkpoint evaluations use the last saved state of each run rather than a checkpoint selected according to validation loss or downstream performance. AoA uses 19 checkpoints distributed across pretraining and is available only for Causal-30M, Hybrid-30M, and Hybrid-65M.

The reported results do not all have the same evaluation provenance. Causal-30M's results come from the server-held-out evaluation. The Hybrid-30M and Hybrid-65M scores are preliminary local aggregates assembled from artifacts spanning two evaluator revisions. Causal-30M and Hybrid-30M are closely matched at the training level, but their benchmark results were not produced by a common evaluation pipeline. Comparisons between the server and local scores are therefore descriptive rather than definitive leaderboard rankings.

6 Results

Table 4 reports the available evaluation results. Causal-30M, Hybrid-30M, and Hybrid-65M make up the objective and model-configuration comparison at the training level, although their reported benchmark results do not share the same evaluation provenance. Dense-30M and Ternary-30M form a separate architecture- and evaluation-matched comparison using a different corpus generation, tokenizer, and context length. We interpret results within each experimental group. Cross-group differences are not controlled effects.

6.1 Causal versus hybrid training

Causal-30M and Hybrid-30M are our most closely matched training comparison. They share architecture, parameter count, initialization, corpus, tokenizer, context length, optimization settings, seed, and total word exposure. Their intended experimental difference is the objective recipe: Causal-30M uses causal language modeling on every update, whereas Hybrid-30M alternates between causal and masked-language-modeling updates. The hybrid condition also changes attention visibility, input corruption, prediction targets, and the number of loss-bearing positions.

On the common held-out causal next-token metric, Causal-30M reaches a final validation loss of 3.5421, compared with 3.7891 for Hybrid-30M. The tested causal recipe therefore produces better held-out causal next-token predictions in this run. This is not an objective-neutral measure of general

Model	BLiMP	Supp.	EWoK	Entity	COMPS	GPIQA	(S)GLUE	Read.	AoA	NLP	Human	Overall
Objective and model-configuration comparison: 8k BPE, context 512												
Causal-30M	66.04	52.79	50.95	19.97	51.93	35.64	64.20	0.63	17.96	48.79	9.29	40.01
Hybrid-30M†	63.73	50.03	51.00	16.46	51.23	36.64	64.53	0.73	-11.83	47.66	-5.55	35.83
Hybrid-65M†	65.27	51.87	50.51	31.30	51.64	34.67	65.82	0.37	3.54	50.15	1.95	39.44
Dense–ternary comparison: 16k BPE, context 256												
Dense-30M	**68.33**	**55.61**	**52.23**	19.93	**51.70**	**36.62**	**62.97**	**0.19**	–	**49.63**	–	**38.62**‡
Ternary-30M	65.90	54.63	52.17	**20.45**	51.63	35.58	62.62	0.18	–	49.00	–	38.13‡

Table 4: Available BabyLM evaluation results. Causal-30M was evaluated using the server-held-out pipeline. Results marked with † are preliminary local results assembled from evaluation artifacts spanning two evaluator revisions. Dense-30M and Ternary-30M were not evaluated on AoA; their displayed Overall scores assign zero to the missing category and are incomplete aggregates (‡). Boldface identifies the numerically higher value within the evaluation-matched dense–ternary comparison and does not indicate statistical significance.

model quality, since causal prediction is optimized on every Causal-30M update but only roughly half of Hybrid-30M's updates.

The benchmark outputs did not come from a common evaluation pipeline. Causal-30M was evaluated through the server-held-out pipeline, whereas Hybrid-30M was evaluated locally using preliminary artifacts spanning two evaluator revisions. We report both rows but treat the numerical gap between their Overall scores as descriptive, not as a controlled causal–hybrid effect.

In the displayed results, Causal-30M is numerically higher on BLiMP, the BLiMP Supplement, Entity Tracking, COMPS, and AoA, whereas Hybrid-30M is numerically higher on EWoK, GlobalPIQA, (Super)GLUE, and Reading. These task-level patterns come from mismatched evaluation pipelines and are therefore descriptive. The preliminary Hybrid-30M AoA value of -11.83 accounts for most of the displayed Overall difference. This value has not been verified under a common frozen evaluator revision, so we treat it and aggregates derived from it as provisional.

6.2 Larger hybrid configuration

Hybrid-65M has 64.57M parameters, about 2.21 times as many as Hybrid-30M. It also differs in width, depth, and attention-head geometry, so this is a comparison between model configurations rather than a controlled change in parameter count alone.

Within the preliminary local evaluation outputs, the larger model raises the Overall aggregate from 35.83 to 39.44 and reduces final validation loss from 3.7891 to 3.6122. The clearest gain is on Entity Tracking, which increases from 16.46 to 31.30. Hybrid-65M also scores higher on BLiMP, the BLiMP Supplement, COMPS, (Super)GLUE, and AoA. Its NLP Average rises from 47.66 to 50.15, the highest displayed NLP Average among the three objective and model-configuration runs. Because both hybrid aggregates are assembled from preliminary local outputs, these differences should also be interpreted descriptively.

The gains are uneven. Hybrid-65M scores below Hybrid-30M on EWoK, GlobalPIQA, and Reading. Its preliminary Overall value is numerically below Causal-30M's server-reported value, although their unmatched evaluation provenance prevents interpreting this difference as a ranking. Causal-30M has a Human-like Average of 9.29, compared with 1.95 for Hybrid-65M.

As a sensitivity calculation, excluding AoA gives means of approximately 43.93 for Hybrid-65M and 42.77 for Causal-30M over the remaining eight displayed categories. The ordering of the displayed aggregates reverses when AoA is removed. The two rows have different evaluation provenance, so this calculation shows how sensitive the aggregate is to AoA. It does not establish an evaluation-matched ranking between the models.

6.3 Dense versus ternary training

Dense-30M and Ternary-30M share the same architecture, parameter count, corpus, tokenizer, context length, batching configuration, seed, and training exposure. The main differences are the ternary forward-pass projections and the lower peak learning rate used for Ternary-30M.

AoA was not evaluated for either model. Across the eight categories measured for both, Dense-30M averages 43.45 and Ternary-30M averages 42.90. In this single comparison, the ternary configuration retains 98.73% of the dense model's measured aggregate. Table 4 also reports the leaderboard-style Overall values obtained by assigning zero to

missing AoA, but we do not use those incomplete aggregates for comparison.

Dense-30M is numerically higher on seven of the eight available categories and has a lower final validation loss (3.9703 versus 4.0296). Ternary-30M is slightly higher on Entity Tracking. The models use different peak learning rates and only one seed, so the remaining gap cannot be attributed to ternarization alone.

7 Discussion

Causal and hybrid training. Causal-30M reaches lower held-out causal next-token loss than the closely matched Hybrid-30M configuration. In this run, the tested causal recipe produces better causal predictions, but the metric is not objective-neutral: Causal-30M optimizes causal prediction at every update, while Hybrid-30M does so on roughly half of them.

The displayed benchmark patterns are also mixed. Causal-30M is numerically higher on several categories, particularly AoA. Hybrid-30M is higher on EWoK, GlobalPIQA, (Super)GLUE, and Reading. The causal results are server-reported and the hybrid results are preliminary local outputs spanning two evaluator revisions, so these differences are descriptive, not evaluation-matched evidence of objective superiority.

Our experiment evaluates the complete hybrid recipe, not masked modeling in isolation. Hybrid training changes attention visibility, input corruption, prediction targets, and the number of loss-bearing positions all at once. Our procedure is also an independent implementation, not an exact replication of GPT-BERT. The differing outcomes suggest that hybrid performance depends on architecture, masking procedure, objective schedule, and evaluation protocol (Charpentier and Samuel, 2024; Hu et al., 2024).

Model configuration and aggregate sensitivity. Within the preliminary local evaluation, Hybrid-65M improves on Hybrid-30M in validation loss, NLP Average, Entity Tracking, and several other categories. It scores lower on EWoK, GlobalPIQA, and Reading. These differences cannot be attributed to parameter count alone: Hybrid-65M is also wider, shallower, and uses different attention-head geometry. The result therefore describes a larger hybrid configuration rather than an isolated scaling effect.

The displayed aggregate ranking is also sensitive to AoA. Hybrid-65M has the highest preliminary NLP Average, whereas Causal-30M has the highest reported Overall score. When AoA is excluded, their displayed ordering reverses. The two rows have unmatched evaluation provenance, so this does not establish a model ranking. Instead, it shows that Overall is strongly influenced by the developmental component. AoA is only one developmental measure, so these results do not support broader claims about cognitive plausibility.

Ternary capability retention. Ternary-30M remains numerically close to Dense-30M, retaining 98.73% of its measured eight-category aggregate. Dense-30M still has lower validation loss and higher scores on seven of the eight available categories. The models use different peak learning rates and only one seed, so the remaining gap cannot be attributed to ternarization alone.

This result is about capability retention under fake-quantized forward passes, not deployment efficiency. Training keeps full-precision latent weights, several components stay dense, and the implementation uses neither packed ternary storage nor specialized inference kernels. Establishing practical benefits will require matched optimization settings, multiple seeds, and a deployment-aware implementation.

8 Conclusion

We trained five compact Transformers under BabyLM Strict-Small. Causal-30M achieved the lowest causal validation loss, Hybrid-65M produced the highest preliminary NLP Average and sharply improved Entity Tracking, and Ternary-30M retained 98.73% of Dense-30M's measured eight-category aggregate. Together, these results show that training objective, model configuration, and weight representation produce distinct strengths under restricted data.

9 Limitations

Our experiments have several limitations. First, each model configuration was trained with a single random seed, so we cannot estimate run-to-run variance or establish statistical significance. Although Causal-30M and Hybrid-30M are tightly matched in architecture, data, tokenizer, optimization, and training exposure, their reported benchmark results differ in provenance: Causal-30M was scored using the server-held-out evaluation, whereas the hybrid

scores are preliminary local aggregates. The exact magnitude of the reported performance difference should therefore be interpreted cautiously. The local hybrid results were assembled from artifacts spanning two evaluator revisions. In addition, the causal and hybrid runs are matched in seed and total exposure but not minibatch-for-minibatch: masked-input construction consumes additional random numbers, causing their sampled training windows to diverge. They are also matched in word exposure rather than in the number of loss-bearing prediction targets.

The Hybrid-30M and Hybrid-65M comparison does not isolate model scale. Hybrid-65M is simultaneously larger, wider, shallower, and configured with different attention-head geometry. Similarly, the dense–ternary comparison does not isolate weight representation completely because Ternary-30M uses a lower peak learning rate than Dense-30M. Age of Acquisition was not evaluated for either model in this pair, so their displayed Overall scores are incomplete aggregates that assign zero to the missing category. The two experimental groups also use different corpus generations, tokenizer vocabulary sizes, context lengths, dropout settings, and batching configurations; consequently, comparisons across groups are descriptive rather than controlled. The training corpora exclude CHILDES and therefore omit a major source of child-directed speech, limiting their developmental and domain diversity.

Our ternary experiment evaluates fake-quantized forward projections rather than an efficient deployment implementation. Training retains full-precision latent weights, several model components remain dense, checkpoints are not bit-packed, and no specialized ternary inference kernels were evaluated. We therefore make no empirical claims about storage reduction, inference latency, throughput, or energy consumption.

Finally, the exposure accounting is not equally precise across the two experimental generations. For Causal-30M, Hybrid-30M, and Hybrid-65M, we reconstructed the sampled token windows and obtained exact counts just below 100 million whitespace-word exposures. For Dense-30M and Ternary-30M, the reported value of 99,999,545 exposures comes from an earlier procedure that multiplies the processed-token count by the corpus-level words-per-token ratio. It should therefore be treated as an estimate rather than an exact word-by-word count. These exposure budgets correspond to approximately 10.41–10.42 exposure-equivalent passes over the optimizer splits, not ten literal epochs, because examples are sampled with replacement.

Exact block-level deduplication was performed within each corpus generation, and no exact normalized-block overlap was found between its training, validation, and test splits. We did not, however, conduct an exhaustive semantic-overlap analysis against every evaluation item. These limitations restrict our conclusions to the configurations, seeds, and evaluation procedures examined here.

References

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *Preprint*, arXiv:1308.3432.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every layer counts BERT. *Preprint*, arXiv:2311.02265.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? *Preprint*, arXiv:2410.24159.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. *Preprint*, arXiv:2412.05149.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark,

Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (EWOK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Preprint*, arXiv:2405.09605.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Fengfu Li, Bin Liu, Xiaoxing Wang, Bo Zhang, and Junchi Yan. 2016. Ternary weight networks. *Preprint*, arXiv:1605.04711.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *Preprint*, arXiv:1711.05101.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit LLMs: All large language models are in 1.58 bits. *Preprint*, arXiv:2402.17764.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2022. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. *Preprint*, arXiv:2210.01963.

Ofir Press, Noah A. Smith, and Mike Lewis. 2021. Shortformer: Better language modeling using shorter inputs. *Preprint*, arXiv:2012.15832.

David Samuel, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2023. Trained on 100 million words and still in shape: BERT meets the British National Corpus. *Preprint*, arXiv:2303.09859.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *Preprint*, arXiv:1508.07909.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *Preprint*, arXiv:2308.02019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Preprint*, arXiv:1706.03762.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. BitNet: Scaling 1-bit transformers for large language models. *Preprint*, arXiv:2310.11453.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2025. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. *Preprint*, arXiv:2504.08165.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2019. BLiMP: The benchmark of linguistic minimal pairs for english. *Preprint*, arXiv:1912.00582.

Can Randomly Permuting the LM Decoding Order Boost Data-Efficiency?

Max Ploner and Alan Akbik

Humboldt-Universität zu Berlin
Science Of Intelligence
<firstname>.<lastname>@hu-berlin.de

Abstract

Causal language models, typically trained with next-token prediction, condition each token's probability on all preceding tokens. When pretraining is limited by the amount of available training data (rather than compute), repeating training over the same corpus yields a largely unchanged learning signal across epochs. In contrast, *permutation language models* (PLMs) predict tokens in a random order, conditioning each prediction on a bidirectional set of tokens and thereby increasing the diversity of contextual evidence a token can receive. We hypothesize that PLMs (1) profit more from longer training on repeated data, (2) outperform classical CLMs in data-limited settings where multiple epochs are allowed, and (3) excel on tasks where bidirectional context is useful.

To test these hypotheses, we train multiple models on the BabyLM strict-small pretraining data consisting of only 10M words and evaluate how decoding order and bidirectional conditioning affect model performance under tight training-data budgets. We find that while PLMs profit from longer training, they do not generally outperform CLMs except on tasks where use of bidirectional context is required.

1 Introduction

Modern pretraining relies on massive scaling of both data and compute (Hoffmann et al., 2022; Grattafiori et al., 2024). In low-data settings, where the amount of *unique* training data is limited, the training process must make the most of the limited available samples. A common approach is to train for multiple epochs on the same dataset.

When training causal language models (CLMs) with the next-token prediction (NTP) objective, each token's prediction is conditioned on the same preceding context. This raises the question of whether the available self-supervision signal is really used as efficiently as possible.

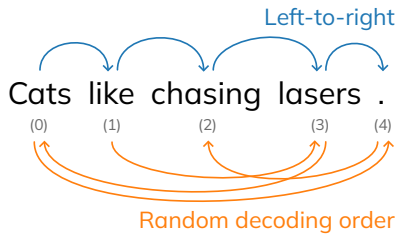


Figure 1: Left-to-right decoding (often implemented via next-token prediction) illustrated by the arrows on top. Random decoding order illustrated by the arrows on the bottom: While the decoding order is random, the tokens' positional embedding indices (in parentheses) stay the same.

In contrast to CLMs, permutation language models decode tokens in random order, and each token is predicted based on a random set of tokens that can be on either side of the current token (see Figure 1). This requires the model to utilize bidirectional context and increases the number of ways a token may need to be predicted. Hence, we hypothesize that PLMs

1. can profit more from longer training on repeated data,
2. can outperform classical CLMs in settings where data is limited, but multiple training epochs are possible, and
3. outperform CLMs in tasks where bidirectional context can be helpful.

To test these hypotheses, we use the pretraining data provided by the *strict-small* track of the BabyLM Challenge (Choshen et al., 2026) to train two different model sizes using varying numbers of epochs. Here, the goal is to train language models on very little data (10M and 100M words, respectively Choshen et al., 2026). This is in stark contrast to contemporary language models, which use *at least* four orders of magnitude more training

Proceedings of the Second BabyLM Workshop, pages 299–308
October 29, 2026 ©2026 Association for Computational Linguistics

data (Hoffmann et al., 2022; Grattafiori et al., 2024). Within the challenge, multiple training epochs (up to ten) are allowed as long as the number of unique tokens stays the same.

2 Background

2.1 Factorization Order

Classical autoregressive language models factorize the likelihood of a sequence $\mathbf{x} = (x_1, ..., x_T)$ as a forward product (Radford et al., 2018; Yang et al., 2019):

$$p(x) = \prod_{t=1}^{T} p\left(x_t | x_{<t}\right)$$

While this may be the most intuitive factorization, $p(x)$ can be factorized in various ways. Each possible permutation $\mathbf{z} \in \mathcal{Z}_T$ of length T induces its own factorization by conditioning each token's probability not on all prior tokens, but on all tokens that come before token t in permutation $\mathbf{z}$. The language modeling objective can be expressed as:

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \in \mathcal{Z}_T} \left[\sum_{t=1}^{T} \log p_\theta \left(x_{z_t} | x_{z_{<t}} \right) \right]$$

In principle, we could train a PLM with next-token prediction either by permuting the order of the inputs (together with their position embeddings) or by adapting the attention matrix. However, it would be unclear which token to predict next.

2.2 Two-Stream Attention

To address this issue of the undetermined next position, Yang et al. (2019) introduce XLNet as a generalized autoregressive pretraining method that trains the LM over multiple permutations of the factorization order, allowing the model to learn deep bidirectionality while using the more efficient autoregressive training. XLNet incorporates two important aspects of Transformer-XL (Dai et al., 2019): Relative positional encodings and the segment recurrence mechanism.

More importantly, though, it uses two residual streams (*two-stream attention*): One stream is used to predict the current token (the decoding stream). It is initialized with a learned vector (similar to the embedding of a mask token) or simply with a zero vector. The second stream enriches the current token representation (initially zero or a learned static vector) with the available context from the first stream. In contrast, CLMs need to encode context information and decode the next token within the same stream. Since during training, they are always conditioned on the previous tokens, they are not trained to use bidirectional context. See Figure 2 for an illustration.

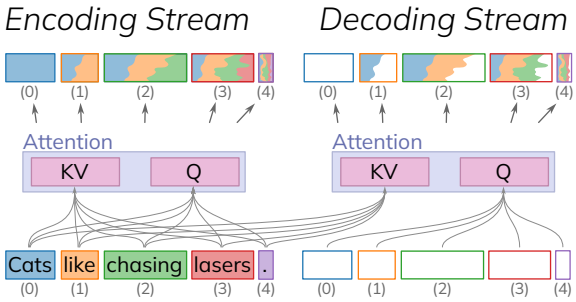


Figure 2: Dual stream attention (here with left-to-right decoding order): The encoding stream initializes the residual stream with the current tokens' embedding, and the decoding stream initializes the residual stream with zero. This way, the encoding stream aggregates information about the current token *and* the context, while the decoding stream aggregates information *only* about the context to predict the current token.

All operations in the two-stream architectures are identical and share weights for the two streams. This means the model's parameter footprint is equal to that of a CLM. The training, however, requires approximately twice the compute and twice the activation memory (for parameters and gradients, the memory stays the same). [1]

We hope that this increase in compute can be offset by the improved deep bidirectionality of the model and potentially by being less prone to overfitting (as we can train on multiple different decoding orders given the same batch). In our work, we apply the idea of two-stream attention to a more recent transformer architecture, namely Llama (Touvron et al., 2023).

Further, Yang et al. (2019) train XLNet using partial prediction, restricting the gradient signal again only to about 15% (one in every 6 or 7 tokens), to stabilize early training and improve convergence. Since we do want the model to also work as a CLM, we instead predict on the complete signal and mix random decoding orders with left-to-right orders.

[1] The alternative for training bidirectional models, masked language modeling (MLM), produces a gradient signal typically only for 15% of the tokens, while the one-stream PLM produces a gradient signal on 100% of the tokens (yielding a higher signal/compute ratio than MLM).

Layers	12
Hidden dimension	768
Attention heads	12
Head dimension	64
Hidden dimensions of feedforward block	3072
Vocabulary size	16384

Table 1: Model configuration (small, 126M parameters).

LR schedule	cosine
Warm up	213 steps
Batch size	16
Max. gradient norm	1.0
Attention dropout	10%
Initializer range	0.02
Sequence length	512 tokens

Table 2: Settings used during pretraining.

3 Experiment 1: Training Length

To test our first hypothesis, we train classical CLMs using NTP and two-stream PLMs in the same training setup and varying numbers of epochs. For the PLMs we additionally vary how many samples need to be decoded in random vs. left-to-right order (random decoding ratio). A zero ratio means the model is trained solely using left-to-right (similar to a single-stream model, but using a dual-stream architecture).

3.1 Experimental Setup

We compare CLMs and PLMs over three different training lengths: Using varying numbers of epochs ($\{2, 5, 7, 10, 12, 15, 20\}$; with 10 epochs matching the maximum allowed training length for submissions to the BabyLM Challenge). For the PLMs, we choose random decoding ratios from $\{0.0, 0.1, 0.25, 0.5, 0.75, 0.9\}$ to test the impact of the random decoding order more gradually.

Model. As a representative for CLM models, we choose a model with Llama architecture (Touvron et al., 2023) and a configuration as reported in Table 1. We tie the word embeddings and use an existing tokenizer built for this corpus. [2]

Pretraining Setting. We pretrain each model on the 10M strict-small track data of the BabyLM Challenge. The used pretraining settings are listed in Table 2. We use a cosine learning rate scheduler and search for the ideal learning rate through grid search over a logarithmic grid ($\{3 \times 10^{-5}, 1 \times 10^{-4}, 3 \times 10^{-4}, 1 \times 10^{-3}\}$) first on classical next-token prediction. We found the learning rate of 3×10^{-4} 3×10^{-4} to produce the best results over all training lengths. A limited additional grid search confirmed this learning rate for PLMs as well. Going forward, we trained only using the learning rate across all settings.

[2] https://huggingface.co/momergul/babylm-baseline-10m-gpt2

Evaluation. In this first set of experiments, we evaluate on BLiMP (Warstadt et al., 2020), as it gave more consistent results (fewer fluctuations across similar hyperparameter settings) during our preliminary experiments. We use *lm evaluation harness* (Sutawika et al., 2024) to test the model. In the second experiment, we then evaluate on all tasks included in the evaluation pipeline (see Section 4).

3.2 Results

There is a clear reduction of performance after training on the next-token objective for 5 epochs (as visible in the first row of Table 3). For the models trained using two-stream attention and a mixture of left-to-right and random decoding orders, the peak performance tends to occur later in training: The more samples are used in random order, the later the peak occurs.

While using a mixture of 25% random decoding orders slightly increases performance, most training mixing ratios perform worse than the one-stream trained models. Hence, this only shows that the training objective is delayed: Since there are fewer left-to-right decoded samples in each batch, the model potentially simply requires more batches. While this does confirm Hypothesis 1, this does not confirm Hypothesis 2.

Takeaway 1

Peak performance requires more training epochs the higher the ratio of random decoding orders.

Comparing these results to the results in Table 7 in the appendix, we find that left-to-right decoding (during evaluation) produces higher scores. This holds regardless of the random decoding ratio that was used during training, though the performance is generally higher for lower mixing ratios (around 25%).

Model	Epochs / Ratio	2	5	7	10	12	15	20
one-stream	-	64.52±0.16	66.90±0.56	65.80±0.32	63.96±0.57	63.11±0.21	62.39±0.08	61.93±0.74
two-stream	0.00	65.08±0.77	66.56±0.53	65.89±0.51	63.82±0.20	63.15±0.72	61.14±1.08	62.94±0.65
	0.10	64.93±1.51	66.33±1.78	66.44±1.36	63.55±0.93	63.20±0.50	62.67±1.13	60.90±0.55
	0.25	64.30±0.93	67.25±0.95	67.16±0.59	65.85±0.95	64.28±0.26	62.91±0.70	61.95±0.78
	0.50	63.54±0.79	65.69±0.47	66.33±0.69	65.24±0.77	64.67±0.46	64.39±0.62	62.27±0.17
	0.75	62.98±0.60	63.79±0.57	65.05±1.28	65.22±0.70	65.61±0.28	64.93±0.78	64.30±1.36
	0.90	61.46±0.90	63.32±0.85	63.73±1.23	63.46±1.45	64.28±0.49	63.49±0.87	63.32±0.40
	1.00	60.14±1.47	60.30±1.40	60.58±0.84	60.92±0.62	61.03±1.21	61.30±2.38	61.01±0.15

Table 3: Evaluation results on BLiMP using two-stream attention and a left-to-right decoding order on models trained using different ratios of random decoding orders (rows; mixed in with left-to-right training) across varying numbers of epochs (columns). The one-stream trained models (llama) are also evaluated using one-stream attention. Each cell represents three independent training runs. The interval represents one standard deviation on each side of the mean. Each setting was run three times with different random seeds.

Takeaway 2

During evaluation, using a left-to-right decoding produces better results even if the model was trained entirely on random decoding orders.

Since multiple decoding orders exist, one could also use multiple decoding orders during evaluation to make the prediction (i.e., using multiple forward passes and averaging the perplexities). While we did not explore this option, it would make for an interesting experiment in future work, as this may not only improve performance but also produce a distribution of perplexity which could be used to estimate the model's confidence.

4 Experiment 2: Impact of Deep Bidirectionality

With the second experiment, we aim to address the remaining two hypotheses: Can the two-stream attention model outperform the one-stream model? Is it particularly suited when using bidirectional context?

4.1 Experimental Setup

To address these questions, we largely use the same setup as in the previous experiment. For each model, we pick the setting that produced the best results.

Model. We use a *small* architecture as in the previous section (see Table 1) as well as a slightly larger *medium* model (see Table 4). For each size, we pre-train three variants: A classical CLM model (single-stream; trained with NTP), a two-stream model with a 25% ratio of random decoding orders (as it performed best in the previous section; see Table 3), and a two-stream model with a 75% ratio to test whether a higher ratio shows improved performance on tasks where bidirectional context may be more important.

Layers	24
Hidden dimension	1024
Attention heads	16
Head dimension	64
Hidden dimensions of feedforward block	4096
Vocabulary size	16384

Table 4: Model configuration (medium model; 419M parameters).

Pretraining Setting. We use the same pretraining setting as in the previous section (see Table 2).

Evaluation. After training, we execute the evaluation pipeline provided by the BabyLM Challenge.

The evaluation pipeline consists of a set of **zero-shot** tasks, including BLiMP (Warstadt et al., 2020), COMPS (Misra et al., 2023), Entity Tracking (Kim and Schuster, 2023), as well as EWoK (Ivanova et al., 2025).

Further, it includes the SuperGlUE (Wang et al., 2019) tasks BoolQ (Clark et al., 2019), MNLI (Williams et al., 2018), MRPC (Dolan and Brockett, 2005), MultiRC (Khashabi et al., 2018), QQP (Wang et al., 2018), RTE (Levesque et al., 2012),

Size	Task / Model	BLiMP	BLiMP (supp.)	COMPS	Entity Tracking	EWoK
medium	one-stream	**69.01**$_{\pm 0.59}$	**58.88**$_{\pm 0.90}$	51.96$_{\pm 0.46}$	**19.41**$_{\pm 0.56}$	51.77$_{\pm 0.44}$
	two-stream (25%)	68.40$_{\pm 0.27}$	58.00$_{\pm 1.19}$	52.15$_{\pm 0.08}$	18.15$_{\pm 0.60}$	51.76$_{\pm 0.74}$
	two-stream (75%)	66.36$_{\pm 0.42}$	55.25$_{\pm 1.43}$	**52.92**$_{\pm 0.18}$	17.87$_{\pm 0.19}$	**52.19**$_{\pm 0.40}$
small	one-stream	68.69$_{\pm 0.44}$	**60.64**$_{\pm 1.04}$	52.19$_{\pm 0.50}$	**19.69**$_{\pm 0.46}$	51.49$_{\pm 0.59}$
	two-stream (25%)	**69.57**$_{\pm 0.57}$	58.37$_{\pm 0.68}$	52.51$_{\pm 0.62}$	18.13$_{\pm 0.63}$	51.72$_{\pm 1.21}$
	two-stream (75%)	66.68$_{\pm 0.30}$	55.29$_{\pm 1.79}$	**52.95**$_{\pm 0.11}$	19.33$_{\pm 0.96}$	**52.29**$_{\pm 0.67}$

Table 5: Performance of the models on zero-shot tasks.

and WSC (Levesque et al., 2012). We use the default settings provided by the pipeline to **finetune** each model under three settings: (1) unidirectionally (the last token is used to make the prediction), (2) bidirectionally (last token), and (3) bidirectionally using the first token to make the prediction..

Additionally, we fine-tune the models on the CoNLL-2003 named entity recognition (NER) task (Tjong Kim Sang and De Meulder, 2003). In NER tasks, where the system needs to tag entities within the sentence, pooling token representations across the entire sequence is insufficient. Instead, a prediction needs to be made for each token (i.e., token within the dataset, not necessarily each subword token). To successfully predict an entity at the start of a sequence, more context may be required. Here, bidirectional context may be especially helpful.

4.2 Results

Zero-Shot. The two-stream model with the lower random decoding ratio (25%) and the one-stream model exhibit similar performance on the zero-shot tasks (see Table 5). In most cases, the performance difference is within one standard deviation. The exceptions are BLIMP (supp.) in the cases of small models and the Entity Tracking task, where the one-stream model has an advantage.

Potentially, training could be further improved, but under this controlled setting, we cannot observe a performance gain simply by using random decoding orders. We must therefore reject our hypothesis that random decoding orders improve training in the small data regime.

Takeaway 3

Models trained with random decoding orders do not generally outperform classical NTP models in a small pretraining data regime.

Fine-Tuning. A selection of the bidirectional fine-tuning results is presented in Table 6. For clarity, we show only the standard metrics reported for GLUE tasks and omit the tasks RTE and WSC (as they exhibit rather large standard deviations). The complete results including the remaining metrics and tasks can be found in the appendix (Tables 8 and 9).

When fine-tuning, both approaches (one- and two-stream) *tend* to profit from bidirectional fine-tuning, though for the one-stream model, unidirectional fine-tuning produces better results in some cases (MNLI, MultiRC, & QQP). The two-stream models tend to profit more from bidirectional fine-tuning and slightly outperform the one-stream models in cases where the one-stream model works better with bidirectional fine-tuning (all tasks, but MultiRC, MNLI & QQP).

On CoNLL-2003, the difference between unidirectional and bidirectional fine-tuning is (as expected) quite large. Here, the performance difference between the one-stream and two-stream models is quite notable (with a higher random ratio yielding better bidirectional fine-tuning results).

Takeaway 4

Random decoding orders can improve a model's ability to use bidirectional context.

5 Related Work

Further Development of Two-Stream Attention. MPNet (Song et al., 2020) further extended XLNet by adding masked tokens to the context to ensure the model has access to their positional encodings (and hence where tokens are still missing) no matter which token it is trying to predict. MPNet, in its

Size	Task / Metric / Model	BoolQ Acc.	CoNLL-2003 F_1	MNLI Acc.	MRPC F_1	MultiRC F_1	QQP F_1	Avg. -
medium	one-stream	69.03$_{\pm 1.03}$	81.35$_{\pm 0.28}$	43.96$_{\pm 0.68}$	81.99$_{\pm 1.03}$	**51.31**$_{\pm 2.65}$	62.45$_{\pm 0.04}$	65.01
	two-stream (25%)	69.34$_{\pm 0.43}$	82.82$_{\pm 0.17}$	48.18$_{\pm 0.67}$	82.57$_{\pm 0.99}$	49.15$_{\pm 7.44}$	**66.25**$_{\pm 0.59}$	66.38
	two-stream (75%)	**70.30**$_{\pm 0.46}$	**83.69**$_{\pm 0.21}$	**48.97**$_{\pm 0.50}$	**82.58**$_{\pm 0.13}$	51.03$_{\pm 1.64}$	65.07$_{\pm 0.99}$	**66.94**
small	one-stream	68.54$_{\pm 0.87}$	82.02$_{\pm 0.45}$	44.61$_{\pm 0.57}$	80.75$_{\pm 2.37}$	50.41$_{\pm 0.99}$	62.15$_{\pm 0.25}$	64.75
	two-stream (25%)	69.75$_{\pm 0.37}$	83.24$_{\pm 0.34}$	48.51$_{\pm 0.54}$	82.51$_{\pm 0.76}$	**50.72**$_{\pm 4.11}$	**65.84**$_{\pm 1.33}$	**66.76**
	two-stream (75%)	**70.30**$_{\pm 0.43}$	**83.75**$_{\pm 0.18}$	**49.14**$_{\pm 0.58}$	**82.51**$_{\pm 0.30}$	48.09$_{\pm 2.54}$	65.00$_{\pm 1.44}$	66.47

Table 6: Performance of the models after bidirectional fine-tuning on GLUE tasks and CoNLL-2003. Bold: best, underlined: second best.

`all-mpnet-base-v2` variant[3], is still commonly used as an embedding model (Brown et al., 2025).

In a publication concurrent with this paper, Monea et al. (2026) propose the use of state-prediction separation, a transformer variant that disentangles the storage of context for future predictions and the prediction of the next token in two separate streams. The authors confirm this to be a similar technique to two-stream attention, albeit for a different purpose.

Hybrid Next Token and Masked Next Token Prediction. Charpentier and Samuel (2024) train a hybrid model with two pretraining objectives: next-token prediction, used in causal language models, and masked next-token prediction (MNTP). MNTP is a variant of MLM in which the model does not predict the masked token at the same position; instead, it predicts that token using the previous token, aligning the task more closely with the NTP objective. This hybrid training aligns the model well with the tasks included in the BabyLM baseline.

However, this hybrid approach may introduce an additional benefit to the limited data setting of the BabyLM challenge: It increases the number of different training signals that can be derived from a single sample.

Lifting the Factorization Curse. While (to our knowledge) permutation language modeling was not further developed after the publication of MPNet, the topic resurfaced in the discussion about the *reversal* or *factorization curse*.

Allen-Zhu and Li (2023b) and Berglund et al. (2024) concurrently observed that a causal language model trained on "Valentina Tereshkova was the first woman to travel to space" may not be able to answer the question "Who was the first woman to travel to space?" (example from Berglund et al., 2024). [4] Golovneva et al. (2024) and Kitouni et al. (2024) both suggest that the use of different factorizations (see Section 2.1) may alleviate this issue.

6 Conclusion

We presented two sets of experiments. In the first, we vary the training length as well as the random decoding ratio, which specifies the proportion of sequences that will need to be predicted in random order during training. These experiments indicate that training with higher random decoding ratios requires longer training.

In the second set, we directly compare three sets of models on several zero-shot and fine-tuning tasks: A classical one-stream architecture trained using next-token predictions and two sets of two-stream models, one with a rather low and one with a rather high ratio of random decoding orders (during training). We find that while the two-stream models perform better in tasks where the use of bidirectional context is required, the hypothesis that models trained using random decoding orders can learn more from repeated training samples must ultimately be rejected. The two-stream models reach a similar performance, but do not manage to outperform the one-stream model on most tasks.

While further improvements of the training setup may improve the performance of the two-stream models, these experiments provide preliminary evidence that simply adding random decoding orders

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[4] Allen-Zhu and Li (2023a) point out that classical MLMs may not be suited any better, at least in cases where the entity mention is rather long, since any entity token may simply attend to other tokens in the same mention, making it unnecessary to learn the relational fact.

to the mix does not by itself manage to do so.

Limitations

The experimental setup is constrained along multiple dimensions, primarily due to resource constraints. We only tested a single architecture (llama) and only used a single pretraining corpus. Only one fine-tuning task (CoNLL-2003) was used, which requires token-level predictions. All other fine-tuning tasks operate on a sequence level.

Further, the used pretraining setup (for training two-stream models) may be suboptimal: Other decoding order sampling strategies (e.g., with a left-to-right bias) may yield better results.

Acknowledgments

We thank all reviewers for their valuable comments. Alan Akbik and Max Ploner are supported under Germany's Excellence Strategy "Science of Intelligence" (EXC 2002/2, project number 390523135). Further, Alan Akbik is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Emmy Noether grant "Eidetic Representations of Natural Language" (project number 448414230).

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2023a. Physics of Language Models: Part 3.1, Knowledge Storage and Extraction.

Zeyuan Allen-Zhu and Yuanzhi Li. 2023b. Physics of Language Models: Part 3.2, Knowledge Manipulation.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Stickland, Tomek Korbak, and Owain Evans. 2024. The Reversal Curse: LLMs trained on "A is B" fail to learn "B is A". *International Conference on Learning Representations*, 2024:18623–18642.

Andrew Brown, Muhammad Roman, and Barry Devereux. 2025. A Systematic Literature Review of Retrieval-Augmented Generation: Techniques, Metrics, and Challenges. *Big Data and Cognitive Computing*, 9(12):320.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both?

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM Turns 4 and Goes Multilingual: Call for Papers for the 2026 BabyLM Workshop.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Olga Golovneva, Zeyuan Allen-Zhu, Jason E. Weston, and Sainbayar Sukhbaatar. 2024. Reverse Training to Nurse the Reversal Curse. In *First Conference on Language Modeling*.

Aaron Grattafiori, Abhimanyu Dubey, and LLama3 Team. 2024. The Llama 3 Herd of Models.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training Compute-Optimal Large Language Models.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of World Knowledge (EWoK): A Cognition-Inspired Framework for Evaluating Basic World Knowledge in Language Models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Najoung Kim and Sebastian Schuster. 2023. Entity Tracking in Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Ouail Kitouni, Niklas Nolte, Diane Bouchacourt, Adina Williams, Mike Rabbat, and Mark Ibrahim. 2024. The Factorization Curse: Which Tokens You Predict Underlie the Reversal Curse and More. In *Advances in Neural Information Processing Systems*, volume 37, pages 112329–112355. Curran Associates, Inc.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pages 552–561, Rome, Italy. AAAI Press.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual Minimal Pair Sentences for testing Robust Property Knowledge and its Inheritance in Pre-trained Language Models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Giovanni Monea, Nathan Godey, Kianté Brantley, and Yoav Artzi. 2026. The State-Prediction Separation Hypothesis.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding.

Lintang Sutawika, Hailey Schoelkopf, Leo Gao, Baber Abbasi, Stella Biderman, Jonathan Tow, ben fattori, Charles Lovering, farzanehnakhaee70, Jason Phang, Anish Thite, Fazz, Aflah, Niklas Muennighoff, Thomas Wang, sdtblck, nopperl, gakada, tttyuntian, researcher2, Julen Etxaniz, Chris, Hanwool Albert Lee, Zdeněk Kasner, Khalid, LSinev, Jeffrey Hsu, Anjor Kanekar, KonradSzafer, and AndyZwei. 2024. EleutherAI/lm-evaluation-harness: V0.4.3. Zenodo.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 517, pages 5753–5763, Red Hook, NY, USA. Curran Associates Inc.

A Additional Results

Epochs / **Ratio**	2	5	7	10	12	15	20
0.00	52.54±0.05	52.64±0.10	52.52±0.02	52.47±0.16	52.28±0.13	52.15±0.09	52.39±0.06
0.10	53.72±0.19	54.07±0.30	54.12±0.34	53.67±0.30	53.43±0.01	52.72±0.37	52.40±0.12
0.25	54.26±0.03	55.09±0.09	55.27±0.33	54.78±0.07	54.71±0.23	54.38±0.14	53.84±0.50
0.50	54.79±0.04	55.77±0.23	56.07±0.13	55.96±0.22	55.82±0.21	55.69±0.33	54.95±0.26
0.75	55.22±0.13	56.23±0.26	56.61±0.10	56.79±0.15	56.75±0.43	56.64±0.25	56.34±0.37
0.90	55.13±0.13	56.52±0.19	56.82±0.16	56.83±0.25	57.17±0.40	56.75±0.35	56.68±0.29
1.00	55.13±0.25	56.17±0.33	56.41±0.35	56.74±0.11	56.30±0.27	56.63±0.28	56.43±0.02

Table 7: Evaluation results on BLiMP using two-stream attention and a random decoding order on models trained using different ratios of random decoding orders (rows; mixed in with left-to-right training) across varying numbers of epochs (columns). Each cell represents three independent training runs. The interval represents one standard deviation on each side of the mean.

Size	**Model**	**Task** / **Metric** / **Variant**	BoolQ Acc.	CoNLL-2003 F_1	MNLI Acc.	MRPC F_1	MultiRC F_1	QQP F_1	Avg. -
medium	one-stream	bi.	69.03±1.03	81.35±0.28	43.96±0.68	81.99±1.03	**51.31**±2.65	62.45±0.04	65.01
		bi. (last)	67.48±0.53	-	47.72±0.63	80.28±1.66	48.56±1.75	63.45±0.41	61.50
		uni.	68.42±0.93	65.74±0.23	**51.44**±1.23	81.87±1.02	50.90±1.25	**67.55**±0.53	64.32
	two-stream (25%)	bi.	69.34±0.43	82.82±0.17	48.18±0.67	82.57±0.99	49.15±7.44	66.25±0.59	66.38
		bi. (last)	68.79±0.25	-	49.71±0.48	81.06±1.03	50.60±2.28	65.86±0.51	63.20
		uni.	66.18±0.66	65.17±0.48	46.84±0.48	78.87±1.29	44.98±6.40	60.50±0.42	60.42
	two-stream (75%)	bi.	**70.30**±0.46	**83.69**±0.21	48.97±0.50	**82.58**±0.13	51.03±1.64	65.07±0.99	**66.94**
		bi. (last)	68.97±0.82	-	50.28±0.33	82.00±0.69	46.80±4.92	65.39±0.60	62.69
		uni.	65.12±0.89	65.81±0.11	42.43±0.99	78.53±1.52	41.76±0.73	54.43±0.24	58.01
small	one-stream	bi.	68.54±0.87	82.02±0.45	44.61±0.57	80.75±2.37	50.41±0.99	62.15±0.25	64.75
		bi. (last)	67.87±0.09	-	47.24±1.26	82.29±1.79	**52.24**±1.08	64.87±0.44	62.90
		uni.	68.44±0.80	64.96±0.64	**50.67**±0.93	82.00±1.96	49.55±2.42	**68.64**±0.21	64.04
	two-stream (25%)	bi.	69.75±0.37	83.24±0.34	48.51±0.54	82.51±0.76	50.72±4.11	65.84±1.33	**66.76**
		bi. (last)	68.91±0.67	-	49.47±0.36	81.67±0.54	49.48±1.08	66.25±0.47	63.15
		uni.	66.54±0.16	65.38±0.38	47.10±0.36	79.09±0.93	43.68±0.71	60.90±0.41	60.45
	two-stream (75%)	bi.	**70.30**±0.43	**83.75**±0.18	49.14±0.58	**82.51**±0.30	48.09±2.54	65.00±1.44	66.47
		bi. (last)	68.93±0.83	-	50.20±0.22	80.54±0.07	49.36±1.68	66.14±0.33	63.04
		uni.	66.91±0.56	65.55±0.04	41.80±0.96	81.08±0.27	29.18±9.31	55.71±0.60	56.71

Table 8: Performance of the models after fine-tuning on some GLUE tasks and CoNLL-2003.

Size	Model	Task	BoolQ			CoNLL-2003	MNLI	MRPC			MultiRC			QQP			RTE			WSC		
		Metric	Acc.	F_1	MCC	F_1	Acc.	Acc.	F_1	MCC	Acc.	F_1	MCC	Acc.	F_1	MCC	Acc.	F_1	MCC	Acc.	F_1	MCC
		Variant																				
medium	one-stream	bi.	69.03±1.03	77.22±0.48	29.85±3.02	81.35±0.28	43.96±0.68	**72.22**±1.58	81.99±1.03	**28.32**±4.96	66.46±0.39	**51.31**±2.65	29.74±0.83	70.91±0.65	62.45±0.04	38.87±0.72	58.75±2.08	**53.86**±4.35	17.22±3.26	**67.31**±0.00	43.49±11.19	**27.92**±2.07
		bi. (last)	67.48±0.53	77.39±0.72	23.70±1.64	-	47.72±0.63	68.79±2.47	80.28±1.66	15.44±8.85	66.23±0.10	48.56±1.75	29.22±0.21	72.31±0.65	63.45±0.41	41.25±0.53	58.51±3.69	43.89±8.94	16.42±8.02	62.18±1.11	23.67±21.63	9.46±9.15
		uni.	68.42±0.93	76.90±1.29	28.27±2.94	65.74±0.23	**51.44**±1.23	70.59±2.25	81.87±1.02	19.78±10.20	**66.85**±0.69	50.90±1.25	**30.53**±1.60	**75.36**±1.93	**67.55**±0.53	**48.03**±2.01	**58.99**±3.14	51.57±7.88	**18.28**±3.70	63.46±1.92	16.05±15.43	11.76±10.71
	two-stream (25%)	bi.	69.34±0.43	78.15±1.23	29.54±1.46	82.82±0.17	48.18±0.67	71.57±1.77	82.57±0.99	25.65±8.37	66.02±0.60	49.15±7.44	29.25±1.11	73.50±0.64	66.25±0.59	44.75±0.63	54.44±0.83	18.78±32.53	3.96±6.86	63.46±1.92	14.13±12.82	12.01±11.03
		bi. (last)	68.79±0.25	77.85±0.71	27.86±2.01	-	49.71±0.48	70.42±1.98	81.06±1.03	21.70±7.23	66.31±0.58	50.60±2.28	29.28±1.29	72.34±0.34	65.86±0.51	43.30±0.43	58.03±2.53	41.02±23.51	14.70±5.92	62.18±2.22	15.67±14.89	6.14±14.91
		uni.	66.18±0.66	76.49±1.08	20.33±0.51	65.17±0.48	46.84±0.48	68.79±1.02	78.87±1.29	21.50±1.28	64.25±0.92	44.98±6.40	24.82±1.92	70.98±0.16	60.50±0.42	37.58±0.45	52.52±4.01	50.15±5.04	4.97±8.14	64.74±5.55	43.02±10.73	20.83±13.66
	two-stream (75%)	bi.	**70.30**±0.46	**79.19**±0.34	**31.14**±1.17	**83.69**±0.21	48.97±0.50	72.22±0.28	**82.58**±0.13	27.57±1.03	66.54±0.39	51.03±1.64	29.88±0.93	72.79±0.34	65.07±0.99	43.04±0.48	55.40±1.90	21.16±34.02	8.86±8.69	65.38±3.33	32.20±28.06	18.33±15.89
		bi. (last)	68.97±0.82	77.93±0.97	28.39±2.52	-	50.28±0.33	71.24±1.02	82.00±0.69	23.87±3.88	66.28±0.44	46.80±4.92	29.76±0.70	73.11±1.01	65.39±0.60	43.68±0.95	55.16±1.50	23.17±28.06	6.62±6.53	64.10±1.11	23.50±10.58	16.59±3.88
		uni.	65.12±0.89	74.82±1.09	19.70±2.25	65.81±0.11	42.43±0.99	67.65±2.55	78.53±1.52	16.40±8.06	60.11±1.14	41.76±0.73	14.99±2.42	69.75±0.30	54.43±0.24	32.83±0.56	51.08±2.49	49.74±2.80	2.38±5.07	63.46±3.85	**50.15**±16.30	23.99±14.66
small	one-stream	bi.	68.54±0.87	77.11±0.97	28.27±1.56	82.02±0.45	44.61±0.57	71.41±2.87	80.75±2.37	28.31±6.36	66.78±0.70	50.41±0.99	30.40±1.66	72.06±0.88	62.15±0.25	40.06±1.12	**55.88**±1.50	44.83±8.29	10.41±2.27	63.46±1.92	18.86±24.88	13.67±12.17
		bi. (last)	67.87±0.09	76.60±0.20	26.69±0.75	-	47.24±1.26	**72.88**±2.42	82.29±1.79	30.87±6.80	66.42±0.43	**52.24**±1.08	29.56±0.98	70.42±0.35	64.87±0.44	40.74±0.35	54.20±3.69	49.09±3.59	7.52±7.30	64.10±2.94	31.48±15.75	17.28±9.00
		uni.	68.44±0.80	77.08±0.82	27.93±1.86	64.96±0.64	**50.67**±0.93	**72.88**±2.21	82.00±1.96	**31.78**±5.20	66.75±0.68	49.55±2.42	30.43±1.53	72.21±0.52	**68.64**±0.21	**46.71**±0.45	55.16±2.20	48.42±3.81	9.05±4.73	62.18±1.11	39.80±7.08	15.32±3.24
	two-stream (25%)	bi.	69.75±0.37	**78.20**±0.93	30.78±0.94	83.24±0.34	48.51±0.54	71.73±1.50	82.51±0.76	26.68±5.73	**66.80**±0.48	50.72±4.11	30.81±1.16	73.66±0.65	65.84±1.33	44.54±1.74	54.68±2.59	20.13±34.87	2.07±14.37	**66.67**±2.94	**40.50**±28.40	**29.30**±10.62
		bi. (last)	68.91±0.67	77.70±0.89	28.55±1.54	-	49.47±0.36	71.57±0.49	81.67±0.54	26.54±2.84	66.53±0.31	49.48±1.08	29.91±0.89	74.23±0.63	66.25±0.47	45.53±0.38	55.64±0.42	**54.03**±2.08	**11.44**±0.87	62.18±1.11	8.00±13.86	4.81±8.34
		uni.	66.54±0.16	75.58±0.48	23.68±1.51	65.38±0.38	47.10±0.36	68.14±0.98	79.09±0.93	16.88±1.99	63.49±0.35	43.68±0.71	22.64±0.85	72.06±0.48	60.90±0.41	39.34±0.54	53.24±2.49	50.90±1.61	6.37±4.69	57.69±3.33	15.30±7.29	-3.86±10.74
	two-stream (75%)	bi.	**70.30**±0.43	77.81±0.89	**33.47**±0.76	**83.75**±0.18	49.14±0.58	72.71±1.23	**82.51**±0.30	30.09±4.18	66.79±0.23	48.09±2.54	**30.87**±0.93	72.31±0.23	65.00±1.44	42.51±1.31	55.16±0.83	38.17±14.45	7.91±2.45	62.82±1.11	6.35±5.50	11.81±10.23
		bi. (last)	68.93±0.83	77.46±1.59	29.26±2.49	-	50.20±0.22	69.44±0.57	80.54±0.07	18.10±3.48	65.76±0.66	49.36±1.68	28.11±1.60	**74.51**±0.49	66.14±0.33	45.75±0.56	55.16±0.42	22.44±24.01	9.35±4.00	60.26±1.11	12.42±14.65	-1.19±10.66
		uni.	66.91±0.56	76.82±0.37	22.51±2.85	65.55±0.04	41.80±0.96	70.42±0.28	81.08±0.27	21.84±2.16	59.63±1.92	29.18±9.31	12.00±5.68	69.82±0.19	55.71±0.60	33.46±0.50	53.96±1.90	52.39±1.79	8.14±2.76	58.97±2.22	29.39±12.18	4.89±8.41

Table 9: Complete fine-tuning results.

When the Weakest Language Keeps Moving: ARMOR-DP for Multilingual BabyLM

Hanyu Liao[2]* **Yijing Chen**[1]* **Tao Ji**[1]✉
[1]Fudan University
[2]East China Normal University
51270400017@stu.ecnu.edu.cn {yijingchen24@m., taoji@}fudan.edu.cn

Abstract

Multilingual BabyLM systems usually fix language sampling proportions even though the weakest language can change during training. We present ARMOR-DP, a modular framework that diagnoses imbalance arising from representation, exposure, capacity, and objective pressure, and uses per-language validation loss to update sampling. Controlled experiments show that equal character budgets create a severe representation artifact for Chinese; byte-level encoding removes much of it, while static Chinese protection can subsequently make English the weakest language. The dynamic controller tracks these transitions and achieves a better score–balance tradeoff than fixed protection. On official evaluation, curriculum-dynamic checkpoints outperform a matched static Chinese-protective baseline while using fewer Chinese epochs; scaling capacity yields the strongest internal ARMOR-DP system. These results support a simple conclusion: multilingual weakness is a training-time state rather than a fixed property of a language. ARMOR-DP therefore separates diagnosis from intervention, allowing protection to follow the current bottleneck instead of a predetermined language label.

1 Introduction

The BabyLM challenge tasks a model with acquiring language from a developmentally plausible data budget, and the 2026 BabyBabelLM track extends this premise to the multilingual setting (BabyLM Organizers, 2026b,a). Naturalistic linguistic input is not a fixed curriculum: properties of child-directed speech change over development and can track children's emerging linguistic abilities (Vosoughi and Roy, 2012). We use this responsiveness as a motivating analogy, not as a claim that caregivers optimize explicit language proportions.

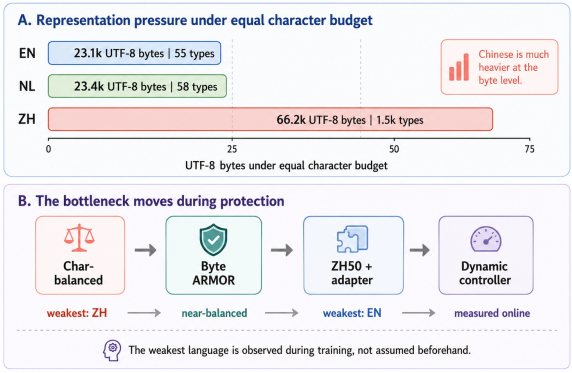


Figure 1: Central hypothesis of ARMOR-DP. Panel A shows that equal character budgets do not create equal representation burden across scripts; Panel B shows that the bottleneck can move after protection is added. The takeaway is that weakest-language protection should be measured during training rather than fixed to one language in advance.

In contrast, standard multilingual pretraining relies on static sampling temperatures or predetermined corpus proportions, which are agnostic to the model's evolving proficiency. Under BabyLM's stringent capacity and data constraints, languages compete for a shared parameter budget, and a fixed ratio can underserve whichever language currently lags. A responsive artificial learner should therefore reallocate exposure online. Doing so requires first diagnosing why a language is weak at a particular moment—a question that cannot be answered by exposure alone.

Our central hypothesis is that a language can be *weak* for at least four different reasons. It can be weak because its script is poorly represented by the tokenizer (**representation**), because it receives insufficient effective **exposure**, because the shared model has insufficient language-private **capacity**,

*Equal contribution.
✉Corresponding author.

Proceedings of the Second BabyLM Workshop, pages 309–319
October 29, 2026 ©2026 Association for Computational Linguistics

or because the objective allocates gradient pressure in a way that amplifies already frequent units (**objective**). Therefore, we propose **ARMOR-DP** (**Armor** for **D**evelopmental **P**ressure), a modular diagnostic and intervention framework.

Our contributions are threefold. First, we show that the identity of the weakest language moves as representation, exposure, and capacity interventions take effect. Second, ARMOR-DP separates four possible sources of weakness and turns per-language validation loss into an online sampling signal. Third, our official evaluations show that curriculum-dynamic training improves on a matched static Chinese-protective system while remaining within the challenge budget.

2 ARMOR-DP Framework

ARMOR-DP is a modular diagnostic and intervention toolkit. It separates four sources of multilingual weakness, each paired with a corresponding probe or intervention. Components can be enabled or omitted according to empirical evidence rather than treated as a mandatory stack.

Representation armor. We use UTF-8 byte units as a script-fair representation. It removes a key comparison artifact, in which Chinese characters are treated as sparse atomic symbols while Latin-script languages reuse a small alphabet. This choice is based on tokenization-free and byte-level models (Xue et al., 2022; Clark et al., 2022; Tay et al., 2022).

Exposure armor. We compare balanced sampling with ZH-protective schedules. Our primary static baseline uses English/Dutch/Chinese ratios of 25/25/50. We also test a curriculum that starts balanced and switches to 25/25/50 halfway through training. This is related to temperature-based multilingual sampling and curriculum learning (Conneau et al., 2020; Bengio et al., 2009).

Capacity armor. We test two lightweight capacity mechanisms: language adapters and a language-conditioned routing gate over attention heads. We employ the routing gate as an interpretable capacity intervention for multilingual adaptation. Rather than introducing a new routing mechanism, our goal is to analyze how capacity allocation interacts with language imbalance within the ARMOR-DP framework (Houlsby et al., 2019; Pfeiffer et al., 2020; Fedus et al., 2022).

Objective armor. We test both a naive rare-token weighted objective and a pressure-aware objective that combines current excess loss, script pressure, and exposure. The rare-token diagnostic multiplies token-level cross-entropy by a weight derived only from that byte type's frequency in the diagnostic corpus: less frequent types receive larger weights, clipped to the range reported in Appendix 10. It is included as a simple frequency-only analogue of loss reweighting (Lin et al., 2017), not as part of the final checkpoint. In our diagnostics it overprotects Chinese after exposure has already helped it, whereas pressure-aware weighting reduces this failure.

Dynamic protection controller. The dynamic controller is responsible for adapting the sampling distribution as multilingual learning progresses. Instead of assuming that the weakest language remains fixed throughout training, it continuously estimates the current learning pressure of each language and reallocates training probability accordingly. At each update step t, the controller measures language ℓ's relative weakness by its validation loss deviation from the multilingual mean. Formally, the pressure score is defined as:

$$\rho_\ell^t = \beta \log \frac{L_\ell^t}{\bar{L}^t} + \lambda_s \log s_\ell - \lambda_e p_\ell^t,$$

The first term amplifies the language-specific loss gap; we set $\beta = 2.0$ to make a moderate 10% loss difference translate into a meaningful sampling shift, while avoiding overreaction to early-training noise. The second term, $\lambda_s \log s_\ell$ with s_ℓ as bytes-per-character, is an optional script-complexity prior—we disable it for all final checkpoints ($\lambda_s = 0$) because our ablations show that this static bias over-corrects Chinese in later training and widens the balance gap. The third term, $-\lambda_e p_\ell^t$ with $\lambda_e = 0.5$, acts as a negative feedback that gently penalizes languages already receiving high sampling probability, preventing the controller from chasing loss fluctuations too aggressively.

We then convert ρ_ℓ^t into a target distribution q_ℓ^t via softmax with temperature τ, and smooth the update with momentum η:

$$q_\ell^t = \frac{\exp(\tau \rho_\ell^t)}{\sum_k \exp(\tau \rho_k^t)}, \qquad \tilde{p}_\ell^{t+1} = (1-\eta) p_\ell^t + \eta q_\ell^t.$$

We set $\tau = 2.0$ to produce moderate distributions (avoiding hard one-hot assignments) and $\eta = 0.5$ to balance responsiveness to new signals against

stability across updates. Finally, we clip $\tilde{p}^{t+1}$ to $[p_{\min}, p_{\max}] = [0.18, 0.50]$ and renormalize. The lower bound ensures that no language drops below 18% sampling probability (roughly half of uniform 33%), preventing starvation; the upper bound caps any single language at 50%, enforcing a balancing constraint that aligns with the 10-epoch BabyLM budget and prevents dominance by one language. The curriculum-dynamic schedule used by submission-facing runs is detailed in Algorithm 1.

3 Experimental Setup

We first conduct diagnostic experiments to isolate the four challenges identified by ARMOR-DP before training submission-facing checkpoints. These probes use the official gated BabyBabelLM dataset rows available through the Hugging Face dataset viewer and construct equal-length 128-character windows with an equal number of samples for each language. Unless otherwise stated, the focused diagnostic probe uses 180 windows per language, sequence length 64, a 2-layer decoder-only Transformer with hidden size 56 and four attention heads, and is trained for 240 steps over five random seeds. These deliberately short runs isolate mechanisms; they are not used as estimates of full-scale language competence. All benchmark claims come from the 1M- and 100M-budget checkpoints described below. We report the mean validation loss across languages together with the balance gap

$$\Delta = \max_{\ell} L_{\ell} - \min_{\ell} L_{\ell},$$

which measures cross-linguistic imbalance. These diagnostic experiments are intended to identify the dominant source of multilingual weakness and guide component selection rather than serve as the final benchmark evaluation.

Based on the diagnostic analysis, we train submission-facing checkpoints by instantiating the three most effective ARMOR-DP components, namely byte-level representation, exposure control, and the dynamic sampling controller. The objective layer is retained as an optional diagnostic module and is evaluated separately through ablation rather than included in the final submission. For byte-level representation, all Unicode text is encoded using standard UTF-8 byte sequences, where each byte is treated as an individual token without language-specific preprocessing or segmentation. Byte values are mapped to training ids by adding an offset of two to reserve ids 0 and 1 for padding and EOS tokens. This mapping is shared throughout training and Hugging Face export, ensuring complete consistency between training and official evaluation.

For the 100M frontier, we train curriculum-dynamic checkpoints with a context length of 512 at model sizes d64, d256, d512, and d768, together with a matched d512 static ZH50 baseline. Training metadata records the actual number of non-padding predicted tokens, wall-clock throughput, per-language consumed epochs, and the maximum consumed epoch, confirming that every reported competition candidate remains below the official ten-epoch limit.

We evaluate all checkpoints using the official BabyLM public zero-shot and fine-tuning pipelines. Before full evaluation, interface compatibility is verified using approximately 100k predicted tokens, 300 official rows per language, training sequence length 128, and exported context length 512. The released equal-trilingual GPT-2 baseline from the official evaluation README is included as a public horizontal reference (BabyLM Organizers, 2026c). Public-suite results are used for model selection because leaderboard-only tasks become available only after submission. The selected d768 checkpoint is finally submitted to the official leaderboard, and the returned multilingual score is reported separately.

Table 1 maps the checkpoint names used throughout the results to their active components.

4 Results

4.1 Balanced input is not balanced pressure

Figure 1 compresses the first failure mode behind ARMOR-DP: a balanced multilingual budget is not automatically a balanced learning pressure. With the same 23,040-character budget per language, Chinese contains 66,228 UTF-8 bytes and 1,535 unique characters, while English and Dutch each have about 23,000 bytes and fewer than 60 unique characters.

This pressure is visible in Table 2. A character-level balanced baseline collapses on Chinese, with a mean loss of 3.066 and a balance gap of 1.858. Switching only the representation to byte units reduces mean loss to 2.362 and gap to 0.185. This is the clearest evidence for the first ARMOR-DP layer.

System family	Representation	Capacity	Exposure	Controller	Objective
Balanced adapters	UTF-8 bytes	Adapters	Balanced	None	Token mean
Static ZH50	UTF-8 bytes	Adapters	25/25/50	None	Token mean
Curriculum-dynamic	UTF-8 bytes	Adapters	Balanced→adaptive	Loss-based	Token mean
Dynamic + pressure	UTF-8 bytes	Adapters	Adaptive	Loss-based	Pressure-aware
Curr.-dyn. + pressure	UTF-8 bytes	Adapters	Balanced→adaptive	Loss-based	Pressure-aware

Table 1: Component map for the checkpoint families reported in Sections 4–7. The d256, d512, and d768 final systems are capacity-scaled instances of curriculum-dynamic.

Condition	Mean loss	Gap	EN	NL	ZH
Char balanced baseline	3.066 ± 0.020	1.858 ± 0.051	2.509	2.415	4.273
Byte balanced baseline	2.362 ± 0.007	0.185 ± 0.063	2.455	2.361	2.269
Byte + ZH50 baseline	2.317 ± 0.012	0.548 ± 0.042	2.531	2.435	1.983
Byte + ZH50 adapters	2.235 ± 0.007	0.511 ± 0.028	2.436	2.344	1.926
Byte balanced→ZH50 adapters	2.246 ± 0.010	0.408 ± 0.022	2.417	2.312	2.008
Byte + ZH50 routing gate	2.272 ± 0.014	0.459 ± 0.051	2.457	2.361	1.998
Byte + ZH50 adapters + rare loss	2.276 ± 0.007	0.663 ± 0.032	2.532	2.425	1.869

Table 2: Controlled ARMOR-DP probe on official BabyBabelLM rows. Representation armor removes the Chinese collapse; exposure and capacity improve mean loss but can transfer worst-language status to English.

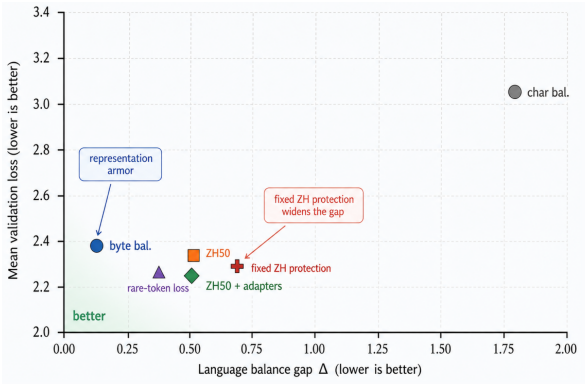


Figure 2: End-of-run score/balance tradeoff across diagnostic systems. The axes are final mean validation loss and final language gap (not training steps); lower is better on both axes. Byte representation removes the first collapse, while ZH-protective adapters improve mean loss but widen the language gap, making English the weakest language.

4.2 Protection moves the bottleneck

Exposure and capacity armor improve the average score but change which language needs protection. Figure 2 makes this tradeoff visible without requiring the reader to scan every loss column in Table 2. Byte + ZH50 adapters obtains the lowest mean loss in the controlled probe, but the worst language is English in all seeds. This transfer is not a failure of the framework; it is the phenomenon the framework should explain. A multilingual BabyLM system should not protect a fixed language. It should protect the language currently under the highest developmental pressure.

4.3 Dynamic control turns movement into signal

We therefore ran a dynamic probe. Every 24 steps, the controller evaluates per-language losses and updates sampling ratios with capped adaptive protection. The pure dynamic adapter system keeps the balance gap close to balanced training while slightly lowering mean loss. A script-prior variant pushes more mass to Chinese and achieves lower mean loss than pure dynamic control, but at a larger gap. Static ZH50 remains the lowest mean-loss configuration in this small run, but has the largest gap among the adapter systems.

The logged update points also provide a direct trace of weakest-language movement. In the objective probe, the weakest language often begins as Dutch, moves to Chinese during early byte-level learning, and then transfers to English after Chinese-protective exposure has taken effect. Across three seeds, the adaptive exposure controller shows two NL→ZH transitions and three ZH→EN transitions. This supports the main claim that "weakest language" is a training-time state, not a fixed label.

4.4 Objectives must not repeat exposure

Before comparing objective variants, we distinguish genuine pressure weighting from a reduction artifact: averaging losses by language already equalizes languages even when explicit pressure weights are weak.

Table 4 compares objective variants under byte-level representation and language adapters. Naive

Condition	Mean loss	Gap
Balanced adapters	2.230	0.179
Dynamic-loss adapters	2.229	0.199
Dynamic-loss routing gate	2.264	0.168
Dynamic + script prior adapters	2.217	0.290
Curriculum adapters	2.216	0.353
Static ZH50 adapters	2.198	0.489

Table 3: Dynamic protection probe over three seeds. Dynamic loss control is most useful as a balance-preserving controller; static ZH50 still optimizes mean loss at the cost of English.

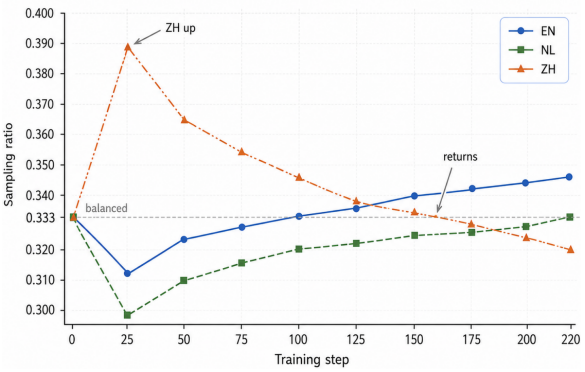


Figure 3: How the dynamic controller changes exposure. Each curve is the sampling probability for one language, and the x-axis is the logged controller-update index from a dynamic probe (unlike the end-of-run axes in Figure 2). The controller first raises Chinese exposure, then returns probability mass to English and Dutch as Chinese becomes easier, showing why fixed ZH protection can become stale.

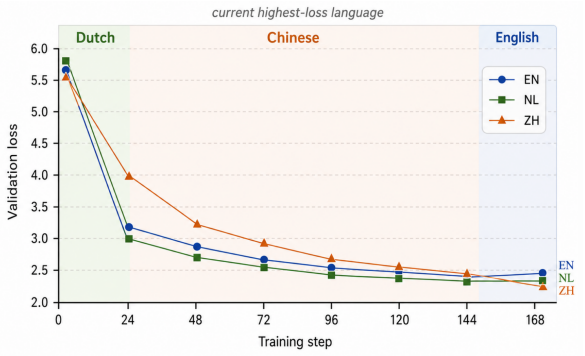


Figure 4: Weakest-language identity over training. The highlighted language at each step is the current highest-loss language. The bottleneck moves from Dutch to Chinese and then to English, supporting the claim that weakness is a training-time state rather than a fixed language label.

Condition	Mean loss	Gap	Worst
Adaptive exp.	2.344	0.118	EN
Balanced	2.346	0.119	EN
Adaptive exp. + pressure	2.360	0.158	EN
ZH50 + pressure	2.354	0.232	EN
ZH50	2.320	0.348	EN
ZH50 + rare token	2.359	0.481	EN

Table 4: Objective-layer probe over three seeds. Pressure-aware weighting is substantially less damaging than rare-token weighting and reduces the over-protection gap under ZH50, but adaptive exposure remains the better balance-preserving controller.

rare-token weighting is the least attractive objective in this confirmation run: it lowers Chinese loss but widens the final gap by making English much worse. Pressure-aware weighting is more controlled. Under a fixed ZH50 exposure schedule, it reduces the gap from 0.348 to 0.232 with only a modest mean-loss cost. However, it still does not beat the balance-preserving adaptive exposure baseline. We therefore treat objective armor as necessary but not solved: the evidence favors a pressure-aware objective, while the final system should tune its strength so it does not duplicate exposure protection.

We later found that objective design has one additional pitfall. A language-level pressure loss can improve balance simply by changing the reduction unit from token means to language means, even when pressure weights are weak. To separate the objective layer from this hidden equalization effect, we added a token-level pressure reduction with warmup, end decay, and a capped maximum weight of 1.4. With strength 0.20, this scheduled objective reduces the local 1M validation gap from 0.253 to 0.240 while keeping mean loss essentially unchanged (2.264 to 2.262). On the public zero-shot suite, it reduces the curriculum-dynamic language gap from 0.0359 to 0.0330, but lowers overall accuracy from 0.4634 to 0.4612. We therefore keep curriculum-dynamic as the scorer and treat scheduled token pressure as evidence for the objective layer rather than as the final checkpoint.

4.5 Public evaluation separates score and diagnosis

Table 5 reports a full pass through the official multilingual zero-shot task groups for the original five 1M-token checkpoints plus the best scheduled token-pressure variant. These are not interface smoke tests: each checkpoint is loaded through the HuggingFace causal-LM path and evaluated with the official English, Dutch, and Chinese `lm-eval` task definitions. Hidden leaderboard tasks are not available locally, so the table should be read as

public-suite evidence.

The official score changes the model-selection story. Static ZH50 remains the best local language-modeling-loss checkpoint, but it is not the strongest zero-shot system. Curriculum-dynamic protection obtains the best mean score by a small margin over balanced byte adapters, and dynamic+pressure obtains the smallest language gap and variance while sacrificing mean accuracy. This is the desired ARMOR-DP separation: the competition-facing scorer, the balance-preserving controller, and the diagnostic pressure objective need not be the same checkpoint.

We evaluate all checkpoints using the official multilingual fine-tuning pipeline over 22 public task-language pairs. Since fine-tuning losses are not directly comparable across task-specific classifiers, Table 6 reports the official accuracy metric. Curriculum-dynamic achieves the highest fine-tuning mean and the best average of zero-shot and fine-tuning performance, primarily driven by improved Chinese accuracy, which increases from 0.210 under balanced adapters and 0.207 under static ZH50 to 0.244. Pressure-aware variants instead prioritize cross-lingual balance, with curriculum-dynamic+pressure achieving the smallest fine-tuning gap while shifting the weakest language from Chinese to Dutch.

4.6 Focused 1M grid

The next local experiment scales the most relevant systems from the 100k-token interface probe to 1M predicted tokens, using 600 cached official rows per language. Table 7 shows the 1M local validation losses and the actual consumed epochs logged by the training sampler. All runs remain below the 10-epoch competition limit. Static ZH50 obtains the lowest mean loss but does so by driving Chinese loss far below English and Dutch, making English the weakest language and widening the gap to 0.389. Curriculum-dynamic protection gives up 0.015 mean loss while reducing the gap to 0.253. Balanced byte adapters are also highly competitive, which reinforces the representation claim: script-fair representation is a prerequisite before exposure control can be interpreted.

Although dynamic+pressure and curriculum-dynamic+pressure improve cross-lingual balance, they incur a reduction in the overall mean score compared with curriculum-dynamic alone. We therefore retain pressure-aware weighting as an optional fourth-layer mechanism for diagnostic analysis rather than including it in the final submission, suggesting that its effectiveness depends on appropriate strength scheduling.

4.7 Capacity scaling makes dynamic protection competitive

The official public suite requires an interface that can tolerate long zero-shot contexts. We therefore closed a 512-context line after the 1M public evaluation. A first budget curve with the tiny d64 model showed that simply training longer was not enough: the best 100M d64 curriculum-dynamic run reached only 0.333 combined. A substantial improvement came from capacity armor. Increasing the same curriculum-dynamic system to d256, d512, and d768 raises the public combined score to 0.371, 0.381, and 0.387 respectively.

This changes the ARMOR-DP submission decision. The 100M d768 checkpoint is the strongest ARMOR-DP public scorer and also has a competitive public language gap, while the d512 checkpoint provides the clearest matched-capacity comparison against static ZH50. A matched d512 static ZH50 baseline falls to 0.359 combined, and its local language-modeling gap is much larger: 0.495 versus 0.079 for d512 curriculum-dynamic. Static ZH50 also consumes 9.78 Chinese epochs, close to the 10-epoch competition limit, whereas curriculum-dynamic remains below 6.61 epochs for every language. Thus ARMOR-DP's dynamic exposure is no longer only diagnostic at the final scale; it is also the stronger public-suite system under matched capacity. The released equal-trilingual GPT-2 baseline still sets a higher public combined reference (0.410 versus 0.387), mainly through zero-shot accuracy, while d768 is comparable on fine-tuning. This makes the current award-facing claim sharper: ARMOR-DP explains and improves the dynamic-protection frontier, but it should not be described as a finished leaderboard winner before hidden evaluation. The organizer GPT-2 uses a conventional subword tokenizer, whereas ARMOR-DP uses bytes; a token therefore does not represent the same amount of linguistic content in the two systems. We include GPT-2 only as an external score reference, not as a controlled tokenizer comparison. Figure 5 turns this model-selection result into the frontier view used for the final submission decision: moving from d64 to larger curriculum-dynamic models improves score without returning to the large local gap of static ZH50.

System	Overall	Worst	Gap	Var.	EN	NL	ZH
Balanced adapters	0.4633	NL 0.4465	0.0334	0.00019	0.4798	0.4465	0.4635
Static ZH50	0.4574	NL 0.4413	0.0379	0.00025	0.4791	0.4413	0.4519
Curriculum-dynamic	**0.4634**	NL 0.4439	0.0359	0.00022	0.4798	0.4439	**0.4665**
Curr.-dyn. + token pressure	0.4612	NL 0.4443	0.0330	0.00018	0.4772	0.4443	0.4620
Dynamic + pressure	0.4541	NL 0.4443	**0.0181**	**0.00006**	0.4623	0.4443	0.4558
Curriculum-dynamic + pressure	0.4554	NL 0.4438	0.0292	0.00016	0.4730	0.4438	0.4495

Table 5: Full public official zero-shot evaluation for 1M-token checkpoints. Overall is the mean over English, Dutch, and Chinese task-group means. Gap and variance are computed over language means, so lower is more balanced.

System	Fine	Combined	Worst	Gap	Var.	EN	NL	ZH
Balanced adapters	0.2326	0.3479	ZH 0.2099	0.0485	0.00040	0.2584	0.2295	0.2099
Static ZH50	0.2254	0.3414	ZH 0.2072	0.0391	0.00026	0.2464	0.2226	0.2072
Curriculum-dynamic	**0.2395**	**0.3514**	NL 0.2180	0.0388	0.00026	0.2567	0.2180	**0.2437**
Dynamic + pressure	0.2228	0.3385	ZH 0.2094	0.0337	0.00021	0.2159	**0.2431**	0.2094
Curriculum-dynamic + pressure	0.2364	0.3459	NL 0.2304	**0.0143**	**0.00004**	0.2447	0.2304	0.2342

Table 6: Full public official fine-tuning evaluation. Fine is the mean over language-level task means. Combined is the unweighted average of the zero-shot overall score in Table 5 and the fine-tuning mean here. Gap and variance are computed over fine-tuning language means.

Final leaderboard submission. We submit the d768 curriculum-dynamic checkpoint as the final challenge-facing ARMOR-DP system. The official leaderboard accepts the submission as challenge eligible and returns a multilingual average of 38.32. The language averages are 35.34 for English, 40.96 for Dutch, and 38.65 for Chinese. This result should be read as a final submission check rather than a new model-selection signal: all system choices above were made from public official evaluations, and the leaderboard result confirms the submitted checkpoint while preserving the paper's main claim as diagnostic rather than leaderboard dominance.

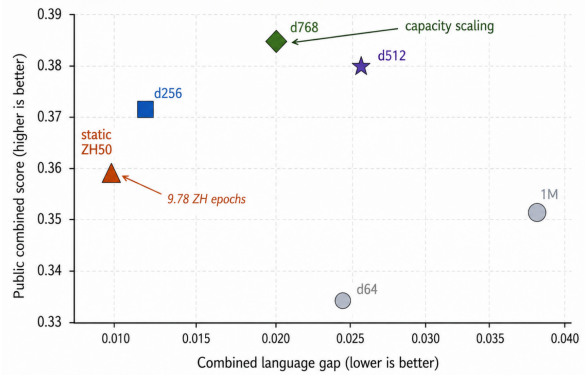


Figure 5: Internal ARMOR-DP public score/balance frontier for the 100M line. The released GPT-2 equal-trilingual baseline is higher (0.4097 combined) and is reported in Table 8, but is not drawn because it is an external organizer baseline rather than an ARMOR-DP variant. Within the internal frontier, capacity scaling turns curriculum-dynamic protection into the best-performing ARMOR-DP scorer, while the matched static ZH50 baseline sits lower despite a small public gap.

5 Related Work

ARMOR-DP builds on three lines of work. First, BabyLM studies sample-efficient pretraining under developmentally plausible data budgets (BabyLM Organizers, 2026b,a; Warstadt et al., 2025). Our setting adds the multilingual constraint that a fixed token or character budget can interact with script and morphology. Second, token-free and byte-level models show that character and byte units can reduce tokenizer-specific assumptions (Xue et al., 2022; Clark et al., 2022; Tay et al., 2022); here we use bytes as a diagnostic representation, not as a claim that bytes are universally optimal. Third, multilingual sampling, adapters, curriculum learning, sparse routing, and loss reweighting provide tools for allocating exposure, capacity, and gradient pressure (Conneau et al., 2020; Houlsby et al., 2019; Pfeiffer et al., 2020; Bengio et al., 2009; Fedus et al., 2022; Lin et al., 2017). ARMOR-DP's contribution is to combine these tools around the moving identity of the weakest language.

6 Discussion

Our experiments show that multilingual imbalance is an evolving training state: the dominant weakness shifts from representation to exposure and then capacity, so static language protection becomes progressively less effective. ARMOR-DP separates

System	Mean	Gap	Worst	EN loss	NL loss	ZH loss	Max ep.
Balanced adapters	2.258	0.226	EN	2.372	2.258	2.145	4.410
Static ZH50	**2.248**	0.389	EN	2.414	2.305	2.025	6.555
Curriculum-dynamic	2.264	0.253	EN	2.389	2.266	2.136	4.523
Dynamic + pressure	2.295	0.337	EN	2.438	2.345	2.101	4.387
Curriculum-dynamic + pressure	2.279	0.273	EN	2.407	2.298	2.133	4.505

Table 7: Focused 1M-token grid on cached official rows. The epoch column is the maximum actual consumed epoch over English, Dutch, and Chinese, recorded in each checkpoint's training metadata.

System	Non-pad tok.	Zero	Fine	Comb.	Worst comb.	Gap
1M curr.-dyn.	1.0M	0.4634	0.2395	0.3514	NL 0.3309	0.0373
100M d64 curr.-dyn.	87.3M	0.4692	0.1976	0.3334	NL 0.3202	0.0248
100M d256 curr.-dyn.	87.3M	0.4842	0.2587	0.3714	NL 0.3663	**0.0118**
100M d512 curr.-dyn.	87.0M	0.4954	0.2659	0.3807	NL 0.3667	0.0256
100M d768 curr.-dyn.	87.0M	0.5032	**0.2699**	0.3866	NL 0.3734	0.0200
100M d512 static ZH50	89.0M	0.4845	0.2330	0.3587	EN 0.3538	0.0095
Official GPT-2 tri. baseline	–	**0.5516**	0.2678	**0.4097**	ZH **0.3929**	0.0306

Table 8: Public official capacity frontier. "Non-pad tok." counts actual predicted non-padding tokens; all ARMOR-DP rows remain under the 10-epoch competition cap. The d512 curr.-dyn. and d512 static rows are matched-capacity systems. The GPT-2 row is the released equal-trilingual baseline from the official evaluation README; its subword tokens are not content-equivalent to ARMOR-DP's byte tokens.

these factors and their interventions, allowing adaptation to follow the current learning state rather than a predetermined schedule.

For the BabyLM challenge, byte-level representation, exposure control, and the dynamic controller consistently improve multilingual performance and form our final system. The objective layer is optional: pressure-aware weighting reduces cross-lingual gaps but modestly lowers the mean score, suggesting that its strength should be scheduled rather than applied unconditionally. This modular separation also makes interactions and future combinations easier to analyze.

7 Submission Choice

The final submission is the curriculum-dynamic d768 checkpoint, which has the best overall public performance while maintaining competitive cross-lingual balance. We retain the matched d512 checkpoint as the principal diagnostic control against static ZH50.

8 Conclusion

ARMOR-DP reframes multilingual BabyLM as protection of the language currently under greatest developmental pressure. Across diagnostics, 1M probes, and capacity scaling, representation establishes a script-fair foundation, exposure and capacity improve performance, and dynamic control tracks bottleneck transitions that fixed protection misses. By separating diagnosis from modular, empirically instantiable interventions, ARMOR-DP offers a flexible basis for future multilingual training.

Limitations

The experiments use small diagnostic models and cached subsets of official dataset rows. The public official suite is complete for the checkpoints reported here, and we include the released GPT-2 equal-trilingual public baseline for orientation. Leaderboard-only tasks were not available during model selection; they were scored only after the final d768 checkpoint was submitted. We therefore treat the official leaderboard average as a post-selection submission result, not as a tuning signal. The 100M-budget runs use a 100M maximum-token training budget, but actual predicted non-padding tokens are lower because many cached examples are shorter than 512 tokens. The 100M capacity frontier is single-seed because of compute cost; diagnostic claims are supported by multi-seed small probes, but large-scale score variance remains unmeasured. The controller is evaluated on only three languages, and a larger language set may produce noisier pressure signals and require different update bounds. We also report one controller configuration rather than a full sensitivity sweep. The script-complexity prior was the most brittle component: it over-corrected Chinese later

in training and was disabled in final runs. A scheduled prior that decays after early training remains an open direction.

References

BabyLM Organizers. BabyBabelLM: Multilingual developmentally plausible language modeling. https://babylm.github.io/babybabellm/, 2026a. Accessed 2026-05-06.

BabyLM Organizers. BabyLM challenge guidelines. https://babylm.github.io/guidelines.html, 2026b. Accessed 2026-05-06.

BabyLM Organizers. BabyLM evaluation pipeline. https://github.com/babylm-org/babylm-eval, 2026c. Accessed 2026-05-10.

Yoshua Bengio, Jerome Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 41–48, 2009.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. CANINE: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91, 2022.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, 2020.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799, 2019.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7654–7673, 2020.

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. Charformer: Fast character transformers via gradient-based subword tokenization. In *International Conference on Learning Representations*, 2022.

Soroush Vosoughi and Deb K. Roy. A longitudinal study of prosodic exaggeration in child-directed speech. In *Proceedings of Speech Prosody 2012*, pages 194–197, 2012. doi: 10.21437/SpeechProsody.2012-50.

Alex Warstadt et al. The third BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge*, 2025.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022.

A Reproducibility Details

The main paper keeps the eight-page submission concise. This appendix records the implementation details that are most likely to matter for reproducing or auditing the ARMOR-DP claim.

Data and tokenization. All controlled diagnostics use English, Dutch, and Chinese rows from the BabyBabelLM public data interface. We form equal-length windows per language for the diagnostic probes so that representation and exposure interventions can be compared under matched surface budgets. Submission-facing checkpoints use a UTF-8 byte tokenizer with the same id mapping in training and HuggingFace export. Padding labels are masked from the language-modeling loss, and all reported token counts are predicted non-padding tokens rather than padded sequence slots.

Evaluation. The public official suite is run through the HuggingFace causal-LM loading path for zero-shot tasks and through the official multilingual fine-tuning pipeline for supervised tasks. Hidden leaderboard tasks are not locally available. The fine-tuning pass contains 22 public task-language pairs for the 100M capacity frontier, with no missing tasks.

Epoch accounting. The training loop records consumed epochs per language by counting the actual examples sampled for that language. This is stricter than reporting optimizer steps alone: a static ZH50 mixer can keep the global token budget modest while nearly exhausting the Chinese-language epoch cap. All competition-facing rows reported in the main paper and in the ledger below remain below the 10-epoch limit.

System	Tok. (M)	Seq.	Batch	d/layers	Steps	EN ep.	NL ep.	ZH ep.	Wall min
1M balanced adapters	0.994	128	8	64/2	985	4.32	4.40	4.41	–
1M static ZH50	0.996	128	8	64/2	985	3.26	3.32	6.56	–
1M curr.-dyn.	0.994	128	8	64/2	985	4.34	4.28	4.52	–
1M dyn.+pressure	0.994	128	8	64/2	985	4.38	4.37	4.39	–
1M curr.-dyn.+pressure	0.994	128	8	64/2	985	4.34	4.29	4.51	–
5M l512 curr.-dyn.	4.402	512	2	64/2	4,893	1.58	1.59	1.72	–
5M l512 static ZH50	4.520	512	8	64/2	1,224	1.22	1.19	2.49	–
100M d256 curr.-dyn.	87.272	512	8	256/6	24,462	6.46	6.35	6.76	26.8
100M d512 curr.-dyn.	87.037	512	8	512/8	24,462	6.56	6.54	6.46	38.7
100M d768 curr.-dyn.	86.992	512	4	768/12	48,924	6.61	6.59	6.37	108.7
100M d512 static ZH50	89.011	512	8	512/8	24,462	4.91	4.87	9.78	32.7

Table 9: Training-accounting ledger. Epochs are actual consumed epochs for English, Dutch, and Chinese, not estimates from optimizer steps. "Wall min" is reported when available from the remote training metadata.

B Hyperparameters and Controller Settings

Component	Setting	Value
Diagnostic corpus	Windows/language	180 for dynamic and objective probes
Diagnostic model	Architecture	Decoder-only Transformer, d=56, 2 layers, 4 heads
Diagnostic training	Steps/seeds	240 steps for dynamic probe; 180 steps for objective probe; three seeds
Rare-token control	Token weighting	Corpus-frequency derived; rarer bytes weighted more; clipped to 0.25–1.8
1M public checkpoints	Architecture	d=64, 2 layers, 4 heads, language adapters
1M public checkpoints	Training/export context	128-token training context; 512-token exported context
100M capacity frontier	Architecture	d256/6 layers, d512/8 layers, d768/12 layers
100M capacity frontier	Optimizer learning rate	3e-4 for d256/d512; 2e-4 for d768
Static ZH50	Sampling ratio	EN/NL/ZH = 0.25/0.25/0.50
Curriculum-dynamic	Start schedule	Balanced sampling, then Chinese-protective/adaptive sampling
Adaptive controller	Update interval	24 diagnostic; 80 for 1M/5M/d768; 40 for d256/d512
Adaptive controller	Bounds	$p_{\min} = 0.18,\ p_{\max} = 0.50$
Adaptive controller	Loss/smoothing/temp.	$\beta = 2.0,\ \eta = 0.5,\ \tau = 2.0$
Adaptive controller	Exposure penalty	$\lambda_e = 0.5$ for curriculum-dynamic runs
Pressure objective	Weight bounds	0.25–1.8 in diagnostic runs; capped at 1.4 in tuned token-pressure probes

Table 10: Core hyperparameters. All competition-facing runs set `max_epochs=10`; the actual consumed-epoch ledger is in Table 9.

Controller pseudocode. **Algorithm 1: Curriculum-dynamic exposure controller.** Initialize $p = (1/3, 1/3, 1/3)$; train balanced until $T/2$; set $p = (0.25, 0.25, 0.50)$; after $T/2$, every K steps, evaluate L_ℓ^t, compute ρ_ℓ^t and q_ℓ^t with $\beta = 2.0, \eta = 0.5, \tau = 2.0, \lambda_e = 0.5, \lambda_s = 0$, then clip to $[0.18, 0.50]$, renormalize, and continue.

Controller sensitivity. We did not run an exhaustive sensitivity sweep. The clearest sensitivity was to the script prior: enabling $\lambda_s > 0$ could help Chinese early but over-corrected it later, so all final checkpoints use $\lambda_s = 0$. A scheduled, decaying script prior may preserve the early benefit without the late bias. The behavior of the bounds and update rule with substantially more than three languages is left for future work.

C Additional Public Evaluation Details

Table 11 adds the 5M 512-context public suite that is summarized only implicitly in the main paper. At 5M, curriculum-dynamic already beats static ZH50 on combined score, mostly because static ZH50 loses more fine-tuning accuracy. At 100M, the same direction becomes the submission-facing line.

System	Zero	Fine	Comb.	EN comb.	NL comb.	ZH comb.	Worst	Gap
5M l512 curr.-dyn.	0.464	0.216	0.340	0.360	0.332	0.329	ZH 0.329	0.031
5M l512 static ZH50	0.466	0.189	0.328	0.338	0.307	0.338	NL 0.307	0.031
100M d256 curr.-dyn.	0.484	0.259	0.371	0.370	0.366	0.378	NL 0.366	0.012
100M d512 curr.-dyn.	0.495	0.266	0.381	0.392	0.367	0.383	NL 0.367	0.026
100M d768 curr.-dyn.	0.503	0.270	0.387	0.393	0.373	0.393	NL 0.373	0.020
100M d512 static ZH50	0.485	0.233	0.359	0.354	0.363	0.359	EN 0.354	0.009

Table 11: Additional public official combined results. Combined is the unweighted average of zero-shot and fine-tuning means.

Table 12 expands the public fine-tuning tasks for the final 100M d768 curriculum-dynamic scorer. The table is not used to tune claims in the main paper; it is included to make clear which public task groups underlie the reported fine-tuning mean.

Task	EN	NL	ZH
ARC	0.235	0.235	0.262
Belebele	0.193	0.233	0.165
BMLAMA	0.123	0.113	0.130
MNLI	0.436	0.365	0.408
SIB200	0.150	0.305	0.425
TruthfulQA	0.232	0.268	0.214
XNLI	0.436	–	0.427
INCLUDE	–	0.321	0.277

Table 12: Public fine-tuning task accuracies for the 100M d768 curriculum-dynamic checkpoint. Dashes indicate language-task pairs not present in the local public suite.

D Negative Results and Deferred Directions

We preserve the failed or deferred branches because they help distinguish the framework from a post-hoc winning recipe. Static ZH50 is retained as the matched baseline that makes the dynamic comparison meaningful, rather than discarded after the final system is chosen.

Simple gates were not enough. The original gate direction was too weak when treated as a generic attention modification. A language-conditioned routing gate became more interpretable as capacity armor and obtained a reasonable diagnostic tradeoff, but it did not beat adapters as the main scorer. We therefore use adapters in the final competition-facing line and keep routing gates as a capacity diagnostic.

Rare-token weighting over-protects Chinese. Rare-token weighting lowers Chinese loss after Chinese-protective exposure has already helped, but it widens the balance gap by making English the weakest language. In the objective probe, ZH50+rare reaches a gap of 0.481, compared with 0.348 for ZH50 and 0.118 for adaptive exposure. We treat this as a negative result for frequency-only objectives.

Pressure-aware weighting is useful but not final. Pressure-aware weighting aligns better with ARMOR-DP than rare-token weighting and gives the smallest public zero-shot gap in the 1M suite and the smallest public fine-tuning gap with curriculum-dynamic exposure. Because it lowers mean public score in the current tuning range, the final scorer omits it; pressure objectives remain a fourth-layer research direction.

Cross-Linguistic Effects in Bilingual Phoneme BabyLMs

Nikitas Theodoropoulos[1], **Maria Lymperaiou**[2], **Giorgos Filandrianos**[3]

[1]Independent Researcher,
[2]National Technical University of Athens,
[3]Instituto de Telecomunicações, Portugal
Correspondence: nikitastheodorop@gmail.com

Abstract

Cross-linguistic effects are a central topic in bilingual first-language acquisition. Artificial learners can help investigate L1–L2 interactions by enabling controlled comparisons across language combinations and learning conditions. Recent work explores this direction by training bilingual language models under developmentally plausible constraints. However, human and model learners still diverge in fundamental ways, with one major difference being input modality: children learn primarily from spoken input, whereas language models are typically trained on orthographic text. To reduce this gap, researchers have trained models on phonemic representations of speech. In this work, we combine these research directions to train bilingual BabyLMs with phonemic input. We keep English fixed as the L2 and vary the L1 across German, Swedish, Persian, and Basque, selected to represent contrasting combinations of syntactic and phoneme-inventory distance from English. Our results show stronger L1-related variation in grammatical learning trajectories under phonemic than orthographic input, while early lexical differences align with phoneme-inventory similarity.

nikitastheo/bilingual-phone-lms
nikitas-theo/BilingualPhoneLMs

1 Introduction

Research in bilingual first-language acquisition has shown that children exposed to two languages early on develop two largely distinct linguistic systems. However, interactions between these systems can produce cross-linguistic effects (Serratrice, 2013). Such effects are often asymmetric, with the first or more dominant language (L1) influencing the weaker or later-acquired language (L2), particularly at the level of morphosyntax. Despite extensive study of this topic, the exact conditions and strength

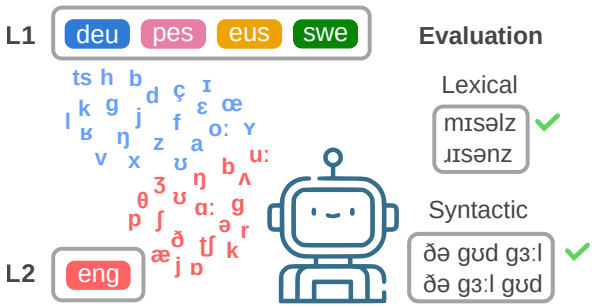


Figure 1: We train matched bilingual orthographic and phoneme LMs with four different L1s and English as the L2. We investigate L1-related differences in English grammatical performance with minimal-pair evaluations. Here, we illustrate the phoneme Syntactic and Lexical metrics from BabySLM (Lavechin et al., 2023).

of cross-linguistic influence remain unclear, with substantial variation across studies even for the same languages and phenomena (Van Dijk et al., 2022). Addressing these questions requires controlled experiments across diverse language pairs and learning conditions.

The BabyLM initiative (Warstadt et al., 2023) supports the study of language acquisition through models trained under developmentally plausible settings that approximate the child's linguistic environment. Such artificial learners enable precise control over input and learning conditions, allowing the simulation of experiments that would otherwise be difficult or impossible with children (Warstadt and Bowman, 2022). Recently this framework has been extended to bilingual acquisition to examine factors such as the effect of bilingualism on learning (Zeng et al., 2026), critical period effects in models and humans (Constantinescu et al., 2025), the role of child-directed speech (Binyamin and Sulem, 2026), and the importance of age of exposure (Issam et al., 2026), among others.

Despite these advances, artificial learners remain limited by their reliance on orthographic input, whereas children acquire language primarily

Proceedings of the Second BabyLM Workshop, pages 320–332
October 29, 2026 ©2026 Association for Computational Linguistics

through speech. This mismatch matters because written and spoken representations encode different linguistic cues, and orthography may introduce patterns unavailable to children. Speech-based initiatives such as BabySLM (Lavechin et al., 2023) and BabyHuBERT (Charlot et al., 2026) offer more ecologically valid input, but suitable child-speech data are scarce and raise privacy and ethical challenges, especially in bilingual settings. Phoneme-based language models provide a practical intermediate approach: they operate on phonemic representations that more closely approximate spoken input while retaining the efficiency, control, and data availability of text-based modeling.

In this work, we investigate whether cross-linguistic effects on L2 grammatical learning differ between bilingual LMs trained on orthographic and phonemic input. We use developmentally plausible multilingual corpora from BabyBabelLM (Jumelet et al., 2026a), to which we apply the G2P+ phonemization tool (Goriely and Buttery, 2025b) to produce aligned phonemic corpora across languages. Specifically, we train matched orthographic and phoneme LMs, keeping English as the L2 and varying the L1 across German, Swedish, Persian, and Basque (Figure 1). These languages represent different combinations of syntactic and phoneme-inventory distance from English, enabling us to examine the relationship between language transfer and specific typological distance measures. Concretely, we seek to answer the following questions: (a) Do cross-linguistic effects manifest differently according to modality (orthographic, phonemic)? (b) Does typological distance (syntactic, phoneme-inventory) correlate with modality-specific effects?

Following prior work on phoneme-based language models (e.g., Bunzeck et al. (2025); Goriely et al. (2024)), we assess cross-linguistic effects through grammatical minimal-pair evaluations, extending our analysis of model behavior throughout training. Our contributions are:

- We introduce a controlled framework for studying cross-linguistic influence in bilingual BabyLMs across orthographic and phonemic input modalities.
- We train matched bilingual models with English as the L2 and four diverse L1s, covering distinct combinations of syntactic and phoneme-inventory distance.
- We provide the first systematic analysis of whether L1-induced effects on L2 grammatical learning change when models are trained on phonemic rather than orthographic input.
- We release the bilingual corpora in orthographic and phonemic form, the data-processing pipeline, trained models, and code to support further research on bilingual and phoneme-based artificial learners.

2 Related work

Cross-linguistic effects Research on bilingual development supports the emergence of independent linguistic systems, while showing that the two languages can influence one another selectively (Paradis and Genesee, 1996; Serratrice, 2013). These effects may be quantitative, facilitating or delaying the acquisition of a linguistic structure, or qualitative, producing patterns not observed in monolingual peers. Their occurrence has been linked to structural overlap and the syntax–pragmatics interface (Hulk and Müller, 2001). A large meta-analysis reports reliable small-to-moderate effects overall, while highlighting substantial variation across studies and identifying significant influence from the societal language to the non-societal language (Van Dijk et al., 2022). Cross-linguistic influence also extends to phonology: Paradis (2001) finds that bilingual children develop separate but non-autonomous phonological systems. Related effects also arise in multilingual language models, where shared training can result in positive or negative interference depending on factors such as data balance, model capacity, and linguistic similarity (Wang et al., 2020; Chang et al., 2024) with syntactic similarity predicting greater benefits. We extend these research directions by investigating cross-linguistic effects in bilingual phoneme LMs and the role of phonemic input in shaping L1–L2 transfer.

Bilingual BabyLMs There has been considerable research effort in modeling bilingual language acquisition with BabyLMs. Zeng et al. (2026) investigate whether bilingualism can negatively or positively influence language learning across various input conditions and scales, concluding no strong effect exists. Issam et al. (2026) vary the onset of exposure to the second language, finding that both language dominance and similarity contribute to cross linguistic effects. Binyamin and Sulem (2026) analyze the effect of child-directed speech

on semantic and syntactic knowledge. In second language acquisition, studies have focused on the effect of typological similarity on morphosyntactic performance (Shen et al., 2024; Oba et al., 2023; Yadavalli et al., 2023), predictive models of non-native human sentence processing (Aoyama and Schneider, 2024), alignment with learner error patterns for different L1 (Gao et al., 2025), interaction-driven continual pre-training Shen et al. (2024), and traces of L1 influence in L2 production (Barbenel et al., 2026). Our work extends these approaches by analyzing L1-induced effects across two input modalities: orthographic and phonemic input.

Phoneme BabyLMs Phonemes have been recently explored as a linguistically grounded input representation for developmentally plausible language models. Grapheme- and phoneme-based BabyLMs can acquire substantial linguistic abilities, with phoneme models approaching grapheme-based performance on various evaluations (Bunzeck et al., 2024, 2025). Goriely et al. (2024) compare orthographic and character-level input with continuous streams of phonemes, finding a modest performance reduction on conventional tasks but advantages for phonological analysis. Goriely and Buttery (2025b) introduce G2P+ and IPA CHILDES, a phonemic child-centered speech resource covering 31 languages and show that phoneme LMs learn phonological features cross-lingually. It is further demonstrated that such models implicitly track word boundaries across languages (Goriely and Buttery, 2025a). We extend this research to bilingual acquisition by comparing L1-induced effects in matched bilingual orthographic and phoneme LMs.

3 Methods

To examine whether the patterns of cross-linguistic influence identified in prior work persist under a more speech-like representation, we compare bilingual orthographic and phoneme LMs. Here, we define a cross-linguistic effect as the difference in model linguistic ability in the L2 that results from the influence of the L1. Given that monolingual children (and models) receive substantially more input in a language compared with bilingual peers, some performance differences are generally expected. Our main goal is thus to compare bilingual learners across different L1s, measuring the influence of input modality and typological distance on the acquisition of L2, while keeping monolingual models as an informative baseline.

Orthographic LMs receive BPE-tokenized written input, whereas the phoneme LMs receive sequences of individual phoneme tokens. We adopt a controlled experimental design in which the same L1–L2 language pairs are used across both conditions. The transformer architecture, source data, training schedule, and optimization settings are held constant where possible, while the modality-specific tokenizers result in different vocabulary and embedding sizes. This setup allows us to isolate how input modality affects the influence of the L1 on L2 learning. We operationalize the L1–L2 relationship using two metrics: syntactic and phoneme-inventory distance. Concretely, we seek to uncover interactions between L1–L2 typological distance and L2 grammatical performance.

3.1 Simulating bilingual acquisition

To model bilingual acquisition, we train GPT-2 models under two matched conditions: orthographic input and phonemic input. We refer to the resulting systems as **orthographic LMs** and **phoneme LMs**, respectively. The models share the same transformer architecture, with 85M non-embedding parameters, while the embedding layers differ in size because of the substantially different vocabularies: approximately 500k parameters for the phoneme and 23M for the orthographic LMs. As a first study of bilingual phoneme models, we opted to compare with BPE-trained orthographic models that constitute the prevailing bilingual learners used in computational works. Grapheme LMs (e.g., Bunzeck et al. (2024); Goriely et al. (2024)) trained on individual characters provide a better matched condition with respect to model capacity but are more rare in the literature, and no grapheme-based bilingual BabyLMs exist to date. This would make it hard to compare our models with prior work. Our framework can readily accommodate such learners in future research experiments.

In the phonemic condition, each phoneme is represented by an individual phoneme token, with dedicated tokens marking word and utterance boundaries. The phoneme-token vocabulary is constructed from the phonemic symbols observed in the corresponding bilingual training corpus. In the orthographic condition, we train a separate BPE tokenizer for each language pair, with a vocabulary of 30,000 tokens. For the phoneme LMs, the vocabulary consists of phonemes occurring more than 10 times in the corresponding bilingual corpus.

We draw the training data from BabyBabelLM (Jumelet et al., 2026a), a multilingual corpus designed for developmentally plausible language modeling. For each bilingual setting, we sample 10M words from the L1 and 10M words from English, for a total of 20M words. To ensure comparability across languages, sampling is calibrated using byte premiums, which account for language-specific differences in the relationship between word counts and encoded data size (Arnett et al., 2024), resulting in content-matched training data. Details about data distributions across languages are given in Appendix C. We then phonemize the same corpora with G2P+ (Goriely and Buttery, 2025b), a tool developed as a unified phonemization backend for languages in CHILDES, with aligned phonemic inventories across languages. This way, we produce paired orthographic and phonemic datasets from identical source utterances.

Following prior work on bilingual LMs (Arnett et al., 2025; Constantinescu et al., 2025), we adopt a sequential-interleaved training regime. Models are first exposed exclusively to the L1 for 10k steps, after which they receive interleaved L1 and English input for a further 20k steps. Thus, our models correspond to early sequential bilinguals. We adopt the training configuration of the Goldfish model suite (Chang et al., 2026); full hyperparameters are reported in App. A. For each language combination, we train two bilingual LMs: one orthographic and one phoneme. As monolingual references, we additionally train one English orthographic and one English phoneme LM. For training, we use the 10M English portion in the bilingual corpora, which we supplement with 10M additional words from BabyBabelLM. These models are trained for 30k steps on the combined 20M corpus, matching the total L1–L2 exposure and data of the bilingual learners.

To increase the robustness of our results, for each model we report performance across 3 different random seeds. For each seed, we re-sample and phonemize all language data from BabyBabelLM to 10M words–leading to distinct but comparable corpora–and train orthographic and phoneme LMs. In the following results, when possible along with the mean over seeds we report standard deviation.

3.2 Language selection

Language selection is central to our design because the effect of an L1 on English may depend on typological similarity. Syntactic measures of similarity have been shown to be predictive of transfer effects (Chang et al., 2024). Given that we evaluate on grammatical benchmarks, we also expect an influence in our models. We additionally consider phoneme-inventory distance to capture differences in the phonemic inventories of the two languages. Therefore, we select L1s that vary along these two dimensions, enabling us to examine whether the patterns observed under orthographic and phonemic input are associated with typological measures.

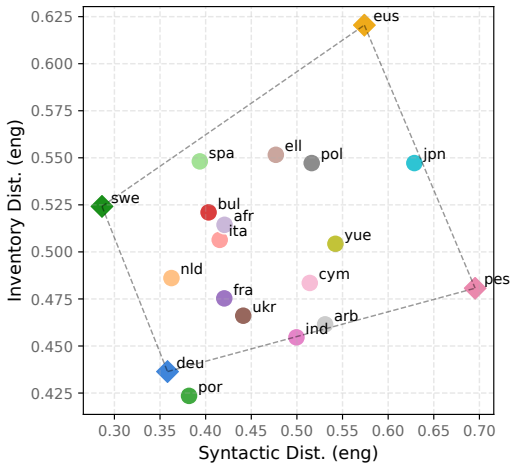


Figure 2: Syntactic and phoneme-inventory distances from English for the eligible language set, computed with URIEL+ (Khan et al., 2025). Languages selected for training the bilingual models are marked with ◇. We opt for languages that occupy contrasting regions of the syntactic and phoneme-inventory distance space.

We keep English fixed as the L2 to facilitate comparison with prior work and because suitable evaluation resources are more widely available for English in both orthographic and phonemic form. Candidate L1s are restricted to languages in BabyBabelLM with at least 10M words of training data, ensuring that all models can be trained under the same developmentally plausible data constraints.

For each candidate language, we compute its syntactic and phoneme-inventory distance from English using URIEL+ (Khan et al., 2025). Syntactic distance captures differences in typological syntactic grammatical features, whereas phoneme-inventory distance reflects differences between the sets of phonemes used by the two languages[1]. These complementary measures are used to characterize syntactic and phoneme-inventory similarity between each L1 and English respectively. Distances are illustrated in Figure 2.

[1]URIEL+ also provides morphological and phonological distance measures, but their coverage was insufficient for the full candidate set.

We select German (deu), Swedish (swe), Persian (pes), and Basque (eus) to represent four contrasting regions of the resulting two-dimensional space: low syntactic and low phoneme-inventory distance for German, low syntactic and high phoneme-inventory distance for Swedish, high syntactic and low phoneme-inventory distance for Persian, and high syntactic and high phoneme-inventory distance for Basque. This contrastive selection samples four regions of the distance space, allowing us to explore whether performance patterns align with syntactic or phoneme-inventory similarity. We hypothesize that syntactic distance will be more closely associated with grammatical evaluation performance, whereas phoneme-inventory distance will be more informative for phonologically oriented evaluations and may exert a stronger influence under phonemic input.

3.3 Evaluation

We use minimal-pair tasks to assess English grammatical learning, targeted grammatical knowledge in both languages, and the effect of input modality. A model receives credit when it assigns higher score to the grammatical sentence, where sentence scores are computed as the sum of token log probabilities. We use the latest BabyLM evaluation pipeline [2] to evaluate on three benchmarks.

BLiMP introduced by Warstadt et al. (2020), comprises 67 minimal-pair datasets across syntactic, morphological, and semantic phenomena. Since English is fixed as the L2, variation across L1 conditions provides our main measure of L1-related effects on L2 grammatical learning. We additionally use MultiBLiMP (Jumelet et al., 2026b), which covers two subject–verb agreement phenomena in 101 languages, as a targeted check of grammatical learning in both English and each L1.

For the phoneme LMs, we convert both benchmarks into phonemic representations using G2P+ (Goriely and Buttery, 2025b), following the same preprocessing used for the training data. We also report the BabySLM syntactic and lexical evaluations (Lavechin et al., 2023): the former tests six simple grammatical phenomena, while the latter contrasts real words with phonotactically plausible pseudowords and provides a targeted subword diagnostic for phoneme LMs.

[2] https://github.com/babylm-org/babylm-eval

			mBLiMP		BabySLM	
	L1	BLiMP	L1	eng	Lexical	Syntactic
Orthographic	eng	69.1	–	83.4	–	88.2
	deu	66.2	84.7	78.8	–	83.8
	eus	63.2	87.2	76.8	–	79.7
	pes	65.2	84.6	78.8	–	83.1
	swe	65.4	–	77.5	–	83.5
Phoneme	eng	66.4	–	86.4	88.6	89.0
	deu	62.6	84.4	77.4	86.8	76.1
	eus	60.8	93.0	74.6	87.3	70.7
	pes	60.9	87.0	74.0	87.1	69.5
	swe	62.6	–	77.8	87.1	74.9

Table 1: Minimal-pair accuracy for the English monolingual references and bilingual LMs trained on orthographic or phonemic input. Results are averaged across 3 different seeds. BLiMP and MultiBLiMP (mBLiMP) are evaluated in their original orthographic form for the orthographic LMs and in phonemic form for the phoneme LMs. BabySLM Lexical is available only for the phoneme condition.

4 Results

4.1 Minimal-Pair Evaluation

Table 1 reports minimal-pair accuracy for the bilingual orthographic and phoneme LMs across all L1 conditions, together with the corresponding English monolingual references. We first examine each input condition separately and then compare the extent and direction of L1-related variation across modalities. Overall, performance across seeds stays consistent, with occasional variations.

Orthographic LMs The English monolingual reference achieves the highest score on all three applicable evaluations: BLiMP, English MultiBLiMP, and BabySLM Syntactic. The bilingual models nevertheless perform above chance on BLiMP and on the English and L1 portions of MultiBLiMP, indicating that they acquire at least some of the evaluated grammatical contrasts in both languages.

Based on prior findings that syntactic similarity can facilitate multilingual transfer (Chang et al., 2024), we expected the German–English and Swedish–English models, whose L1s are syntactically closer to English, to achieve higher English grammatical performance. We observe some evidence for this pattern in the orthographic condition. German and Swedish obtain relatively high scores on BLiMP and BabySLM Syntactic, while the more syntactically distant Basque model performs lowest on both benchmarks. However, the pattern is not fully consistent: Persian performs

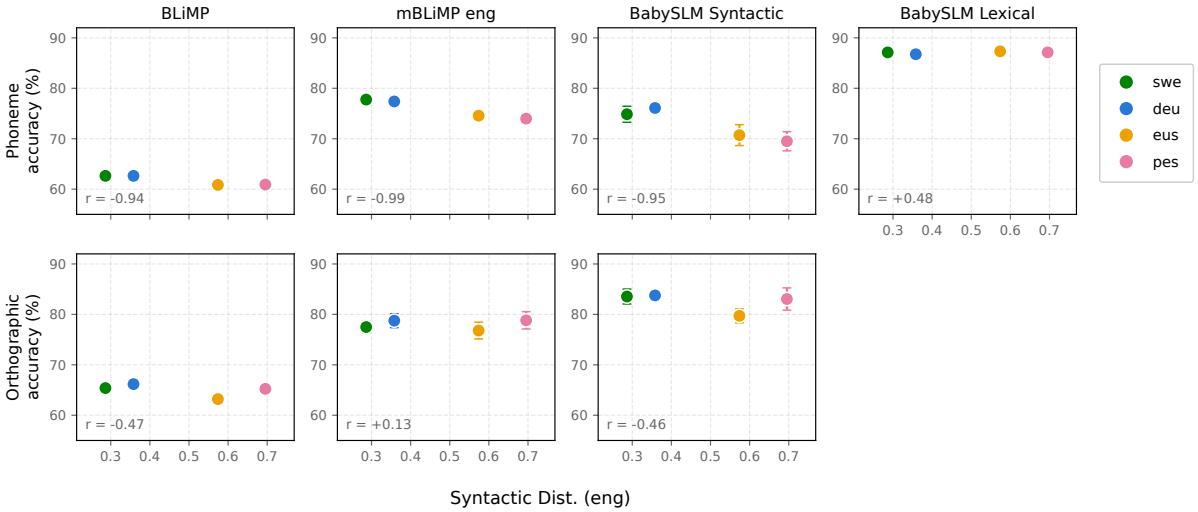


Figure 3: English evaluation accuracy as a function of L1–English syntactic distance for phoneme (top) and orthographic (bottom) LMs. For each model we report an average across 3 seeds with SD error bars. Pearson correlation (r) over averages shown for each benchmark. BabySLM Lexical is available only in phonemic form.

similarly to German and Swedish on BabySLM Syntactic and BLiMP, and German and Persian obtain the same rounded score on English MultiBLiMP. Accordingly, syntactic distance shows only a weak and benchmark-dependent association with grammatical performance for orthographic LMs. This is broadly consistent with Zeng et al. (2026), who find no large or uniform effects of bilingual training under orthographic input.

Phoneme LMs The English reference achieves the highest absolute performance across all applicable evaluations. Importantly, we notice it also surpasses the matched orthographic English LM in two out of the three common evaluations (BabySLM Syntactic, mBLiMP eng), consistent with previous findings of increased performance for phoneme LMs in some cases (Bunzeck et al., 2025; Goriely et al., 2024). For bilingual LMs we notice overall matched or reduced performance when compared with orthographic counterparts.

However, a clearer L1-related ordering emerges across the grammatical benchmarks for bilingual phoneme LMs. German and Swedish, the two L1s with lower syntactic distance from English, consistently outperform Persian and Basque on BLiMP, English MultiBLiMP, and BabySLM Syntactic.

The ordering among bilingual models is more consistent across the grammatical evaluations. German achieves the highest BabySLM Syntactic scores, while German and Swedish obtain similar scores for English BLiMP and MultiBLiMP. Persian and Basque generally obtain lower grammatical scores. This pattern aligns with syntactic distance: German and Swedish are syntactically closer to English than Persian and Basque. Because German, Swedish, and English are also genealogically related, the observed grouping may reflect a combination of syntactic similarity, lexical overlap, and other shared properties rather than the URIEL+ syntactic measure alone. Persian is particularly informative because it has relatively low phoneme-inventory distance but high syntactic distance from English. Its lower grammatical scores indicate that phoneme-inventory similarity alone is not sufficient to predict performance on these grammatical evaluations. However, because each combination of syntactic and phoneme-inventory distance is represented by only one language, the observed ordering cannot be attributed independently to syntactic distance rather than to other language-specific properties.

The BabySLM lexical task exhibits little variation at the final checkpoint: all bilingual phoneme LMs score between 86.8 and 87.3, close to the monolingual score of 88.6. Final accuracy therefore provides no evidence of a persistent L1-related difference on this task. As shown by the checkpoint analysis, however, this convergence conceals differences that arise earlier in training.

Cross-modal comparison Figure 3 compares grammatical performance across L1s in the orthographic and phoneme conditions. The orthographic LMs achieve higher absolute scores overall, but the ordering of L1 conditions is considerably more con-

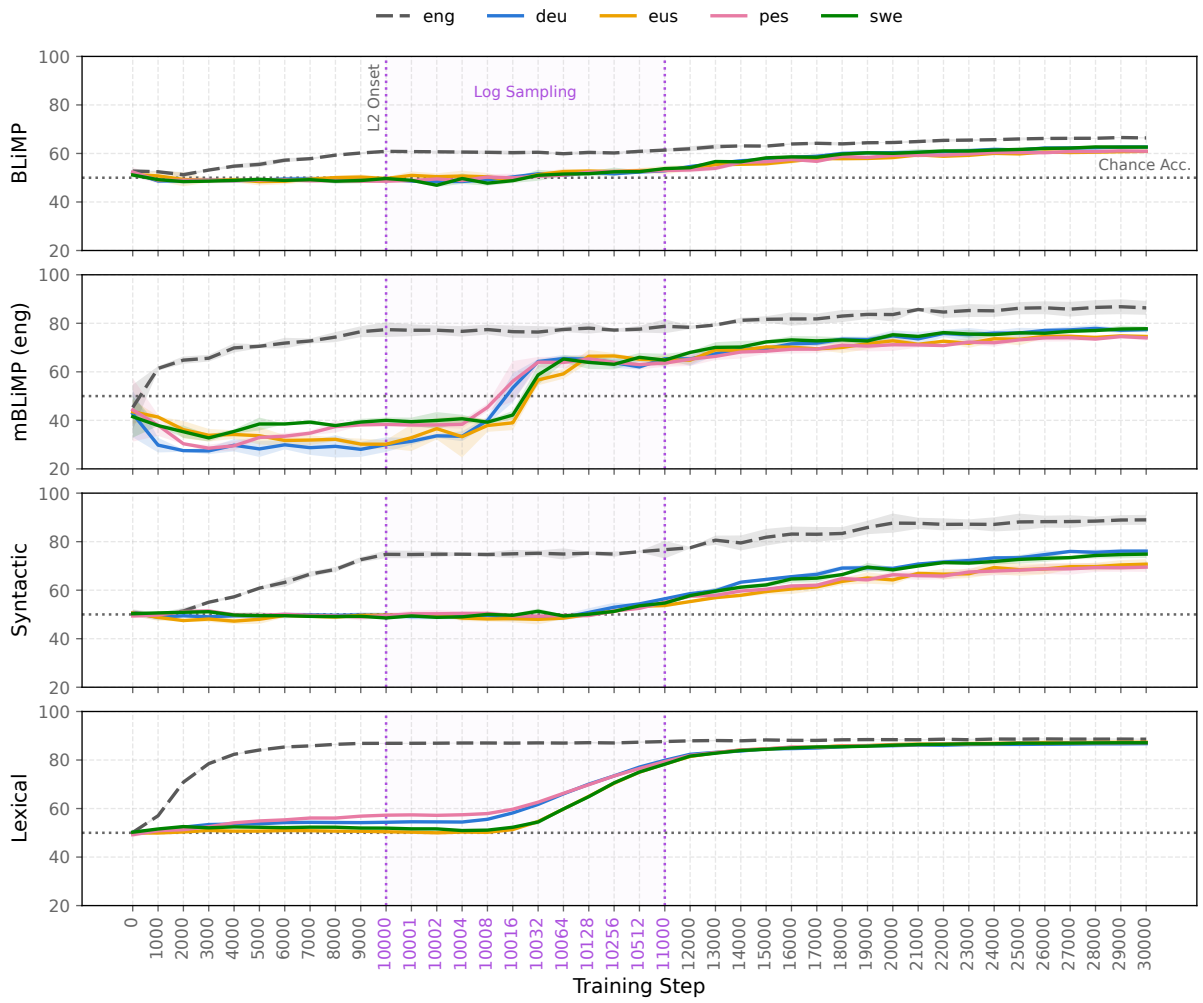


Figure 4: Evaluation accuracy over training for the bilingual **Phoneme LMs**. The first vertical dotted line marks the onset of joint L1–English training; horizontal dotted lines indicate chance performance. Lines correspond to average performance across 3 random seeds with ± 1 SD shading. For the first 1,000 steps after L2 introduction (10k–11k) we save logarithmically spaced checkpoints, highlighted with violet.

sistent among the phoneme LMs, with German and Swedish outperforming Persian and Basque across grammatical evaluations. This is also visible in the correlation coefficients computed over L1 accuracy and syntactic distance. Specifically, phoneme LMs achieve correlations of -0.94, -0.99, and -0.95 for BLiMP, MultiBLiMP, and BabySLM Syntactic, respectively, compared to -0.47, $+0.13$ and -0.46 for orthographic LMs. However, due to the small number of L1s, these correlations are descriptive and should be interpreted cautiously rather than as evidence of a predictive relationship between syntactic distance and performance. Overall, models show stronger alignment between L1 identity and grammatical performance under phonemic than orthographic input.

4.2 Training Dynamics

Final-checkpoint scores can conceal differences in when and how linguistic knowledge emerges. We therefore evaluate intermediate checkpoints throughout training. Checkpoints are saved every 1,000 steps, with additional logarithmically spaced checkpoints immediately following the introduction of English. This allows us to observe the effect of the L1–L2 interaction on the learning trajectory.

Figure 4 presents phoneme LM trajectories for the four benchmarks: BLiMP, MultiBLiMP and BabySLM Syntactic and Lexical metrics. Corresponding trajectories for the orthographic LMs are reported in Figure 5. In the orthographic condition, differences among L1 trajectories are smaller and more benchmark-dependent, with no persistent ordering that consistently follows syntactic distance.

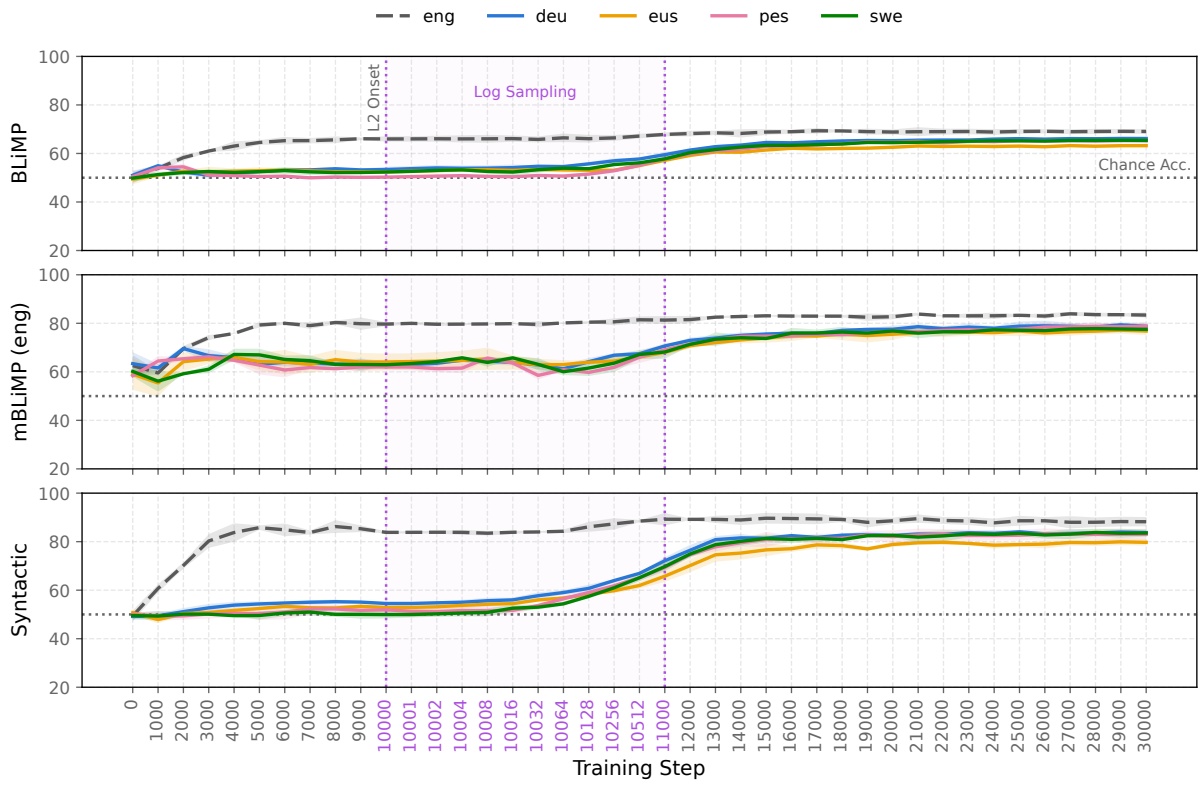


Figure 5: Evaluation accuracy over training for the bilingual **Orthographic LMs**.

Detailed trajectories for each BLiMP sub-category of phenomena are included in Appendix B.

Grammatical learning In phoneme LMs, the BabySLM syntactic trajectories reveal a sustained separation among the L1 conditions after the introduction of English. The German model achieves the highest accuracy throughout much of joint training, while Swedish also generally remains above Persian and Basque. This ordering emerges over the course of English exposure and persists toward the end of training, suggesting that L1 identity is associated not only with final performance but also with the trajectory of English grammatical learning under phonemic input. The separation is also present, but less pronounced, in the phonemic versions of BLiMP and English MultiBLiMP. Benchmark composition may contribute to these differences in sensitivity. BLiMP averages over a large and heterogeneous set of phenomena, potentially obscuring effects confined to particular categories, whereas MultiBLiMP evaluates only two subject–verb agreement phenomena. BabySLM Syntactic covers a smaller set of relatively simple contrasts, some involving word order, and may therefore expose L1-related differences that are diluted in broader aggregate scores. Dynamics for individual BLiMP phenomena are analyzed in Appendix B.

More generally, bilingual-acquisition research suggests that cross-linguistic influence should be property-specific rather than a uniform consequence of global language similarity. In particular, influence has been associated with structural overlap in the relevant construction (Hulk and Müller, 2001). The stronger BabySLM separation may reflect overlap in particular word-order or syntactic patterns, rather than syntactic distance as a whole.

Lexical learning The BabySLM lexical trajectories reveal a transient pattern that is absent from the final scores. During L1-only training, the German and Persian models start with higher English lexical accuracy than the Swedish and Basque models. This similarity vanishes after the first 1,000 steps of L2 exposure, where all models rapidly converge to near 88% accuracy. German and Persian are the two selected languages with lower phoneme-inventory distance from English, making this ordering consistent with an effect of phoneme-inventory similarity in early lexical performance. Although the effect has a small duration, it is consistent with an L1-specific influence in accelerating the acquisition of a linguistic property that has been observed in studies with bilingual children (Serratrice, 2013).

5 Discussion

Our results suggest that cross-linguistic influence varies across linguistic levels and stages of learning. Phoneme-inventory similarity is associated with an early lexical advantage that disappears after sustained English exposure, whereas syntactic similarity aligns with more persistent grammatical differences. This is consistent with accounts in which bilingual systems remain differentiated but interact selectively depending on specific properties (Paradis and Genesee, 1996; Serratrice, 2013). Importantly, the same pattern was substantially weaker and less consistent in orthographic text models.

The lexical trajectories further show that cross-linguistic influence may affect learning rate rather than final performance, highlighting the limits of final-checkpoint evaluation. Across modalities, orthographic input yields higher grammatical accuracy, whereas phonemic input produces clearer L1-related ordering. These interpretations remain exploratory, as each distance configuration is represented by a single language and the L1 corpora differ substantially in composition (Appendix C).

6 Conclusion

In this work, we investigated cross-linguistic effects in bilingual BabyLMs trained on orthographic and phonemic input. Orthographic LMs achieved higher overall grammatical accuracy, whereas phoneme LMs showed clearer L1-related variation: syntactic similarity aligned with more persistent differences, while phoneme-inventory similarity aligned with an early lexical advantage that disappeared after English exposure. These findings suggest that input modality shapes both performance and cross-linguistic influence during learning.

Limitations

Our work is limited in its cross-linguistic analysis by the selection of only four L1 languages (German, Swedish, Persian, Basque) and only one L2 (English). To strengthen evidence for cross-linguistic effects it is important to expand analysis to more typologically diverse pairs of languages. Additionally, we only utilized one model architecture (GPT-2) with hyperparameters from the Goldfish models (Chang et al., 2026). We train bilingual LMs using sequential-interleaved training, but other data mixing methods have been explored in the literature (Constantinescu et al., 2025; Arnett et al., 2025), including pure sequential training (L1 followed by L2) or interleaved training (L1 mixed with L2). Future work should train phoneme LMs to disentangle the effect of these conditions. Lastly, grapheme LMs (Goriely et al., 2024; Bunzeck et al., 2024) provide an interesting alternative to both tested configurations. Exploring whether character-based training aligns more closely with phoneme or orthographic LMs would help better identify the factors of the observed cross-linguistic differences.

Our goal when training phoneme LMs was to provide more ecological training conditions when simulating language acquisition. However, phonemes are only an intermediate step in getting to the real input that children are exposed to: speech. In this case, while it allows us to get closer to child input, phoneme input omits key features that are crucial in acquisition, such as prosody, timing, and stress. Cross-linguistic effects should also be investigated with raw speech input, following the quest to build a more accurate reconstruction of the (bilingual) child learner (Dupoux, 2018).

References

Tatsuya Aoyama and Nathan Schneider. 2024. Modeling Nonnative Sentence Processing with L2 Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4927–4940, Miami, Florida, USA. Association for Computational Linguistics.

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Catherine Arnett, Tyler A. Chang, James A. Michaelov, and Ben Bergen. 2025. On the acquisition of shared grammatical representations in bilingual language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20707–20726, Vienna, Austria. Association for Computational Linguistics.

Laura Barbenel, Lily Goulder, Aoife O'Driscoll, Suchir Salhan, Catherine Arnett, Andrew Caines, and Paula Buttery. 2026. L1 influence in L2 language models: A human-centric approach. In *Proceedings of the 1st Workshop on Computational Developmental Linguistics (CDL)*, pages 92–116, Grand Hyatt Manchester San Diego, 1 Market Pl, San Diego, CA 92101. Association for Computational Linguistics.

Liel Binyamin and Elior Sulem. 2026. Learning from child-directed speech in two-language scenarios: A French-English case-study. In *Findings of the Association for Computational Linguistics: EACL 2026*,

pages 6412–6426, Rabat, Morocco. Association for Computational Linguistics.

Bastian Bunzeck, Daniel Duran, Leonie Schade, and Sina Zarrieß. 2024. Graphemes vs. phonemes: battling it out in character-based language models. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 54–64, Miami, FL, USA. Association for Computational Linguistics.

Bastian Bunzeck, Daniel Duran, Leonie Schade, and Sina Zarrieß. 2025. Small language models also work with small vocabularies: Probing the linguistic abilities of grapheme- and phoneme-based baby llamas. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6039–6048, Abu Dhabi, UAE. Association for Computational Linguistics.

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin Bergen. 2026. Goldfish: Monolingual language models for 350 languages. In *Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026)*, pages 3750–3781, Palma, Mallorca, Spain. European Language Resources Association (ELRA).

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K. Bergen. 2024. When is multilinguality a curse? language modeling for 250 high- and low-resource languages. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4074–4096, Miami, Florida, USA. Association for Computational Linguistics.

Théo Charlot, Tarek Kunze, Maxime Poli, Alejandrina Cristia, Emmanuel Dupoux, and Marvin Lavechin. 2026. Babyhubert: Multilingual self-supervised learning for segmenting speakers in child-centered long-form recordings. *Preprint*, arXiv:2509.15001.

Ionut Constantinescu, Tiago Pimentel, Ryan Cotterell, and Alex Warstadt. 2025. Investigating critical period effects in language acquisition through neural language models. *Transactions of the Association for Computational Linguistics*, 13:96–120.

Emmanuel Dupoux. 2018. Cognitive science in the era of artificial intelligence: A roadmap for reverse-engineering the infant language-learner. *Cognition*, 173:43–59.

Yuan Gao, Suchir Salhan, Andrew Caines, Paula Buttery, and Weiwei Sun. 2025. BLiSS: Evaluating bilingual learner competence in second language small language models. In *Proceedings of the First BabyLM Workshop*, pages 160–174, Suzhou, China. Association for Computational Linguistics.

Zébulon Goriely and Paula Buttery. 2025a. BabyLM's first words: Word segmentation as a phonological probing task. In *Proceedings of the 29th Conference on Computational Natural Language Learning*, pages 522–539, Vienna, Austria. Association for Computational Linguistics.

Zébulon Goriely and Paula Buttery. 2025b. IPA CHILDES & G2P+: Feature-rich resources for cross-lingual phonology and phonemic language modeling. In *Proceedings of the 29th Conference on Computational Natural Language Learning*, pages 502–521, Vienna, Austria. Association for Computational Linguistics.

Zébulon Goriely, Richard Diehl Martinez, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2024. From babble to words: Pre-training language models on continuous streams of phonemes. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 37–53, Miami, FL, USA. Association for Computational Linguistics.

Aafke Hulk and Natascha Müller. 2001. Bilingual first language acquisition at the interface between syntax and pragmatics. *Bilingualism: Language and Cognition*, 3:227–244.

Abderrahmane Issam, Yusuf Can Semerci, Jan Scholtes, and Gerasimos Spanakis. 2026. Language models as artificial learners: Investigating crosslinguistic influence. *Preprint*, arXiv:2601.21587.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Aditya Khan, Mason Shipton, David Anugraha, Kaiyao Duan, Phuong H. Hoang, Eric Khiu, A. Seza Doğruöz, and En-Shiun Annie Lee. 2025. URIEL+: Enhancing linguistic inclusion and usability in a typological and multilingual knowledge base. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6937–6952, Abu Dhabi, UAE. Association for Computational Linguistics.

Marvin Lavechin, Yaya Sy, Hadrien Titeux, María Andrea Cruz Blandón, Okko Räsänen, Hervé Bredin, Emmanuel Dupoux, and Alejandrina Cristia. 2023. Babyslm: language-acquisition-friendly benchmark of self-supervised spoken language models. In *INTERSPEECH 2023*, page 4588–4592. ISCA.

Miyu Oba, Tatsuki Kuribayashi, Hiroki Ouchi, and Taro Watanabe. 2023. Second Language Acquisition of

Neural Language Models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13557–13572, Toronto, Canada. Association for Computational Linguistics.

Johanne Paradis. 2001. Do bilingual two-year-olds have separate phonological systems? *International Journal of Bilingualism*, 5(1):19–38.

Johanne Paradis and Fred Genesee. 1996. Syntactic acquisition in bilingual children: Autonomous or interdependent? *Studies in Second Language Acquisition*, 18:1–25.

Ludovica Serratrice. 2013. Cross-linguistic influence in bilingual development. *Linguist. Approaches Biling.*, 3(1):3–25.

Zhewen Shen, Aditya Joshi, and Ruey-Cheng Chen. 2024. BAMBINO-LM: (Bilingual-)Human-Inspired Continual Pre-training of BabyLM. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 1–7, Bangkok, Thailand. Association for Computational Linguistics.

Chantal Van Dijk, Elise Van Wonderen, Elly Koutamanis, Gerrit Jan Kootstra, Ton Dijkstra, and Sharon Unsworth. 2022. Cross-linguistic influence in simultaneous and early sequential bilingual children: a meta-analysis. *J. Child Lang.*, 49(5):897–929.

Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020. On negative interference in multilingual models: Findings and a meta-learning treatment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4438–4450, Online. Association for Computational Linguistics.

Alex Warstadt and Samuel R Bowman. 2022. What artificial neural networks can tell us about human language acquisition. In *Algebraic structures in natural language*, pages 17–60. CRC Press.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell, editors. 2023. *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Singapore.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Aditya Yadavalli, Alekhya Yadavalli, and Vera Tobin. 2023. SLABERT talk pretty one day: Modeling second language acquisition with BERT. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11763–11777, Toronto, Canada. Association for Computational Linguistics.

Linda Zeng, Steven Y. Feng, and Michael C. Frank. 2026. Bringing up a bilingual babylm: Investigating multilingual language acquisition using small-scale models. *Preprint*, arXiv:2603.29552.

A Training Details

Training Steps	
Phase one (L1)	10k
Phase two (L1+L2)	20k
Monolingual	30k
Embedding params (orthographic / phoneme)	23M / 500k
Non-embedding params	85M
Total	108M / 85M
Layers	12
Embedding size	768
Intermediate hidden	3072
Attention heads	12
Attention head size	64
Learning rate	1e-4
Batch size	32
Epochs per language	10
Activation function	GELU
Max sequence length	512
Position embedding	Absolute
Learning rate decay	Linear
Warmup steps	10% of pretraining
Adam ϵ	1e-6
Adam β_1	0.9
Adam β_2	0.999
Dropout	0.1
Attention dropout	0.1

Table 2: Training hyperparameters.

We use training hyperparameters from the GoldFish (Chang et al., 2026) suite of models with minimal changes, which we include in Table 2. Orthographic LMs are trained with a BPE tokenizer and a vocabulary of 30,000 tokens. For phoneme LMs we construct a phoneme-based tokenizer consisting of phonemes with more than 10 occurrences in the bilingual corpus. A separate tokenizer is trained for each language pair on the bilingual training data. For both training phases (sequential, interleaved) we keep the same optimizer and learning rate scheduler, changing only the data source. Training was done with mixed-precision (bf16) and took approximately 1 hour on `1x RTX 5090 GPU` for each model in both orthographic and phoneme conditions. In total, for 3 different seeds, this amounts to 30 hours of compute across both modalities.

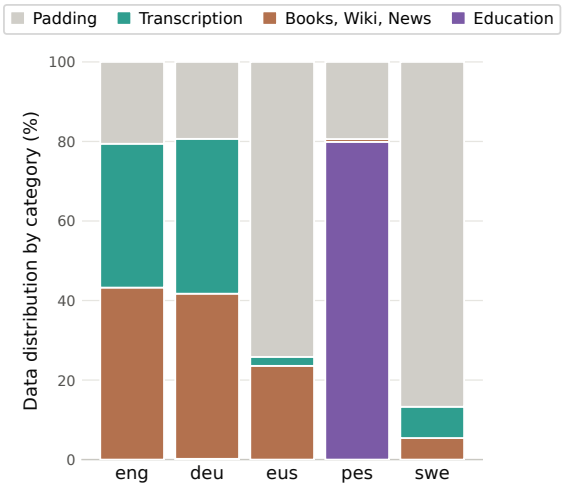


Figure 6: Data category distribution across languages.

Differences between modalities are mainly attributed to the tokenizer, which accounts for diverging embedding sizes (23M vs. 500k), and a reduced effective context in phoneme LMs. Even though all models are trained with a sequence length of 512 tokens, the uncompressed phoneme stream allows less content to fit in one block compared to BPE.

B Detailed BLiMP Performance

	L1	Morphology	Syntax	Syntax/Semantics	Semantics
Orthographic	eng	79.0	64.9	67.9	64.0
	deu	75.6	61.7	65.4	62.0
	eus	71.9	59.8	63.0	57.4
	pes	74.8	62.2	66.2	55.7
	swe	74.2	62.1	65.7	58.3
Phoneme	eng	79.4	60.4	67.8	57.8
	deu	71.6	58.1	64.2	57.0
	eus	68.3	58.1	63.2	52.6
	pes	67.4	58.0	63.9	54.0
	swe	72.1	59.1	64.9	53.0

Table 3: BLiMP accuracy across linguistic fields.

Aggregate benchmark performance may hide L1 or modality effects on the level of specific linguistic phenomena. To investigate this, we plot learning trajectories for every BLiMP category (Figures 7, 8) and average performance for each linguistic field in Table 3. Overall, models followed different learning patterns for each phenomenon. Performance either saturated early (Filler-gap Dependencies, Control/Raising), improved throughout training (Determiner-Noun Agr., Ellipsis), or stayed near chance (Island Effects, NPI Licensing). As expected, English monolingual LMs generally performed better than bilingual for both modalities, particularly in morphology. Overall, models score highest in morphology and lowest in semantics.

Orthographic LMs showed no strong cross-linguistic effects (Fig. 7), except in the case of `eus` were we occasionally see reduced performance (S-V Agr., Quantifiers). In contrast, for phoneme LMs, we notice that the languages most syntactically similar to English (`deu`, `swe`) scored higher for specific phenomena, two in morphology (Det-Noun Agr., S-V Agr.) and one in syntax (Ellipsis), indicating an influence of syntactic or genealogical similarity on bilingual model performance that was also observed for the aggregate BLiMP accuracy.

When comparing across modalities, phoneme LMs achieve similar scores to the orthographic ones, except for semantics. This might stem from the reduced context length of phoneme LMs, which makes it harder to learn long-range semantic dependencies. Training stability also depended on modality, with phoneme LMs taking longer to converge for some phenomena (Irregular Forms, Anaphor Agr., Quantifiers) compared to orthographic ones.

C Data Distribution

Data distribution for all languages is illustrated in Figure 6. Languages in the 100M word tier (German, Persian, English) were downsampled to 10M equivalent English words, adjusted using byte-premiums (Arnett et al., 2024), and approximating as much as possible the data distribution of English. Data source — transcribed spontaneous speech or written language — can influence model performance, particularly for phoneme LMs where training data must undergo phonemization. While an important factor, we opted for developmentally plausible corpora rather than parallel or category-matched training datasets. The influence of data-source on cross-linguistic effects should be investigated in future work.

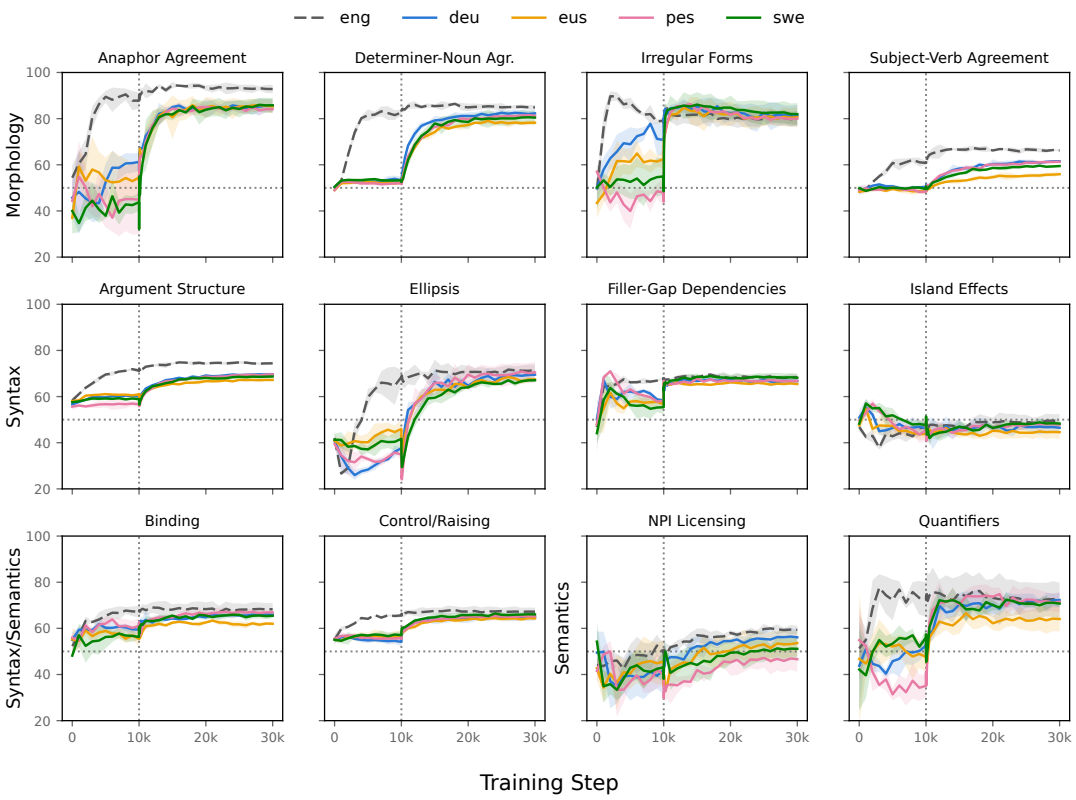


Figure 7: BLiMP accuracy per category for **Orthographic LMs** throughout training. Dashed lines indicate L2 onset and random change accuracy (50%). Lines correspond to average performance across 3 random seeds with ± 1 SD shading. Plots in each row are grouped according to the broader linguistic field: Morphology (first row), Syntax (second row), Syntax/Semantics (third row left), Semantics (third row right).

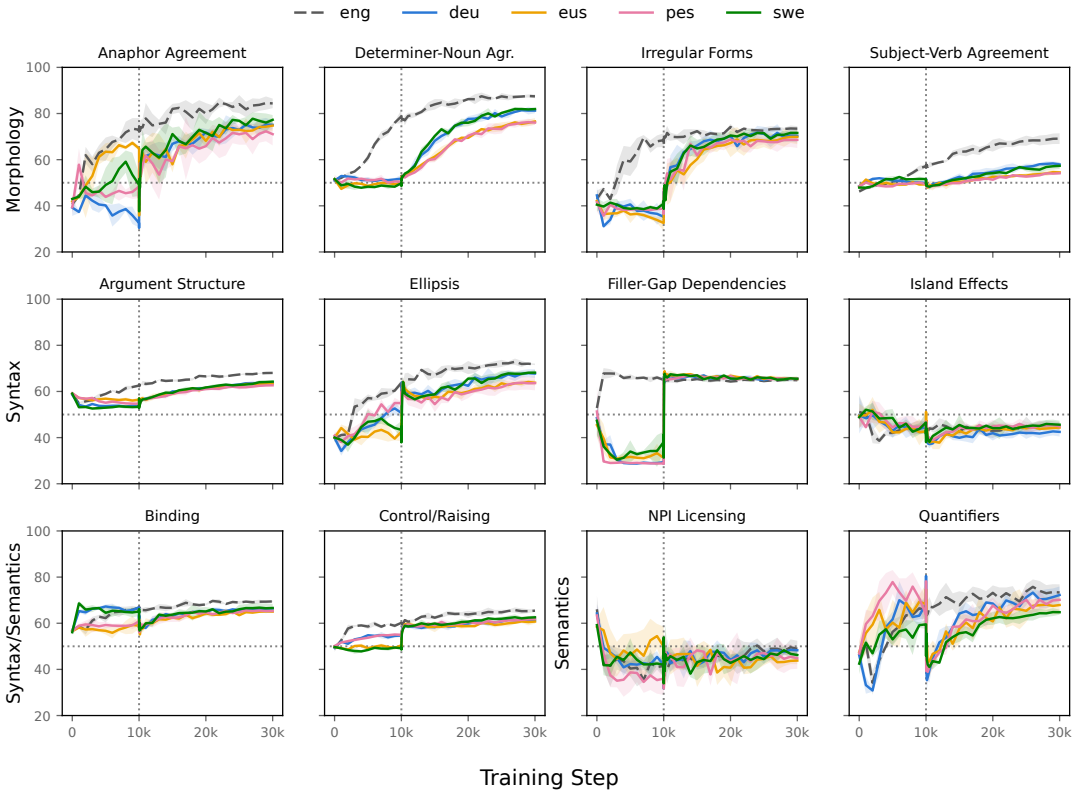


Figure 8: BLiMP accuracy per category for **Phoneme LMs** throughout training.

TinyLibrary: Do Age-Graded Curricula Help Small Vision-Language Models?

Dheeraj Varghese
University of Amsterdam
d.varghese@uva.nl

Abstract

Developmentally plausible curricula are a recurring hypothesis in research on cognitively plausible language-model training, but it remains unclear which notions of difficulty produce useful training curricula. We ask whether annotations derived from children's literature yield a usable ordering signal. We introduce TinyLibrary, a corpus of 46K illustrated children's-book pages drawn from a collection for readers aged 3–13. We infer each book's age band from its OCR text and use the resulting label to condition the generation of multimodal training data. This design lets us test whether the developmental structure of children's literature can be used to generate synthetic multimodal training data as an effective curriculum signal. We train small vision–language models for up to 10 epochs, using a pretrained vision encoder and a language model trained from scratch, and compare five curriculum orderings at matched token budgets. We evaluate the resulting models on the zero-shot BabyLM 2026 suite. Our results show that the age progression only partly aligns with teacher-model predictability: readability increases monotonically as intended, whereas teacher perplexity is U-shaped, with text for the youngest readers among the least predictable per token. Consistent with this mismatch, age-based ordering shows no reliable advantage over random ordering. Teacher-perplexity ordering achieves the highest mean BLiMP score and exceeds its seed-matched random counterpart in 10 of 12 comparisons, but this advantage transfers unevenly across teacher families and is entangled with annotation format. No ordering yields a reliable grounding advantage: Winoground remains at chance, while VQA is above chance but variable across conditions and seeds. We release the annotations and code.

1 Introduction

Curriculum learning asks whether models learn more effectively when training examples are presented in a structured progression rather than in random order (Elman, 1993; Bengio et al., 2009). This question is especially relevant to the BabyLM challenge (Warstadt et al., 2023; Hu et al., 2024; BabyLM Organizing Committee, 2026), where models must learn from developmentally plausible data budgets. A useful curriculum could improve learning without requiring more data or a larger model. Its success, however, depends on what counts as "easy": input that is appropriate for a young reader is not necessarily predictable or useful to a language model. The central question is therefore not only whether curriculum learning helps, but whether a human developmental ordering provides the right difficulty signal for the learner.

Previous BabyLM studies have tested curricula based on properties such as sentence length, readability, or teacher-model perplexity, where a separate pretrained language model scores how predictable each sample is (Martinez et al., 2023; Saha et al., 2024). Reported gains have been weak or setting-dependent. These measures are typically applied *post hoc* to corpora that were not created around an intended progression. Children's literature provides a complementary test case: books are written and illustrated for readers at different developmental levels, offering a source-derived, human-motivated ordering signal. We ask whether this age-targeted structure can be transferred into synthetic multimodal training data and used as an effective curriculum for small vision–language models. We address this question through three contributions:

1. **We introduce TinyLibrary, a multimodal corpus constructed to study age-based curriculum ordering.** Constructed from the English-language portion of the International Children's Digital Library (ICDL; Druin, 2005), TinyLibrary contains 46,721 illustrated pages and 128,823 synthetic captioning, visual question-answering, and reasoning sam-

Proceedings of the Second BabyLM Workshop, pages 333–347
October 29, 2026 ©2026 Association for Computational Linguistics

ples, totaling 21.7M words. Because reliable per-book age metadata was unavailable, we infer one of four coarse age bands from each book's original OCR text. To create image-grounded VLM examples while preserving this age signal, we use the inferred band to condition the generation of captioning, visual question-answering, and reasoning samples.

2. **We find that inferred age order does not provide a stable curriculum advantage.** Using nanoVLM (Wiedmann et al., 2025), we compare developmental age order and its reverse with readability, teacher-perplexity, and random ordering under matched token budgets, data splits, and seeds. As intended, age conditioning produces progressively higher Flesch–Kincaid grades. However, mean per-token negative log-likelihood under a teacher language model is U-shaped across age bands: it is highest for the youngest and oldest bands and lowest for the middle bands. Neither developmental nor reverse-age ordering yields a stable BLiMP advantage across schedules and training durations; at 10 epochs, random ordering is higher in all 12 seed-matched comparisons.

3. **Teacher-perplexity ordering achieves the highest mean BLiMP, but does not establish a portable curriculum advantage.** Its advantage transfers unevenly across teacher-model families. Moreover, the resulting difficulty quartiles largely separate annotation types: the easiest samples are predominantly question answering and reasoning, while the hardest are almost entirely captions. The observed advantage therefore cannot be attributed confidently to a general notion of model-relative difficulty.

Together, these results distinguish human age appropriateness from model predictability: the former does not yield stable curriculum gains in our setting, while the apparent benefit of the latter remains entangled with teacher choice and task format.

2 Related Work

Learning under constrained data. The BabyLM challenge provides a standardized setting for studying how effectively language models learn from developmentally motivated data budgets (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025a; BabyLM Organizing Committee, 2026). Prior work shows that small models can acquire substantial grammatical knowledge from child-directed corpora (Huebner et al., 2021), while simplified synthetic data can also support strong learning under limited exposure (Eldan and Li, 2023). Since the introduction of the multimodal track, visual input has not consistently translated into stronger text-only grammatical performance (Hu et al., 2024). Findings from the third BabyLM challenge likewise emphasize architectural and objective choices among successful systems (Charpentier et al., 2025b). Our work instead holds the model and training budget fixed to study whether the order of multimodal training examples improves learning.

Curricula for language models. The "starting small" hypothesis proposes that learning improves when models encounter simpler or more restricted input before more complex data (Elman, 1993). Rohde and Plaut (1999) questioned the generality of this effect, while later work formalized curriculum learning as ordering examples by difficulty (Bengio et al., 2009; Platanios et al., 2019). In BabyLM, CLIMB (Martinez et al., 2023) found no reliable gains from vocabulary or data curricula, and Saha et al. (2024) reported only setting-dependent gains for heuristic multimodal curricula. Most closely related to our question, Feng et al. (2024) find that the local properties of child-directed speech affect learning, whereas its global developmental ordering does not. We extend this line of investigation to multimodal data derived from age-targeted children's books. We compare inferred age order with readability and teacher perplexity, and by treating the ordering signal and its training schedule as separate design choices.

Developmentally inspired multimodal training. Developmental research links word learning to the structure of children's visual experience: Yu and Smith (2012) find that toddlers' learning of object names is associated with the named object dominating their view during naming. In computational work, Sheybani et al. (2023) find that presenting infant egocentric videos in developmental order improves self-supervised visual learning, with benefits linked to the slower-changing and spatially simpler views of younger infants. For multimodal learning, Vong et al. (2024) demonstrate word-referent learning from paired visual and linguistic input recorded from one child's experience,

while BabyVLM (Wang et al., 2025) constructs synthetic child-directed training data for vision-language models. Our work examines a complementary source of developmental structure: illustrated books written for readers of different ages. We test whether inferred reader age provides an effective ordering signal for synthetic multimodal annotations derived from these books.

3 The TinyLibrary Corpus

Source and page filtering. TinyLibrary[1] is derived from the International Children's Digital Library (ICDL; Druin, 2005), a curated collection of children's books for readers aged 3–13. Of 2,801 OCR-processed books, 2,795 receive an age-band label using the procedure described below. We retain the 2,416 English-language books, comprising 251,410 scanned pages.

Because our goal is to construct image-grounded training examples, we select pages with substantial non-text visual content. We remove pages dominated by OCR-detected text (PaddleOCR; Du et al., 2020), near-blank pages, and pages whose visual complexity largely disappears after detected text is removed. This process yields 46,721 illustrated pages. Appendix A provides the full filtering criteria, and Appendix B shows example pages.

Inferring age bands. We use Mistral-7B-Instruct (Jiang et al., 2023) to clean each book's original OCR text and infer one of four target-age bands: 3–5, 6–8, 9–12, or 12+. Long books are divided into text chunks, classified separately, and assigned a band by majority vote.

Generating multimodal annotations. We generate annotations in two stages. First, CogVLM2 (Hong et al., 2024) produces a detailed description of each selected page image. Llama-3-8B-Instruct (Grattafiori et al., 2024) then converts the description into LLaVA-style (Liu et al., 2023) captioning, visual question-answering, and reasoning samples. The generation prompt is conditioned on the book's inferred age band and specifies corresponding vocabulary, sentence structure, and reasoning complexity. The resulting corpus contains 128,823 samples and 21.7M words.

Corpus composition and ordering signals. Table 1 summarizes the corpus by inferred age band. The distribution is imbalanced: the 12+ band contains 60% of all words. We characterize the generated target text using Flesch–Kincaid grade (Kincaid et al., 1975) and mean per-token negative log-likelihood (NLL) under a SmolLM2-135M teacher language model (Allal et al., 2025), conditioned on the sample prompt.

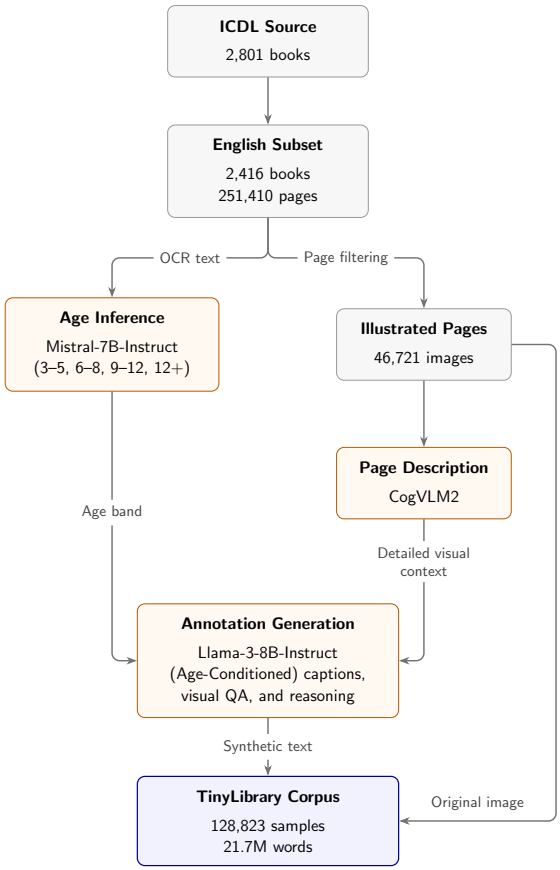


Figure 1: TinyLibrary construction. OCR text is used to infer a book-level age band, while illustrated pages are selected for annotation. Page descriptions are generated from images and converted into age-conditioned captioning, visual QA, and reasoning samples.

As intended, the age conditioning produces progressively higher Flesch–Kincaid grades, from 3.51 for the 3–5 band to 10.76 for the 12+ band, indicating that the generated annotations follow the requested reading levels.

Teacher-model NLL follows a different pattern. It is highest for the 3–5 band, lowest for the two middle bands, and rises again for the 12+ band. The same band-level pattern is reproduced by Qwen3-0.6B and Gemma-3-270M, whose sample-level rankings correlate strongly with the SmolLM2 ranking (Spearman $\rho = 0.91$ and 0.96; Appendix F). Thus, inferred age order and teacher-NLL order encode different sample

[1]Dataset on Hugging Face; code on GitHub.

Band	Samples	Words	Tok.	FK	NLL
3–5	34,623	3.36M	128	3.51	2.856
6–8	18,389	2.51M	170	5.34	2.542
9–12	15,488	2.70M	212	7.76	2.523
12+	60,323	13.1M	264	10.76	2.748
Total	128,823	21.7M	—	—	—

Table 1: TinyLibrary by inferred age band: sample and word counts, mean tokens per sample, mean Flesch–Kincaid grade, and mean per-token NLL under a SmolLM2-135M teacher. The 12+ band contains 60% of the corpus words.

progressions. This divergence motivates treating them as separate curriculum orderings.

4 Experimental Setup

Model and training. We use nanoVLM (Wiedmann et al., 2025), comprising a SigLIP2-base-512 vision encoder (Tschannen et al., 2025), a two-layer MLP projector, and a language model with the SmolLM2-360M architecture (Allal et al., 2025), for 460M parameters in total. The vision encoder is initialized from pretrained weights, while the projector and language model are initialized from scratch; all components are then trained jointly. Images are processed at 512×512 resolution, producing 64 visual tokens, and samples are packed into 4,096-token sequences. Appendix C provides the remaining training details.

Curriculum orderings. We distinguish the sample *ordering* from the schedule used to present that ordering during training. We compare five orderings:

1. **Developmental:** inferred age bands in ascending order, beginning with 3–5;
2. **Anti-developmental:** the reverse age-band order, beginning with 12+;
3. **Readability:** ascending Flesch–Kincaid grade;
4. **Teacher perplexity:** ascending per-token NLL under SmolLM2-135M; and
5. **Random:** a fresh random permutation of the corpus in each epoch.

The anti-developmental condition tests whether the direction of the inferred age progression matters. Readability and teacher perplexity provide continuous alternatives based on surface complexity and teacher-model predictability, respectively.

Schedule implementations. We implement each of the four non-random orderings using two schedules. The staged schedule partitions the corpus into four consecutive groups: the four age bands for the developmental conditions and score quartiles for readability and teacher perplexity. Samples are shuffled within each group, but group membership and group order remain fixed across epochs. The age-band groups retain their naturally unequal sizes, whereas the continuous-score quartiles contain equal numbers of samples. Each staged epoch presents every training sample once.

The competence-based schedule (Platanios et al., 2019) samples with replacement from an expanding prefix of the ranked corpus. At training step t out of T, the available fraction is

$$c(t) = \min\left(1, \sqrt{c_0^2 + \frac{t}{T}(1 - c_0^2)}\right), c_0 \quad = 0.1.$$

Training therefore begins by sampling from the first 10% of the selected ordering, and the available pool expands gradually until the full corpus becomes available at the end. Because sampling is performed with replacement, early-ranked examples can be repeated and some later-ranked examples may never be observed.

The random condition does not use either schedule: it presents every sample once per epoch and reshuffles the full corpus between epochs. Crossing the four non-random orderings with the two schedules, together with the random baseline, produces nine core conditions.

Matched comparisons. All runs use the same model architecture, the same static 2% sample-level validation split, training hyperparameters, and consume the same total number of training tokens. We reuse the same three seed values across orderings, allowing each curriculum run to be paired with the corresponding random-ordering run.

In staged and random runs, each epoch is one complete corpus pass, so their comparison isolates presentation order. For competence-based runs, the 5- and 10-epoch labels denote equivalent token budgets rather than complete corpus passes. Because competence-based sampling changes both presentation order and unique-sample coverage, we base our primary ordering comparisons on the staged conditions and the random baseline and treat the competence-based results as secondary.

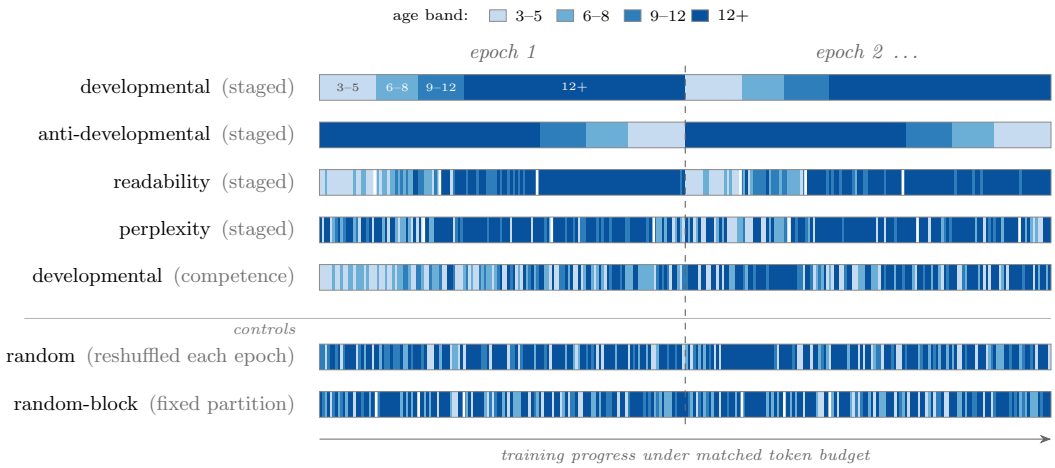


Figure 2: Age-band composition induced by the curriculum orderings and schedules. Color denotes inferred age band, and every row consumes the same training-token budget. Staged conditions repeat fixed groups across epochs, competence-based conditions sample from an expanding eligible pool, and random ordering reshuffles the complete corpus each epoch. The random-block control repeats fixed groups containing randomly assigned samples.

Runs and controls. We train the nine core conditions for 5 and 10 equivalent epochs using three seeds, yielding 54 runs. We treat the 10-epoch budget as primary and the 5-epoch budget as a second training duration at which to assess whether the direction of an effect persists.

We run three additional controls at the 10-epoch budget. First, the random-block control randomly assigns samples to four fixed groups matching the token masses of the age-band stages. Samples are shuffled within each group, but group membership and order are reused across epochs. This preserves the repeated staged structure without grouping samples by age. Second, we construct teacher-perplexity orderings using Qwen3-0.6B and Gemma-3-270M to test whether the SmolLM2-derived ordering transfers across teacher families. Third, we train text-only models under developmental, teacher-perplexity, and random ordering, removing image tokens while retaining the corresponding text. These controls bring the total to 72 trained runs, in addition to three untrained model initializations.

Evaluation. For linguistic evaluation, we evaluate the trained language-model component without image inputs using the zero-shot subset of the official BabyLM 2026 pipeline (BabyLM Organizing Committee, 2026): BLiMP and BLiMP Supplement (Warstadt et al., 2020), EWoK (Ivanova et al., 2024), COMPS (Misra et al., 2023), and entity tracking (Kim and Schuster, 2023). For multimodal evaluation, we use the complete vision–language model on the BabyLM VQA and Winoground tasks.

Final checkpoints are evaluated on the full task sets. To examine linguistic performance over training, we also evaluate checkpoints saved every 250 steps using the pipeline's fast subsets.

5 Results

Table 2 reports final zero-shot performance at the primary 10-epoch budget. The first nine rows form the core experimental grid; the remaining rows report the random-block and alternative-teacher controls together with the untrained reference. Results for the 5-epoch budget and individual seeds are provided in Appendix D and Appendix E, respectively.

Metric sensitivity. With this 21.7M-word corpus, EWoK and COMPS remain near their 50% chance levels in every condition. BLiMP Supplement varies across conditions but does not produce a consistent curriculum ranking. Entity tracking is also difficult to interpret: the untrained model scores 41.8 $\pm$0.1, while training generally lowers performance and produces substantial seed variance. We therefore use BLiMP as the primary outcome for comparing the curriculum orderings; the other linguistic metrics do not yield a consistent or interpretable ranking of those orderings.

Order	Schedule	BLiMP	Supplement	EWoK	COMPS	Entity tracking
Random	—	53.4 ±0.4	44.5 ±1.7	49.1 ±0.6	49.8 ±0.2	30.8 ±5.5
Developmental	Staged	51.8 ±0.6	46.5 ±2.6	49.4 ±0.4	50.2 ±0.4	24.5 ±15.2
Developmental	Competence	52.2 ±1.0	46.0 ±1.0	50.3 ±0.5	50.0 ±0.1	34.8 ±5.7
Anti-developmental	Staged	51.5 ±1.2	48.1 ±0.7	50.5 ±0.6	49.8 ±0.1	24.7 ±4.0
Anti-developmental	Competence	51.7 ±0.6	47.1 ±3.2	50.1 ±0.3	49.9 ±0.2	27.1 ±11.1
Readability	Staged	52.8 ±1.7	46.2 ±1.3	50.1 ±0.4	50.0 ±0.4	27.2 ±8.5
Readability	Competence	52.5 ±0.6	44.6 ±0.9	50.2 ±0.3	50.3 ±0.5	35.1 ±3.4
Teacher perplexity	Staged	**55.2 ±2.1**	48.0 ±1.0	49.6 ±0.3	50.1 ±0.1	17.1 ±2.9
Teacher perplexity	Competence	53.5 ±1.0	46.1 ±1.4	50.5 ±0.4	50.1 ±0.6	30.1 ±2.8
Random-block (control)	Staged	52.8 ±1.1	45.8 ±1.6	50.1 ±0.3	49.8 ±0.0	36.7 ±7.7
Teacher perplexity, Gemma (control)	Staged	54.0 ±0.2	46.0 ±3.0	50.4 ±0.5	50.1 ±0.5	17.3 ±4.4
Teacher perplexity, Qwen3 (control)	Staged	52.6 ±1.1	45.8 ±2.5	50.1 ±0.6	50.0 ±0.1	25.8 ±12.7
Untrained reference		50.0 ±0.4	47.6 ±1.0	50.0 ±0.5	50.4 ±0.3	41.8 ±0.1

Table 2: **Teacher-perplexity/staged has the highest mean BLiMP score, while EWoK and COMPS remain near chance.** Zero-shot accuracy (%) after 10 epochs, reported as mean ± standard deviation over three seeds. The first nine rows form the core grid; random-block and the alternative-teacher orderings are 10-epoch controls. Bold indicates the highest mean BLiMP score.

5.1 Age-based ordering does not yield a stable advantage

In the primary staged comparison at 10 epochs, both age-based orderings finish below random ordering. Developmental/staged scores 51.8 ±0.6 BLiMP, a mean seed-paired difference of −1.56 points from random, while anti-developmental/staged scores 51.5 ±1.2, a paired difference of −1.84. Every seed-level difference for both conditions is negative.

The competence-based age conditions also fall below random at 10 epochs: developmental/competence by −1.21 points and anti-developmental/competence by −1.69 points on average. Because these comparisons reflect differences in sample exposure as well as presentation order, we treat them as secondary. Across the two age directions, two schedules, and three seeds, all 12 paired differences favor random ordering at this budget.

At 5 epochs, the age-based results are mixed: anti-developmental/staged is above random, anti-developmental/competence is below it, and the developmental conditions remain close to random (Appendix D). None of the positive five-epoch differences persists at 10 epochs. Thus, neither age direction provides an advantage that is consistent across schedules and training budgets.

Readability ordering likewise fails to produce a stable improvement. Although Flesch–Kincaid grade increases monotonically with inferred age band, neither readability condition exceeds the random mean at either training budget. Together, these results provide no reliable evidence that either inferred reader age or its induced readability progression supplies a beneficial curriculum ordering in this setting.

5.2 Teacher-perplexity ordering has the highest mean but does not transfer reliably

Teacher-perplexity ordering produces the highest mean BLiMP score at both training budgets, although the strongest schedule differs. At 10 epochs, teacher-perplexity/staged reaches 55.2 ±2.1, a mean seed-paired advantage of +1.87 points over random; the three paired differences range from −0.50 to +4.15. Teacher-perplexity/competence is effectively tied with random at this budget, with a mean paired difference of +0.12.

At 5 epochs, both teacher-perplexity conditions are above random: staged by +1.62 points and competence by +2.42. Across the two schedules, two budgets, and three seeds, teacher-perplexity ordering exceeds its seed-matched random counterpart in 10 of 12 comparisons. This repeated direction makes teacher perplexity the strongest ordering signal in the core grid, but the seed variation and schedule dependence prevent a stronger claim.

The alternative-teacher controls further qualify this result. A staged ordering derived from Gemma-3-270M reaches 54.0 ±0.2, retaining a smaller mean paired advantage of +0.66 over random. The Qwen3-0.6B ordering reaches 52.6 ±1.1 and retains no advantage, with a mean paired difference of −0.74. The gain observed with the SmolLM2

teacher therefore does not transfer consistently across teacher-model families.

Teacher NLL is also strongly entangled with annotation format. The two easiest NLL quartiles contain almost no captions and consist almost entirely of visual question-answering and reasoning samples, whereas the hardest quartile is 99.6% captions (Appendix H). Consequently, under teacher-perplexity ordering, question-answering and reasoning samples are presented predominantly earlier in training, whereas captions are presented predominantly later. The observed BLiMP advantage could therefore reflect teacher–learner alignment, task-format order, or an interaction between them. These experiments do not establish teacher perplexity as a general, portable measure of curriculum difficulty.

5.3 Repeated fixed partitions are associated with late degradation

Figure 3 reports checkpoint performance on the fast BLiMP subset. These trajectories characterize changes during training; final performance in Table 2 is measured on the full task.

All conditions improve rapidly at the beginning of training. Mean Fast-BLiMP rises from 50.2 at initialization to approximately 53.6 within the first 250 steps, around half an epoch, and improves little thereafter. This early saturation leaves limited room for the curriculum orderings to separate.

Several staged conditions subsequently decline. Developmental and anti-developmental staging peak between steps 1,250 and 1,500 and lose 2.5–3.7 points by the final checkpoint. Readability/staged loses 2.2 points, while the fully reshuffled random baseline loses 1.3. In contrast, the SmolLM2 teacher-perplexity/staged condition loses only 0.4 points and retains the highest final mean.

The random-block control tests whether this decline requires age-homogeneous content. It reaches a mean Fast-BLiMP peak of 56.2 at step 500 and then declines by 3.4 points, despite its blocks containing randomly assigned samples. Its mean late-training slope is also more negative than random's (-0.50 versus $+0.03$ points per 1,000 steps; Appendix G). The comparable decline shows that age-based grouping is not necessary for the pattern and is consistent with repeated partition membership contributing to it.

This control does not identify a mechanism or show that every fixed partition is harmful. The SmolLM2 teacher-perplexity partition remains comparatively stable, whereas the Qwen3-teacher partition declines by 2.9 points after an early peak. Fixed partition reuse is therefore associated with the late degradation but is not sufficient to explain when it occurs.

5.4 Visual inputs do not consistently explain the BLiMP ranking, and grounding remains weak

Text-only control. We compare the full VLMs with text-only models trained on the same annotation targets under developmental, teacher-perplexity, and random ordering. The seed-paired VLM-minus-text-only BLiMP differences are $+0.5 \pm 1.1$ for random, $+2.1 \pm 2.7$ for teacher perplexity, and -1.9 ± 0.6 for developmental ordering. The direction of the difference therefore depends on the ordering, and the seed variation is substantial for the positive comparisons. Removing image tokens does not produce a consistent shift in BLiMP or explain the curriculum ranking. This control is not an equivalence test and does not rule out smaller modality effects.

Multimodal grounding. Across the 12 evaluated configurations, Winoground remains at its 50% chance level, with condition means between 49% and 53%. VQA requires selecting among eight candidate answers, giving a chance accuracy of 12.5%. Condition means range from 20% to 31%, but performance varies substantially across seeds, with standard deviations as large as 8 percentage points.

For the staged core orderings and the random-block control, the mean seed-paired VQA differences relative to random are positive. However, the pattern does not support a coherent curriculum effect: anti-developmental and teacher-perplexity ordering produce similarly high means, and the VQA ranking aligns with neither developmental order nor the BLiMP results.

We therefore find no reliable or interpretable evidence that a particular curriculum improves multimodal grounding. Winoground provides no evidence of compositional grounding above chance, while VQA performance is above chance but variable and does not support a coherent ordering effect. The linguistic effects, including the teacher-perplexity BLiMP advantage, do not translate into a consistent multimodal advantage. Appendix J reports the complete results.

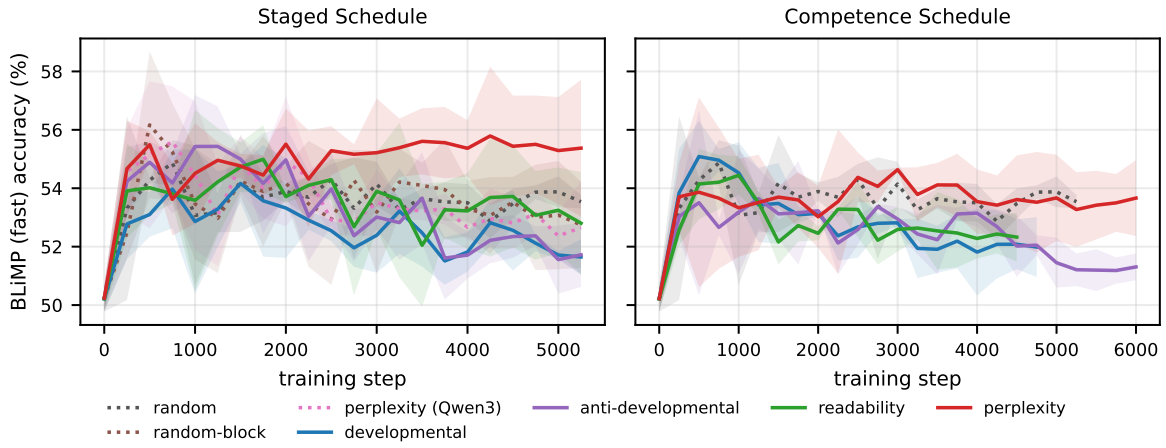


Figure 3: Fast-BLiMP accuracy over 10-epoch training, reported as the mean over three seeds with shading for ± 1 standard deviation. The random curve is repeated in both panels for comparison. The staged panel also includes the random-block and Qwen3-teacher controls. Several conditions with fixed partitions decline late in training, while the SmolLM2 teacher-perplexity/staged condition remains comparatively stable.

6 Discussion

Developmental order is not necessarily learner-relative difficulty. Our results do not imply that children's literature lacks meaningful developmental structure. They show that inferred reader age and the resulting readability progression do not necessarily correspond to what a small language model finds predictable or useful during training. Age grading reflects communicative, lexical, and conceptual demands on human readers, whereas teacher NLL measures conditional predictability. This mismatch offers one plausible interpretation of the age-ordering result, although our experiments do not establish that it causes the absence of an age-ordering advantage. More broadly, this finding complements prior work in BabyLM and related developmentally inspired settings showing weak or setting-dependent curriculum effects (Martinez et al., 2023; Saha et al., 2024; Feng et al., 2024): a human-motivated progression is not automatically an effective model-training curriculum.

Curriculum effects combine ranking, task composition, and schedule. Teacher-perplexity ordering produces the highest mean BLiMP scores, but its uneven transfer across teacher families does not support treating it as a portable difficulty measure. It also changes the order of annotation formats, placing question-answering and reasoning samples predominantly before captions. The random-block results introduce a second complication: they suggest that outcomes may also depend on how fixed sample groups are reused across epochs, not only on the signal used to rank them. At the same time, the comparatively stable SmolLM2 teacher-perplexity staged schedule shows that fixed partitions alone do not explain the observed dynamics. Taken together, the tested curricula alter sample ranking, task composition, and repeated grouping across epochs. The present experiments cannot isolate which of these components accounts for the teacher-perplexity advantage or the late-training decline.

Scope and implications. The text-only controls provide no consistent evidence that visual inputs explain the BLiMP ranking, while the grounding evaluations remain weak and do not reveal a coherent curriculum effect. Our conclusions therefore concern linguistic curriculum behavior within a multimodal training setup, not successful grounded language acquisition from vision. Our results suggest that curriculum difficulty must be defined relative to the learner and objective while remaining separable from task format and schedule construction. In this setting, inferred age organizes the generated data along the intended readability progression but does not provide a reliable curriculum rule; teacher perplexity provides a stronger empirical signal, but the source of that advantage remains unresolved.

7 Conclusion

We introduced TinyLibrary and used it to test whether inferred reader age provides an effective curriculum signal for small vision–language mod-

els under matched token budgets. Across two training budgets, neither developmental nor reverse-age ordering yields a stable BLiMP advantage over random ordering, and no ordering produces a reliable grounding benefit. Teacher-perplexity ordering achieves the highest mean BLiMP scores, but its uneven transfer across teacher families and entanglement with annotation format do not support treating it as a portable measure of model-relative difficulty. Overall, developmentally meaningful structure for human readers does not automatically translate into an effective model-training curriculum. Curriculum effects can depend on the signal used to rank samples, the task formats that the ranking places earlier or later, and how the ranked samples are presented across epochs.

Limitations

Our conclusions concern synthetic age-conditioned annotations derived from model-inferred age bands, not the books' original prose or publisher metadata. We test a single vision-language model architecture and scale, so the findings may not generalize to other model families or sizes.

Ethics Statement

The source books are works from the International Children's Digital Library. We do not distribute the books or the page-image set; the three pages in Figure 4 are shown for illustration of what the corpus looks like. We release the processing pipeline, the synthetic annotations, difficulty scores, curriculum order files, and training code, which suffice to reproduce our comparisons on any similarly structured corpus. The synthetic annotations were generated by Llama-3 and may contain model biases; they are intended for research on training dynamics, not as children's educational material. No personal or user data is involved.

Acknowledgments

The author thanks Yuki M. Asano for supervision and guidance. The author also thanks SURF (https://www.surf.nl) for the support in using the National Supercomputer Snellius.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. 2025. SmolLM2: When smol goes big – data-centric training of a small language model. *ArXiv preprint*, abs/2502.02737.

BabyLM Organizing Committee. 2026. BabyLM turns 4: Call for papers for the 2026 BabyLM workshop. *ArXiv preprint*, abs/2602.20092.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025a. BabyLM turns 3: Call for papers for the 2025 BabyLM workshop. *ArXiv preprint*, abs/2502.10645.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025b. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420. Association for Computational Linguistics.

Allison Druin. 2005. What children can teach us: Developing digital libraries for children with children. *The Library Quarterly*, 75(1):20–41.

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. 2020. PP-OCR: A practical ultra lightweight OCR system. *ArXiv preprint*, abs/2009.09941.

Ronen Eldan and Yuanzhi Li. 2023. TinyStories: How small can language models be and still speak coherent English? *ArXiv preprint*, abs/2305.07759.

Jeffrey L. Elman. 1993. Learning and development in neural networks: the importance of starting small. *Cognition*, 48(1):71–99.

Steven Y. Feng, Noah D. Goodman, and Michael C. Frank. 2024. Is child-directed speech effective training data for language models? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. 2024. The Llama 3 herd of models. *ArXiv preprint*, abs/2407.21783.

Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. 2024. CogVLM2: Visual language models for image and video understanding. *ArXiv preprint*, abs/2408.16500.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. *ArXiv preprint*, abs/2412.05149.

Philip A. Huebner, Elior Sulem, Fisher Cynthia, and Dan Roth. 2021. BabyBERTa: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 624–646, Online. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R.T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyurek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *ArXiv preprint*, abs/2405.09605.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *ArXiv preprint*, abs/2310.06825.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

J. Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command, Millington TN Research Branch.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Richard Diehl Martinez, Hope McGovern, Zebulon Goriely, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2023. CLIMB – curriculum learning for infant-inspired model building. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 112–127, Singapore. Association for Computational Linguistics.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Douglas L.T. Rohde and David C. Plaut. 1999. Language acquisition in the absence of explicit negative evidence: how important is starting small? *Cognition*, 72(1):67–109.

Rohan Saha, Abrar Fahim, Alona Fyshe, and Alex Murphy. 2024. Exploring curriculum learning for vision-language tasks: A study on small-scale multimodal training. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*. ArXiv:2410.15509.

Saber Sheybani, Himanshu Hansaria, Justin Wood, Linda Smith, and Zoran Tiganj. 2023. Curriculum learning with infant egocentric videos. *Advances in Neural Information Processing Systems*, 36:54199–54212.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, et al. 2025. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *ArXiv preprint*, abs/2502.14786.

Wai Keen Vong, Wentao Wang, A. Emin Orhan, and Brenden M. Lake. 2024. Grounded language acquisition through the eyes and ears of a single child. *Science*, 383(6682):504–511.

Shengao Wang et al. 2025. BabyVLM: Data-efficient pretraining of VLMs inspired by infant learning. *ArXiv preprint*, abs/2504.09426.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Luis Wiedmann, Andrés Marafioti, Lewis Tunstall, et al. 2025. nanoVLM: The simplest repository to train your VLM in pure PyTorch. https://github.com/huggingface/nanoVLM.

Chen Yu and Linda B Smith. 2012. Embodied attention and word learning by toddlers. *Cognition*, 125(2):244–262.

A Illustrated-page filtering.

Starting from 251,410 scanned pages, we apply three filters to identify pages containing substantial visual content beyond printed text. First, we sum the areas of the text boxes detected by PaddleOCR (Du et al., 2020) and divide by the total page area. We remove pages for which this estimated text coverage is at least 30%.

Second, we convert each page to grayscale and compute a normalized Canny edge map using thresholds of 100 and 200. We define the page's edge-content score as the standard deviation of this map and remove pages with scores below 0.2 as near-blank.

Finally, we create a temporary copy of each remaining page in which the detected text regions are filled with a mean color sampled from the surrounding image. We recompute the edge-content score and retain pages with scores of at least 0.3. This final step removes pages whose visual complexity is primarily attributable to printed text. The modified copies are used only for filtering; the original page images are retained for training. The complete procedure yields 46,721 illustrated pages.

B Corpus Examples

Figure 4 shows three source pages; Figure 5 shows the age-conditioned register realized for one page

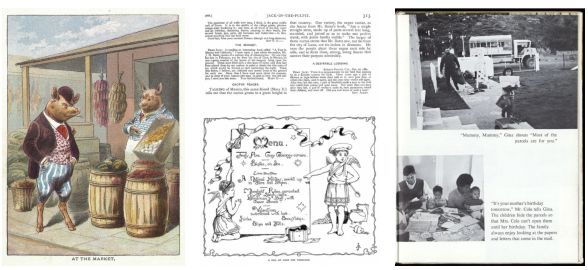


Figure 4: Illustrative sample pages from the corpus. Pages are shown for illustration only.

Captions generated for the *same* page, by band:

3–5: Pigs buy fruits at the market.
6–8: Pigs at the market: one pig sells fruits, the other talks.
9–12: Two well-dressed pigs in fancy clothes at a market.
12+: Two pigs, one stylishly dressed, the other selling fruits.

Reasoning, 3–5: *"Why so many parcels?" — "It's almost her mom's birthday… Gifts come on special days."*
VQA, 12+: *"Why is the cactus important?" — "The letter mentions a cactus that grows in Mexico… notable for its unusual height…"*

Figure 5: Age-conditioned synthetic annotations.

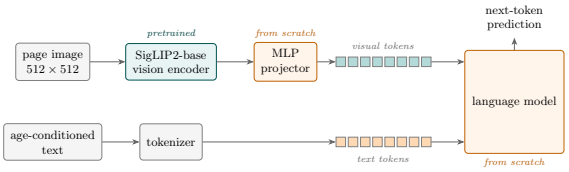


Figure 6: Model architecture. Only the vision encoder is pretrained; the projector and language model are trained from scratch on TinyLibrary. Components are subsequently trained jointly.

across all four bands.

C Training Details

Training uses bfloat16 precision on a single NVIDIA A100 40GB GPU, with a micro-batch size of 2 and eight gradient-accumulation steps. We use AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, weight decay 0.01) and clip gradients to a maximum norm of 1.0. Peak learning rates are 5.12×10^{-3} for the projector, 5×10^{-5} for the vision encoder, and 3×10^{-4} for the language model. Each learning rate uses a 3% linear warmup followed by cosine decay to 10% of its peak value. The step limit is set to 97% of the estimated capacity of each generated training sequence to prevent the data iterator from restarting. Final evaluations use the last checkpoint saved at a 250-step interval. Training takes approximately three hours for the 5-epoch budget and six hours for the 10-epoch budget.

D Five-Epoch Grid

Table 3 reports the complete results at the 5-epoch budget. Figure 7 shows the corresponding checkpoint trajectories.

E Seed-Matched Differences from Random Ordering

As the same seeds are used across conditions, each run can be compared with the random-ordering run

Order	Schedule	BLiMP	Supplement	EWoK	COMPS	Entity tracking
Random	—	53.2 ±0.9	46.0 ±1.6	50.1 ±0.7	50.4 ±0.3	25.0 ±6.2
Developmental	Staged	53.9 ±0.5	45.8 ±0.8	50.2 ±0.2	50.0 ±0.4	27.7 ±14.1
Developmental	Competence	53.3 ±1.7	46.6 ±0.7	50.2 ±0.8	50.1 ±0.4	32.6 ±13.7
Anti-developmental	Staged	54.4 ±1.9	45.6 ±0.8	50.0 ±0.9	49.9 ±0.1	26.2 ±14.8
Anti-developmental	Competence	52.2 ±0.9	45.8 ±2.3	50.0 ±0.6	50.2 ±0.5	30.8 ±5.6
Readability	Staged	53.1 ±0.7	47.6 ±3.3	50.3 ±0.7	50.1 ±0.6	38.5 ±3.6
Readability	Competence	52.8 ±1.5	45.6 ±3.5	49.3 ±0.3	50.0 ±0.3	28.9 ±11.7
Teacher perplexity	Staged	54.8 ±2.0	47.4 ±1.3	50.5 ±0.1	50.0 ±0.1	27.5 ±13.8
Teacher perplexity	Competence	**55.6 ±1.0**	47.9 ±1.8	50.8 ±0.2	50.0 ±0.7	20.6 ±5.6

Table 3: Zero-shot accuracy (%) at the 5-epoch budget, reported as mean ± standard deviation across three seeds. Bold indicates the highest mean BLiMP score.

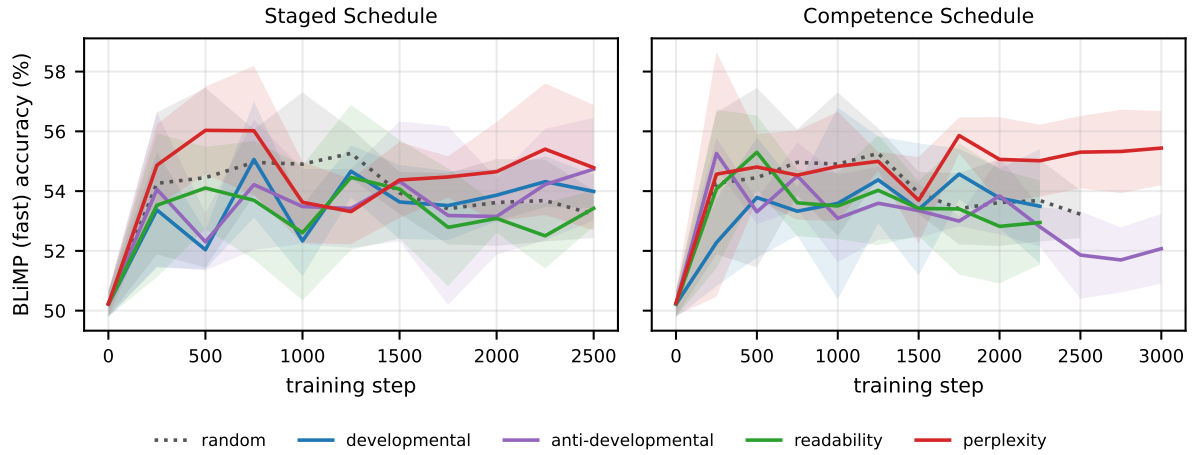


Figure 7: Fast-BLiMP accuracy over the 5-epoch budget, reported as the mean across three seeds with shading for ±1 standard deviation. The random curve is repeated in both panels. The late declines visible in several 10-epoch staged runs are less developed at this shorter budget.

using the same seed and training budget. Table 4 reports these paired BLiMP differences; positive values favor the listed condition. With only three seeds, we treat the differences as descriptive rather than as formal statistical tests.

F Robustness across Teacher Models

Figure 9 visualizes the mismatch between inferred age order and teacher NLL under the primary SmolLM2-135M teacher. To test whether this pattern depends on the teacher model, we rescore every sample using Qwen3-0.6B and Gemma-3-270M. We follow the same procedure for all three teachers, computing the mean per-token NLL of the target text conditioned on the prompt. As the models use different tokenizers, we compare their band-level patterns and sample rankings rather than their absolute NLL values.

All three teachers produce the same band-level pattern, with the highest NLL for the 3–5 band, the lowest NLL for the middle bands, and an increase for the 12+ band. Their sample-level rankings are also strongly correlated with the SmolLM2 ranking, with Spearman correlations of 0.91 for Qwen3 and 0.96 for Gemma-3. Although the samples assigned to the easiest and hardest quartiles are not identical across teachers. Agreement is lower for the easiest quartile, particularly between SmolLM2 and Qwen3.

Despite this agreement in estimated difficulty, the corresponding curriculum benefit does not transfer consistently. As reported in Table 2, the Gemma-derived ordering retains a smaller BLiMP advantage over random, whereas the Qwen3-derived ordering retains no advantage. Cross-teacher agreement in sample difficulty therefore does not guarantee equivalent training outcomes.

G Fixed-Partition Analysis

The random-block control tests whether the late decline observed in several staged conditions requires samples to be grouped by difficulty. Both ran-

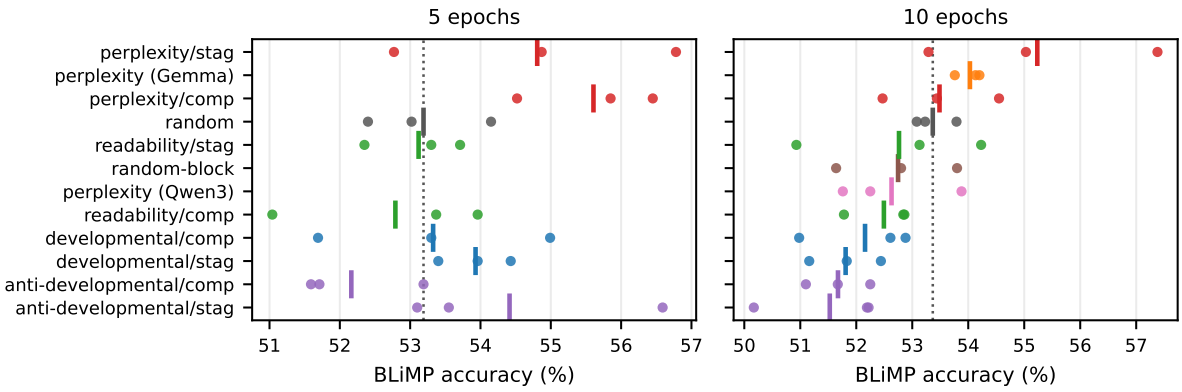


Figure 8: Final BLiMP scores by seed at the 5-epoch and 10-epoch budgets. Dots represent individual seeds, horizontal marks represent condition means, and dotted lines represent the random-ordering mean. The controls were trained only at the 10-epoch budget.

	5-epoch budget		10-epoch budget	
Condition	Seed differences	Mean	Seed differences	Mean
Developmental / Staged	+2.0, +0.4, −0.2	+0.74	−2.6, −0.6, −1.4	−1.56
Developmental / Competence	−0.7, +0.3, +0.8	+0.14	−0.9, −2.1, −0.6	−1.21
Anti-developmental / Staged	+0.7, +0.5, +2.4	+1.22	−1.6, −2.9, −1.0	−1.84
Anti-developmental / Competence	−0.7, −1.4, −1.0	−1.03	−2.7, −0.8, −1.6	−1.69
Readability / Staged	+0.9, +0.7, −1.8	−0.07	−2.9, +1.2, −0.1	−0.60
Readability / Competence	+1.6, −2.0, −0.8	−0.40	−2.0, −0.2, −0.4	−0.87
Teacher perplexity / Staged	+0.4, +1.9, +2.6	+1.62	−0.5, +2.0, +4.2	+1.87
Teacher perplexity / Competence	+3.5, +1.5, +2.3	+2.42	+0.8, +0.4, −0.8	+0.12
Random-block / Staged	—	—	−1.0, −1.4, +0.6	−0.62
Teacher perplexity (Gemma) / Staged	—	—	+0.4, +1.1, +0.5	+0.66
Teacher perplexity (Qwen3) / Staged	—	—	−2.0, +0.8, −1.0	−0.74

Table 4: Seed-matched BLiMP differences from random ordering, in percentage points. The mean is computed across the three paired differences.

	SmolLM2 135M	Qwen3 0.6B	Gemma-3 270M
NLL, band 3–5	2.86	3.14	2.96
NLL, band 6–8	2.54	2.67	2.62
NLL, band 9–12	2.52	2.57	2.57
NLL, band 12+	2.75	2.71	2.76
Spearman ρ vs. SmolLM2	—	0.91	0.96
Easiest-quartile overlap	—	0.70	0.82
Hardest-quartile overlap	—	0.91	0.92

Table 5: Mean per-token NLL by inferred age band and sample-level agreement with the SmolLM2-derived ordering. All three teachers reproduce the same band-level pattern, but their sample rankings are not identical.

dom and random-block ordering present every sample once per epoch without using a content-based ranking. Random ordering reshuffles the complete corpus each epoch, whereas random-block assigns samples to four fixed groups matching the token masses of the age-band stages and reuses those assignments across epochs. The comparison therefore isolates the association between repeated group membership and training dynamics.

Because peak-to-final differences can exaggerate decline by selecting a noisy maximum, we also calculate two measures that do not depend on identifying a peak. The first is the linear Fast-BLiMP slope over the second half of training, expressed in points per 1,000 steps. The second is the difference between mean performance during the final fifth of training and mean performance from steps 500–1,250.

Random-block has a more negative mean slope and late–early difference than the fully reshuffled random baseline. This pattern is consistent with repeated group membership contributing to late degradation even when the groups contain randomly selected samples.

The effect is not uniform across fixed-partition conditions. Anti-developmental ordering also declines, whereas teacher-perplexity ordering re-

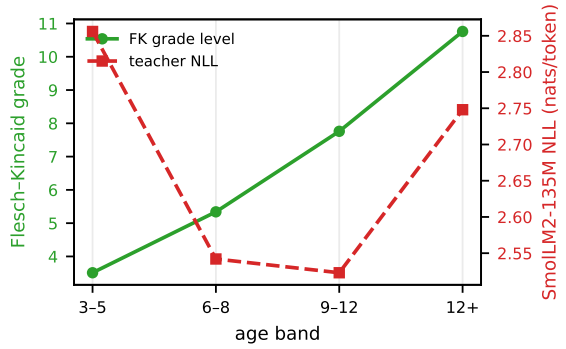


Figure 9: Readability and teacher-model perplexity encode different age-band progressions. Flesch–Kincaid grade increases monotonically with inferred age, whereas mean per-token NLL under the primary SmolLM2-135M teacher is highest for the youngest and oldest bands and lowest for the middle bands.

Condition	Late-half slope/1k	Late−early
Random	+0.03 ±0.33	-0.32 ±0.55
Random-block	-0.50 ±0.42	-1.38 ±0.97
Teacher perplexity	+0.06 ±0.65	+0.83 ±2.08
Developmental	-0.22 ±0.39	-1.13 ±1.11
Anti-developmental	-0.45 ±0.14	-2.94 ±0.86
Readability	+0.01 ±0.81	-0.60 ±2.62

Table 6: Late-training Fast-BLiMP dynamics, reported as mean ± standard deviation over three seeds. Late-half slope is calculated over the second half of training; late−early compares the final fifth of training with steps 500–1,250.

mains approximately flat; readability has a near-zero mean slope but substantial variation across seeds. Fixed-partition reuse is therefore associated with late degradation in some conditions, but it is not sufficient to explain when degradation occurs.

H Teacher-NLL Quartiles Largely Track Annotation Format

The annotation formats differ substantially in predictability under the SmolLM2-135M teacher. Captions have a mean per-token NLL of 3.64, compared with 2.25 for visual question answering and 2.29 for reasoning. Consequently, the equal-sized NLL quartiles largely separate samples by annotation format. Under ascending teacher-NLL ordering, VQA and reasoning samples are therefore presented predominantly earlier in training, while captions are presented predominantly later.

I Entity-Tracking Results

Entity tracking does not provide a stable basis for comparing the curriculum orderings. On the full evaluation, the untrained reference scores 41.8 ±0.1, whereas the trained models score between 17.1 and 36.7 at the 10-epoch budget. Performance also varies substantially across seeds, with standard deviations as large as 15.2 points.

NLL quartile	Caption	VQA	Reason.	Med. tok.
Q1 (easiest)	0.3%	52.1%	47.6%	211
Q2	0.7%	51.9%	47.5%	289
Q3	31.6%	30.0%	38.4%	215
Q4 (hardest)	99.6%	0.1%	0.3%	73

Table 7: Annotation-format composition of the equal-sized teacher-NLL quartiles. The two lowest-NLL quartiles consist almost entirely of VQA and reasoning samples, whereas the highest-NLL quartile consists almost entirely of captions.

The checkpoint trajectories in Figure 10 show that scores generally decrease during training, but our experiments do not establish why. The pattern could reflect properties of the evaluation at this model scale, the architecture, or the training data. Without a known strong reference model under the same evaluation protocol, the untrained score cannot be interpreted as evidence of entity-tracking ability. We therefore do not use this metric to draw conclusions about the relative effectiveness of the curriculum orderings.

J Multimodal and Text-Only Controls

Multimodal evaluation. Following the official BabyLM evaluation pipeline, we rank candidate answers by their conditional log-likelihood given the image. This evaluation uses the complete VLM. VQA contains 25,230 questions with approximately eight candidate answers each, while Winoground contains 746 two-way comparisons.

Winoground remains near chance across all orderings. VQA performance exceeds chance but varies substantially across seeds. Relative to seed-matched random runs, the staged differences are $+5.7 \pm 4.9$ for developmental, $+8.6 \pm 5.5$ for anti-developmental, $+2.8 \pm 2.0$ for readability, $+8.0 \pm 5.3$ for teacher perplexity, and $+4.9 \pm 3.4$ for random-block.

Although these mean differences are positive, their ranking does not follow developmental order, teacher perplexity, or the BLiMP results. Together with the chance-level Winoground scores and substantial seed variation, they do not provide coherent evidence that a particular ordering improves multimodal grounding.

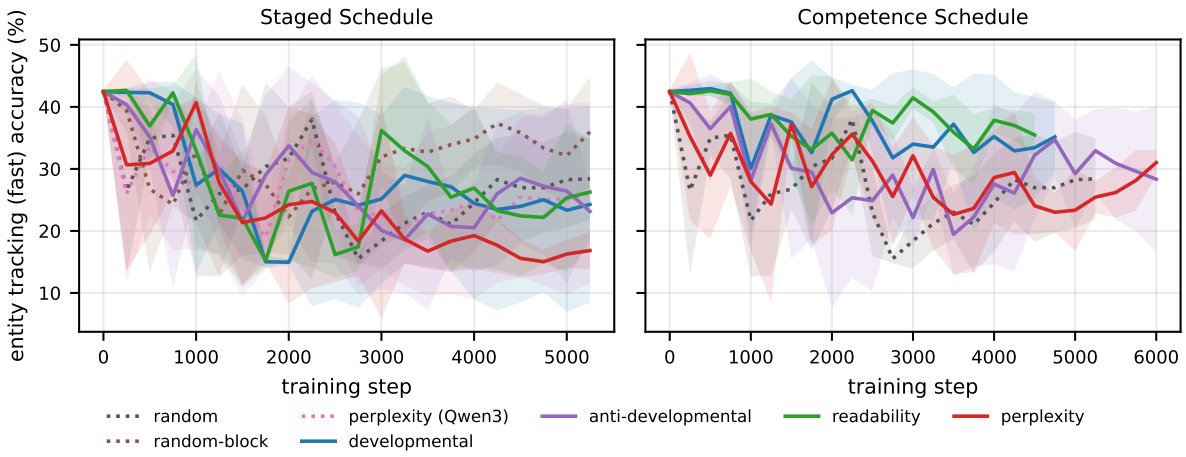


Figure 10: Fast entity-tracking accuracy over 10 epochs, reported as the mean across three seeds with shading for ± 1 standard deviation. Performance generally decreases from its initial level and varies substantially across seeds.

Ordering and schedule	VQA	Winoground
Random	20.3 $\pm$1.2	52.2 $\pm$1.7
Random-block / staged	25.2 $\pm$4.6	52.5 $\pm$1.1
Developmental / staged	26.0 $\pm$5.6	51.5 $\pm$0.5
Anti-developmental / staged	28.9 $\pm$4.8	50.8 $\pm$0.9
Readability / staged	23.1 $\pm$2.3	52.0 $\pm$1.9
Teacher PPL / staged	28.3 $\pm$4.8	52.6 $\pm$0.4
Qwen3 PPL / staged	25.5 $\pm$2.5	51.7 $\pm$2.6
Gemma PPL / staged	23.3 $\pm$0.3	51.6 $\pm$1.2
Developmental / competence	21.8 $\pm$8.7	52.4 $\pm$1.2
Anti-developmental / competence	23.8 $\pm$6.9	51.5 $\pm$1.9
Readability / competence	31.3 $\pm$4.7	49.2 $\pm$2.0
Teacher PPL / competence	30.6 $\pm$7.5	50.8 $\pm$0.5
Chance	12.5	50.0

Table 8: Multimodal zero-shot accuracy (%) after 10 epochs, reported as mean $\pm$ standard deviation over three seeds.

Text-only control. We train text-only models under developmental, teacher-perplexity, and random ordering using the same decoder architecture and annotation targets but without image tokens. The training budget is adjusted to preserve the same exposure to annotation text. After 10 epochs, the text-only models score 53.7 BLiMP under developmental ordering, 53.2 under teacher-perplexity ordering, and 52.9 under random ordering. The corresponding seed-paired VLM-minus-text-only differences are -1.9 ± 0.6, $+2.1 \pm 2.7$, and $+0.5 \pm 1.1$. Because the direction varies across orderings, the control provides no evidence of a uniform benefit or penalty from visual inputs. It does not establish modality independence: the teacher-perplexity advantage over random decreases from 1.87 points in the VLM runs to 0.27 points in the text-only runs.

Not All Irregularity Is Equal: Causally Isolating a Rare Failure Mode in Japanese Morphological Inflection

Wen Zhang
wenzhang0222@gmail.com

Abstract

Neural morphological generation systems often achieve high aggregate accuracy on benchmark datasets, yet such performance can conceal systematic errors clustered in rare morphological subclasses. We present an orthography-aware diagnosis of Japanese past-tense verb inflection, treating *hiragana* not merely as a transcriptional medium but as a representational system that encodes morphophonological structure. Using two character-level Transformer architectures evaluated across five random seeds, we show that although both systems exceed 97% aggregate accuracy, a single structurally specific irregular subtype, verbs whose stems end in /e/ and require gemination before the past-tense suffix and make up fewer than 1% of the data, accounts for a disproportionate 30–43% share of residual errors and contributes roughly 34–48× its prevalence to total errors. We then move from diagnosis to causal isolation: controlled ablation experiments show that removing this subtype alone produces larger accuracy gains than removing all irregular verbs combined. These findings indicate that error concentration in neural morphological learning is not driven by irregularity *per se*, but by the interaction between extreme low-frequency morphological patterns and specific orthographic processes. We argue that morphological evaluation should incorporate fine-grained subclass analysis, and discuss implications for data-efficient, developmentally plausible language model pretraining.[1]

1 Introduction

Japanese verbs are written in a hybrid system of *kanji*, *hiragana*, and *katakana*. While verb stems may appear in either *kanji* or *hiragana*, inflectional suffixes are consistently marked in *hiragana*. For example, the verb 書く/かく *kaku* becomes 書いた/かいた *kaita* in the past tense, where た *ta* marks the past-tense suffix. *Hiragana* is a moraic script, representing phonological timing units and marking processes such as gemination (the small っ *tsu*) and vowel lengthening that are central to Japanese morphophonology (Kubozono et al., 2009; Labrune, 2012). Following the view that writing systems are structured representational systems rather than transparent encodings of speech (Sproat, 2000; Daniels and Bright, 1996), we treat *hiragana* as a linguistically meaningful layer that may shape neural model generalization, and use it to eliminate confounds from *kanji* homography by restricting all forms to *hiragana*.

Most evaluations of Japanese morphological generation report aggregate exact-match accuracy (Cotterell et al., 2016; Vylomova et al., 2020; Goldman et al., 2023), often exceeding 95% for high-resource languages such as Japanese. However, high aggregate accuracy does not necessarily indicate robust generalization, and can obscure systematic weaknesses concentrated in rare morphological subclasses (Goldman et al., 2022). This is a manifestation of the "long tail" challenge in morphological learning (Cotterell et al., 2018), where models favor high-frequency regularities over infrequent but productive rules.

In this paper we combine two complementary methodologies to study this phenomenon in Japanese past-tense inflection. First, we conduct an orthography-aware **diagnosis**, using a fine-grained error taxonomy and subgroup-level accuracy analysis, aggregated across five random seeds and two architectures, to characterize *where* models fail. Second, we conduct a **causal isolation** study, using controlled ablation experiments that selectively remove irregular verb subtypes, to test *whether* the identified subtype is actually responsible for destabilizing generalization, and not irregularity in general. Combining them lets us both locate the failure and test its cause.

Our contributions are:

[1] Code and data available at https://github.com/wenzhang0222/not-all-irregularity.

Proceedings of the Second BabyLM Workshop, pages 348–357
October 29, 2026 ©2026 Association for Computational Linguistics

- A structural subgroup evaluation framework for Japanese morphological inflection, combining a six-category orthography-aware error taxonomy with a Disparity Ratio metric for quantifying subgroup-level error concentration.

- A multi-seed, two-architecture empirical analysis showing that a single low-frequency irregular subtype (<1% of the data) accounts for 30–43% of residual errors.

- Controlled ablation experiments showing that selective removal of this subtype alone produces larger accuracy gains than removing all irregular verbs combined, suggesting that structural position, not frequency alone, is the primary driver of the effect.

1.1 Japanese Past-Tense Morphology

Japanese verbs are traditionally classified by inflectional behavior. Past-tense formation involves the suffix た *-ta* and morphophonological alternations such as consonant mutation, gemination, and vowel changes, all of which are systematically reflected in *hiragana* orthography. ***Godan* (*u*-) verbs** form the past tense through suffix-conditioned stem alternations, often involving consonant changes and gemination, e.g., かく *kaku* 'to write' → かいた *kaita* 'wrote'. ***Ichidan* (*ru*-) verbs** exhibit stable stems, with the past tense formed via direct suffix attachment, e.g., たべる *taberu* 'to eat' → たべた *tabeta* 'ate'. **Canonical irregular verbs** are a small closed class including する *suru* 'to do' → した *shita* 'did' and くる *kuru* 'to come' → きた *kita* 'came'. Because these alternations are encoded transparently in *hiragana*, past-tense inflection is a dense testbed for studying morphophonological learning at the character level, without requiring lexical segmentation.

2 Related Work

Morphological Learning and Generalization. Neural morphological inflection has been driven largely by Transformer-based encoder–decoder architectures (Vaswani et al., 2017), achieving high aggregate accuracy on SIGMORPHON benchmarks (Vylomova et al., 2020; Goldman et al., 2023). There is growing recognition that aggregate metrics obscure performance disparities across morphological classes (Goldman et al., 2022). We contribute to this line of work by showing how a single structurally idiosyncratic subclass can act as a stress test for neural generalization, and by testing this causally. This work synthesizes and extends our prior analyses of Japanese morphological inflection (Zhang, 2026a,b), combining orthography-aware error diagnosis with causal ablation methodology.

Subgroup Analysis and Fairness in NLP. The challenge of modeling rare data is not unique to morphology; it mirrors broader concerns about subgroup disparity in neural models, where models often prioritize majority-group performance at the expense of underrepresented strata (Sagawa et al., 2020; Buolamwini and Gebru, 2018). We adopt diagnostic tools from this literature, in particular a Disparity Ratio, to quantify error concentration across morphological subtypes, while noting that our "subgroups" are defined structurally and not socially (Blodgett et al., 2020).

Inductive Bias and Irregularity. The learnability of irregular morphology has long been central to debates between rule-based and connectionist accounts of language processing (Rumelhart and McClelland, 1986; Pinker, 1994). Our findings align with evidence that neural models can capture systematic patterns of morphological irregularity (Wu et al., 2019; Kakolu Ramarao et al., 2025), but extend this literature by showing that not all irregular forms are equally destabilizing. Orthographic transparency and structural frequency, not irregular status alone, predict where models fail.

Evaluation Metrics and Orthographic Bias. Recent work shows that normalized crosslingual evaluation metrics introduce systematic biases rooted in tokenization, encoding, and orthographic differences (Yang et al., 2026). Our findings offer a parallel concern at the subtype level within a single language: aggregate accuracy within Japanese morphological evaluation similarly obscures systematic failures clustered in structurally specific subclasses, suggesting that subgroup-aware diagnostics are necessary at both the crosslingual and intra-paradigm levels.

3 Data

We use a Japanese verb inflection dataset in standard morphological transduction format, following the setup of Vylomova et al. (2020) and Goldman et al. (2023). All forms are converted to *hiragana* to maintain orthographic consistency and

eliminate confounds introduced by *kanji* homography (Zhang, 2023). Each instance consists of three tab-separated fields: lemma, target form, and a placeholder tag (no explicit morphosyntactic features are provided, so the model must learn the lemma-to-form mapping directly), e.g. ねがえる ねがえった _.

3.1 Verb Classification

Verbs are classified according to traditional Japanese conjugation classes, refined to capture orthography-sensitive variation. Canonical irregular verbs (する, くる; Type 3) and polysemous lemmas with multiple inflected forms are excluded to maintain a one-to-one lemma–form mapping. The remaining verbs fall into three types:

- **Type 1 (Godan)**: regular u-verbs, e.g. かく *kaku* → かいた *kaita*. Count: 2,503.
- **Type 2 (Ichidan)**: regular ru-verbs, e.g. たべる *taberu* → たべた *tabeta*. Count: 1,298.
- **Type 4** (irregular): further subdivided into three subtypes:
  - **Type 4-1**: stem-final /i/ + gemination, e.g. まじる *majiru* → まじった *majitta*. Count: 119.
  - **Type 4-2**: stem-final /e/ + gemination, e.g. あきれかえる *akirekaeru* → あきれかえった *akirekaetta*. Count: 37.
  - **Type 4-3**: localized idiosyncratic deviations, e.g. いく *iku* → いった *itta*. Count: 1.

Table 1 summarizes dataset composition. Type 4-2 is the smallest structurally coherent subclass, comprising only 0.9% of the data; the full verb list appears in Appendix A.

Verb Type	Count	Prop. (%)
All verbs	3,958	100
Type 1 (Godan)	2,503	63.2
Type 2 (Ichidan)	1,298	32.8
Type 4 (irregular)	157	4.0
4-1 (/i/ + gemination)	119	3.0
4-2 (/e/ + gemination)	37	0.9
4-3 (localized)	1	0.03

Table 1: Dataset statistics by verb type. Canonical irregulars (する, くる; Type 3) are excluded, as described in §3.1.

4 Models

We evaluate two character-level Transformer encoder–decoder architectures for Japanese past-tense inflection, both operating over *hiragana* strings and generating inflected forms autoregressively. The first is a **Pointer-Generator Transformer** (henceforth **PGT**), incorporating a pointer-generator copying mechanism (See et al., 2017) with attention over the input lemma, trained and evaluated on the morphological inflection data (Vylomova et al., 2020). The second follows the architecture of Goldman et al. (2023), a **Lemma-Split Transformer** (henceforth **LST**) that additionally uses a lemma-split training and evaluation regime, preventing a lemma from appearing in both training and test data and improving generalization to unseen lemmas.

5 Experimental Setup

5.1 Training Regime

Both models are trained with cross-entropy loss and teacher forcing, using the default hyperparameter configurations from their respective shared-task baselines (Vylomova et al., 2020; Goldman et al., 2023), optimized with Adam (Kingma and Ba, 2015) and standard Transformer learning-rate scheduling (Vaswani et al., 2017).

5.2 Multi-Seed Baseline Evaluation

Neural model performance is sensitive to random initialization, data shuffling, and optimization dynamics (Reimers and Gurevych, 2017). To reduce selective-reporting bias, we train each model on the full dataset five times with different random seeds, using an 80/10/10 train/development/test split. The error taxonomy and quantitative error-distribution analyses (§6.2–6.4) are aggregated across all five seeds. The controlled ablation experiments (§6.7) involve eight training conditions per model and are conducted on the primary seed.

5.3 Controlled Ablation Conditions

To causally test the contribution of individual irregular subtypes, we train both architectures under eight controlled data conditions: the full dataset; regular verbs only (Types 1–2); the full dataset minus one irregular subtype (removing 4-1, 4-2, or 4-3 individually); and the full dataset minus each pairwise combination of subtypes. For every con-

dition, the same verb types are removed from both training and test data.

5.4 Evaluation Metrics

We report **exact-match accuracy** at the lemma level (Cotterell et al., 2017; Goldman et al., 2023); **subgroup accuracy**, computed separately per verb type (Kann and Schütze, 2016; Makarov and Clematide, 2018; Vylomova et al., 2020); and the **Disparity Ratio** for a subgroup g,

$$DR_g = \frac{\text{Error Share}_g}{\text{Data Share}_g},$$

where a value greater than 1 indicates a disproportionately high error burden relative to prevalence, following diagnostic tools from subgroup fairness analysis (Sagawa et al., 2020; Buolamwini and Gebru, 2018; Blodgett et al., 2020).

6 Results

6.1 Baseline Performance

On the primary run, both systems achieve high aggregate accuracy on the full dataset: 97.98% (PGT) and 97.73% (LST). Averaged across all five seeds, accuracy is 97.17% ± 0.65 (PGT) and 96.97% ± 0.82 (LST) (mean ± sample standard deviation), confirming that high aggregate accuracy is a stable property of both architectures. Despite this overall stability, errors cluster in specific low-frequency subclasses, as we show next.

6.2 Error Taxonomy

We defined a taxonomy of six orthography- and morphophonology-sensitive failure modes (Table 2): gemination omission, gemination insertion, phonological substitution, morpheme boundary errors, character recognition (UNK) errors, and compound verb structural errors. We used this taxonomy to manually classify all residual errors (Table 3). This taxonomy highlights that residual failures are systematic, not random noise. For example, failures to insert, or spuriously insert, the small っ reflect specific difficulty with consonant doubling.

6.3 Quantitative Error Distribution

Aggregated across five seeds, we observe 104 total errors, 53 from the PGT system and 51 from the LST system. Table 3 reports the six observed error types. Gemination-related errors (omission and insertion combined) dominate residual failures, accounting for 75.5% (PGT) and 80.3% (LST) of all errors, with omissions far more frequent than insertions in both systems.

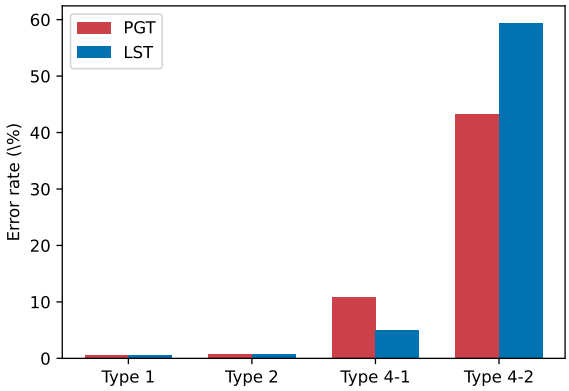


Figure 1: Per-type error rate (errors / total items of that type) for PGT and LST systems. Type 4-2 shows an error rate of 43% (PGT) and 60% (LST), more than 70 times the rate of Type 1 verbs.

6.4 Verb-Class Asymmetry and Disparity Ratios

Table 4 breaks the same errors down by verb type. Type 4-2 verbs are massively overrepresented, accounting for 30.2% (PGT) and 43.1% (LST) of all errors while comprising just 0.9% of the dataset. Type 4-1 is also somewhat overrepresented, while Type 1 and Type 2 show error rates below their dataset share. Figure 1 shows the per-type error rate directly, making the disproportionate difficulty of Type 4-2 visually apparent.

Table 5 converts this into Disparity Ratios. Type 4-2 contributes 34–48× its proportional representation to total errors, as visualized in Figure 2, the largest disparity of any subclass in either system. This pattern is consistent across both architectures.

To confirm this over-representation is not an artifact of small counts, we tested it directly. Under the null hypothesis that a verb's probability of erring is independent of its type (i.e., errors are distributed in proportion to each type's data share), we would expect Type 4-2 to account for only 0.9% of errors. A one-sided exact binomial test rejects this null decisively for both systems (PGT: 16/53 errors, $p = 3.6 \times 10^{-20}$; LST: 22/51 errors, $p = 2.7 \times 10^{-31}$).

6.5 Cross-Model Consistency

Despite architectural differences, error patterns are broadly consistent across the PGT and LST systems. Gemination-related failures dominate in both (75.5%/80.3%), Type 4-2 is the most overrep-

Error Type	Description	Orthographic/Phonological Property
Gemination omission	Failure to insert small っ	Consonant doubling
Gemination insertion	Spurious insertion of っ	Consonant doubling
Phonological substitution	Incorrect consonant or vowel in stem	Sound alternation
Morpheme boundary error	Misaligned suffix attachment	Boundary detection
Character recognition error	Generation of UNK symbol	Encoding/representation
Compound verb error	Structural collapse in compound verbs	Compound segmentation

Table 2: Error taxonomy for *hiragana*-based inflection.

Error Type	PGT	LST
Gemination omission	33 (62.3%)	32 (62.7%)
Gemination insertion	7 (13.2%)	9 (17.6%)
Phonological substitution	6 (11.3%)	4 (7.8%)
Morpheme boundary	3 (5.7%)	4 (7.8%)
Character recognition (UNK)	1 (1.9%)	1 (2.0%)
Compound verb error	3 (5.7%)	1 (2.0%)
Total	**53**	**51**

Table 3: Quantitative error distribution by error type.

Verb Type	PGT Err.	LST Err.	Data %
Type 1	15 (28.3%)	13 (25.5%)	63.2
Type 2	9 (17.0%)	10 (19.6%)	32.8
Type 4-1	13 (24.5%)	6 (11.8%)	3.0
Type 4-2	**16 (30.2%)**	**22 (43.1%)**	**0.9**
Type 4-3	0	0	0.03
Total	**53**	**51**	**100**

Table 4: Error distribution by verb type against dataset share.

Verb Type	*DR* (PGT)	*DR* (LST)
Type 1	0.45	0.40
Type 2	0.52	0.60
Type 4-1	8.17	3.93
Type 4-2	**33.6**	**47.9**

Table 5: Disparity Ratio (error share / data share) by verb type. Values >1 indicate error over-representation relative to prevalence.

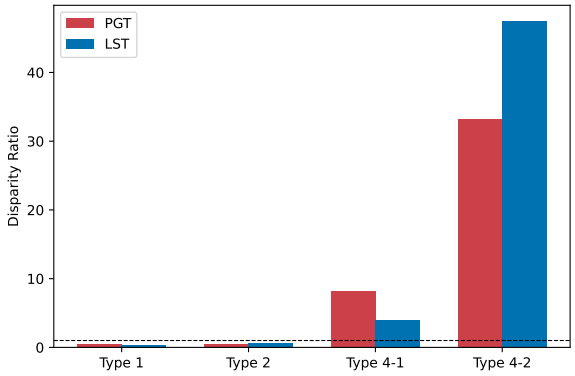


Figure 2: Disparity Ratio by verb type for PGT and LST systems. Type 4-2 shows a dramatically higher disparity ratio than all other subtypes, contributing 34–48× its proportional share to total errors.

resented subclass in both (Table 4), and an identical single UNK error appears in both systems' outputs (つっぷした → つ<UNK>した). This consistency, together with the seed-level stability reported in §6.1, suggests the failures reflect systematic properties of orthographic representation rather than idiosyncrasies of a single model.

6.6 Qualitative Error Patterns

The dominant Type 4-2 failure is omission of required gemination before a stem-final /e/, e.g. あきれかえった *akirekaetta* → あきれかえた *akirekaeta*. Spurious gemination insertion occurs mainly in Type 2 verbs, e.g. おきた *okita* → おきった *okitta*. Morpheme boundary errors arise when compound boundaries are unmarked in *hiragana*, e.g. ほめたたえた *hometataeta* → ほめたえた *hometaeta*.

6.7 Causal Ablation: Isolating Type 4-2

The analyses above are descriptive: they show *where* errors concentrate, but not whether removing the implicated subtype actually improves generalization more than removing irregularity broadly. Table 6 and Figure 3 summarize the accuracy gain over the full-data baseline for each condition. Table 7 reports test accuracy under all eight ablation conditions. Because the same verb types are removed from both training and test across all conditions, accuracy differences between conditions reflect differences in which subtypes destabilize learning of the remaining vocabulary, not differences in test set composition.

For the **LST system**, removing only Type 4-

Condition	PGT gain	LST gain
−4 (all irregulars)	+0.97	+1.22
−4-1	+0.98	+0.45
−4-2	**+1.00**	**+2.02**
−4-3	+0.76	+1.01
−4-1,4-2	+0.97	+2.01
−4-1,4-3	+0.72	+0.71
−4-2,4-3	+0.49	+1.00

Table 6: Accuracy gain over full-data baseline for each ablation condition. Boldface marks the single-subtype condition producing the largest improvement in each system.

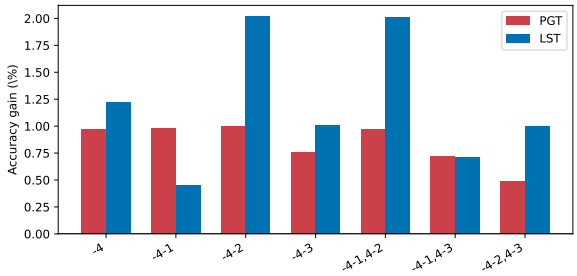


Figure 3: Accuracy gains from Table 6 visualized. The −4-2 condition stands out clearly in both systems.

2 yields the largest observed gain: 97.73% → 99.75% (+2.02), corresponding to an approximately 89% reduction in error rate. The gain is substantially larger than removing all irregular verbs (+1.22) and matches or exceeds every other single- or paired-subtype removal. For the **PGT system**, removing Type 4-2 alone is still the single best condition (97.98% → 98.98%, +1.00).

In both systems, removing the full irregular set (Type 4) never yields the maximal accuracy. Some irregular subtypes (4-1, 4-3) can be retained without cost: in the LST system, removing only Type 4-2 while retaining 4-1 and 4-3 (condition −4-2) achieves 99.75%, and additionally removing 4-1 as well (condition −4-1,4-2) performs nearly as well at 99.74%. This indicates that the relevant factor is not irregular status *per se*, but the specific structural configuration instantiated by Type 4-2.

7 Discussion

Taken together, the diagnostic and causal results support a consistent picture. Irregularity is not uniformly detrimental to neural morphological learning. A single low-frequency subtype, verbs requiring gemination after a stem-final /e/, is structurally distinct from other irregulars. It accounts for a disproportionate share of errors (34–48× its data share) and, for LST, is causally responsible

Condition	PGT	LST
Full (1–4)	97.98	97.73
−4 (all irregulars)	98.95	98.95
−4-1	98.96	98.18
−4-2	**98.98**	**99.75**
−4-3	98.74	98.74
−4-1,4-2	98.95	99.74
−4-1,4-3	98.70	98.44
−4-2,4-3	98.47	98.73

Table 7: Test-set exact-match accuracy (%) under each ablation condition. Bold marks the best condition per system.

for most of the achievable accuracy gain from irregular-verb ablation.

The picture is more nuanced for the PGT architecture. Type 4-2 removal is directionally consistent with the LST result, remaining the best single condition, but the margin over alternative ablations is small. We view this as a genuine and informative asymmetry and not a weakness to be smoothed over: the diagnostic finding, that Type 4-2 is where errors concentrate, is robust across architectures and seeds, while the strength of the causal evidence for removing Type 4-2 specifically varies between them.

Several factors likely contribute to Type 4-2's structural distinctiveness. First, consonant gemination (small っ) introduces an additional mora explicitly represented in *hiragana*, altering a word's rhythmic structure in a way character-level models must generate in the correct context (Pimentel et al., 2020). Second, orthographic form alone does not encode all conditioning factors (e.g., lexical frequency), so some verbs remain more error-prone than surface similarity would suggest. Third, at under 1% of training data, Type 4-2 instances are too sparse for models to reliably learn the conditioning environment, while still being frequent enough, and structurally regular enough, to be systematically attempted rather than simply memorized or ignored. This contrasts with Type 4-3, whose single instance produces zero errors in both systems, likely because it is either trivially memorized or entirely absent from test predictions in a way that does not surface as a systematic failure.

This has a direct implication for evaluation practice. Aggregate accuracy near 98% in both our systems obscures an effect visible only under subtype-level analysis; we suggest the same is plausibly true of aggregate scores on existing benchmarks. Per-item predictions can be regrouped by struc-

tural subclass and scored with the Disparity Ratio metric introduced in §5.4. Evaluation protocols for data-efficient pretraining may therefore benefit from subgroup-aware diagnostics of the kind proposed here, and not aggregate accuracy alone.

Future Directions. The structural confusability effect identified here raises two natural extensions. First, whether the same effect appears when models are evaluated zero-shot at scale, testing whether frontier language models show the same Type 4-2 sensitivity or whether scale eliminates the inductive bias. Second, whether the Disparity Ratio metric generalizes beyond Japanese to other morphologically rich languages, testing whether structural confusability constitutes a domain-general constraint on morphological learning.

8 Conclusion

We presented a combined diagnostic and causal analysis of Japanese past-tense inflection. Aggregating over five seeds and two architectures, we showed that a single irregular subtype comprising under 1% of the data accounts for 30–43% of residual errors (a 34–48$\times$ disparity ratio), concentrated in gemination-related failures. Controlled ablation experiments show that removing only this subtype improves generalization more than removing all irregular verbs. These results demonstrate that high aggregate accuracy can mask structurally localized failures, that not all irregularity is equally destabilizing to neural generalization, and that fine-grained, subgroup-aware evaluation is necessary to reveal and causally test such effects. We argue this evaluation practice is particularly relevant to sample-efficient, developmentally plausible language modeling. Future work will test whether the same structural confusability effects appear in large language models, asking whether scale alone is sufficient to overcome the inductive biases identified here.

Limitations

Several limitations qualify our findings. First, our ablation experiments were each run once per condition; replication across multiple training runs would strengthen the causal claim. Second, our study focuses on a single language, a single morphological paradigm, and two Transformer-based architectures. Cross-linguistic validation is necessary to determine whether comparable low-frequency structural subtypes produce similar disparities in other morphologically rich languages, and whether larger pretrained models exhibit different sensitivity to the patterns we identify. Third, our analysis is correlational with respect to the specific mechanism (frequency vs. structural complexity vs. their interaction) underlying Type 4-2's distinctiveness; the ablation results support a causal role for this subtype's presence in training data, but do not isolate frequency from structural complexity as independent factors. We leave controlled frequency-matched comparisons, and evaluation under alternative input representations (e.g., IPA or romanization), to future work.

Ethics Statement

This work involves no sensitive data, human subjects, or personally identifiable information. Data and models are derived from publicly available databases and SIGMORPHON shared-task resources.

Acknowledgments

We thank the BabyLM 2026 workshop for their thoughtful feedback, which strengthened this work.

References

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 77–91. PMLR.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

Peter T. Daniels and William Bright, editors. 1996. *The World's Writing Systems*. Oxford University Press, Oxford.

Omer Goldman, Khuyagbaatar Batsuren, Salam Khalifa, Aryaman Arora, Garrett Nicolai, Reut Tsarfaty, and Ekaterina Vylomova. 2023. SIGMORPHON–UniMorph 2023 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 20th SIGMORPHON workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 117–125, Toronto, Canada. Association for Computational Linguistics.

Omer Goldman, David Guriel, and Reut Tsarfaty. 2022. (un)solving morphological inflection: Lemma overlap artificially inflates models' performance. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 864–870, Dublin, Ireland. Association for Computational Linguistics.

Akhilesh Kakolu Ramarao, Kevin Tang, and Dinah Baer-Henney. 2025. Frequency matters: Modeling irregular morphological patterns in Spanish with transformers. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4474–4489, Vienna, Austria. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 555–560, Berlin, Germany. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Haruo Kubozono, Junko Ito, and Armin Mester. 2009. Consonant gemination in japanese loanword phonology. In The Linguistic Society of Korea, editor, *Current Issues in Unity and Diversity of Languages: Collection of Papers Selected from the 18th International Congress of Linguists [CIL 18]*, pages 953–973. Dongam Publishing Co., Republic of Korea.

Laurence Labrune. 2012. *The Phonology of Japanese*. Oxford University Press.

Peter Makarov and Simon Clematide. 2018. Imitation learning for neural morphological string transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2877–2882, Brussels, Belgium. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Steven Pinker. 1994. *The Language Instinct*. William Morrow and Company.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

David E. Rumelhart and James L. McClelland. 1986. On learning the past tenses of english verbs. In James L. McClelland, David E. Rumelhart, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2, pages 216–271. MIT Press.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations (ICLR)*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Richard Sproat. 2000. *A Computational Theory of Writing Systems*. Cambridge University Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Maria Ponti, Rowan Hall Maudslay, Ran Zmigrod, Josef Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrew Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, and 9 others. 2020. SIGMORPHON 2020 shared task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 1–39, Online. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Timothy O'Donnell. 2019. Morphological irregularity correlates with frequency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5117–5126, Florence, Italy. Association for Computational Linguistics.

Xiulin Yang, Ethan Gotlieb Wilcox, and Catherine Arnett. 2026. Apples to apples? Towards comparable crosslingual language model evaluation. *arXiv preprint arXiv:2608.25089*.

Wen Zhang. 2023. Pronunciation ambiguities in Japanese kanji. In *Proceedings of the Workshop on Computation and Written Language (CAWL 2023)*, pages 50–60, Toronto, Canada. Association for Computational Linguistics.

Wen Zhang. 2026a. Mind your moras: Orthography-aware error analysis of neural Japanese morphological generation. *arXiv preprint arXiv:2605.20043*.

Wen Zhang. 2026b. When irregularity helps: A subclass analysis of inductive bias in neural morphology. *arXiv preprint arXiv:2605.20558*.

A Type 4-2 Verb Inventory

The following table lists all 37 Type 4-2 verbs in our dataset with their lemma and past-tense forms in *hiragana*.

Verbs 1–19		Verbs 20–37	
Lemma	**Past Tense**	**Lemma**	**Past Tense**
はねかえる	はねかえった	おもいふける	おもいふけった
にげかえる	にげかえった	ねがえる	ねがえった
ひっくりかえる	ひっくりかえった	ねそべる	ねそべった
あきれかえる	あきれかえった	あざける	あざけった
ける	けった	むせかえる	むせかえった
よみふける	よみふけった	しゃべる	しゃべった
みかえる	みかえった	そりかえる	そりかえった
うらがえる	うらがえった	あまがける	あまがけった
よみがえる	よみがえった	さえかえる	さえかえった
しげる	しげった	はべる	はべった
ひるがえる	ひるがえった	かげる	かげった
おさまりかえる	おさまりかえった	わかがえる	わかがえった
せる	せった	すりへる	すりへった
たちかえる	たちかえった	おいしげる	おいしげった
いきかえる	いきかえった	にえかえる	にえかえった
あせる	あせった	ほてる	ほてった
わきかえる	わきかえった	まがりくねる	まがりくねった
せせる	せせった	つねる	つねった
ひねる	ひねった		

Table A1: All 37 Type 4-2 verbs in our dataset, with their lemma and past-tense forms in *hiragana*.

TRLM: Shared-Weight Recursive Reasoning for Non-Autoregressive Language Generation

Doğa Oytaç
İzmir Katip Çelebi University
dogaoytac@gmail.com

Abstract

Large language models achieve strong reasoning but require substantial parameters and memory, limiting deployment in resource-constrained settings. We introduce the Tiny Recursive Language Model (TRLM), a small masked non-autoregressive model that increases effective computation depth through shared-weight recursion. TRLM represents a masked sequence with input, answer, and latent reasoning states. A shared Transformer repeatedly refines the latent reasoning state, conditioning each update on trajectory time and step size via adaptive layer normalization. Training combines long- and short-trajectory answer-prediction losses with representation-level consistency. At inference, TRLM recursively refines latent states and fills all masked answer tokens in a single parallel decoding pass. We pretrain TRLM on approximately one billion OpenWebText and FineWeb-Edu tokens, then fine-tune and evaluate it on GSM8K and SVAMP against scratch-pretrained GPT-2-style, Gemma-4-Text-style, Diffusion-Gemma-style, and Mamba-2-style baselines. TRLM matches the best observed GSM8K result with 2.88% exact-match accuracy and achieves the highest SVAMP point estimate in our controlled comparison at 9.00%. Performance is non-monotonic with recursion depth: the highest observed GSM8K accuracy occurs at eight steps, whereas SVAMP peaks at four, suggesting that the preferred recursion depth may depend on the task in this setting.

1 Introduction

Scaling dominates the current paradigm in Natural Language Processing. State-of-the-art Large Language Models (LLMs), such as DeepSeek R1 (Guo et al., 2025) and Gemini 3 (Team et al., 2025), achieve their capabilities primarily through massive parameter counts and computationally expensive inference strategies like Test-Time Compute (TTC) (Snell et al., 2024). To solve complex reasoning tasks, these models rely heavily on Chain-of-Thought (CoT) prompting (Wei et al., 2023), attempting to mimic human deduction by generating intermediate steps. However, this approach faces a fundamental bottleneck. Auto-regressive generation is inherently brittle; a single incorrect token early in the sequence can cascade, rendering the entire reasoning chain and the final answer invalid (You et al., 2025). Furthermore, the efficacy of CoT is contingent upon high-quality reasoning data, which is often scarce or unavailable, making the process both data-hungry and computationally expensive (Shum et al., 2024).

These limitations are particularly relevant in small-scale language modeling, where increasing model size is not always a desirable or practical route to improved reasoning. This motivates architectures that can make more effective use of a limited parameter budget. In this work, we study a complementary form of efficiency: rather than increasing the number of learned parameters, we ask whether a compact model can increase its effective reasoning depth by reusing the same computation module recursively. This makes it possible to trade additional inference-time computation for reasoning depth while keeping the parameter count fixed.

A promising alternative to sheer scale is *recursive reasoning*, where a smaller model iteratively refines its internal state to solve a problem. This direction was recently pioneered by the Hierarchical Reasoning Model (HRM) (Wang et al., 2025), demonstrating that a relatively small, 27-million-parameter model could outperform substantially larger language models on several puzzle-based reasoning benchmarks. HRM consists of two coupled recurrent modules operating at different update rates: a *low-level* module (f_L), which updates frequently to perform rapid, detailed computations, and a *high-level* module (f_H), which updates less frequently to support slower, more abstract reason-

Proceedings of the Second BabyLM Workshop, pages 358–368
October 29, 2026 ©2026 Association for Computational Linguistics

ing. Its training method is theoretically motivated by the Implicit Function Theorem (IFT) (Krantz and Parks, 2002) and deep equilibrium models (Bai et al., 2019). HRM uses a one-step gradient approximation that backpropagates only through the most recent update of each module while treating earlier recurrent states as constants.

The Tiny Recursive Model (TRM) (Jolicoeur-Martineau, 2025) is a streamlined architecture built on the philosophy that "Less is More." Unlike its predecessor, TRM uses a single ultra-compact network consisting of only two layers. Instead of complex hierarchies, it employs a straightforward recursive process in which the model repeatedly updates a latent reasoning trace (z) and a predicted answer (y). TRM uses Deep Supervision, in which prediction losses are applied repeatedly across recursive updates rather than only to the final output. With $N_{\text{sup}} = 16$ supervision steps, this allows the small architecture to emulate the effective computational depth of a much deeper model.

More recently, Tiny Autoregressive Recursive Models (Rauba et al., 2026) adapted TRM-style refinement to causal next-token prediction under controlled, compute-matched settings. Although simpler two-level refinement baselines performed competitively, the full autoregressive TRM architecture did not provide reliable additional gains. In contrast, our work investigates recursive refinement in a non-autoregressive masked-prediction setting, where a latent reasoning state is refined before answer tokens are predicted in parallel.

Instead of learned halting, TRLM uses explicitly specified, trajectory-conditioned recursion schedules.

Our objective is to adapt this architecture, originally validated on puzzles such as ARC-AGI (Chollet, 2019), to generative, text-based reasoning tasks. We discard the fixed-point assumptions and biological constraints of earlier formulations and adopt a simpler interpretation: y represents the answer state used for prediction, while z serves as a persistent latent reasoning state. Deliberately keeping the model small substantially reduces its parameter count and per-recursion computational cost. Rather than providing constant memory efficiency, the architecture offers a flexible compute–memory trade-off: resource usage increases with the number of recursive steps, but trajectory conditioning enables the model to achieve performance close to that of much deeper recursive trajectories even when only a small number of steps is used. Thus, TRLM increases computational depth through recursion rather than additional parameterized layers. This makes it well suited to small-model settings because the parameter budget remains fixed while inference-time computation scales with recursive steps. Our main contributions are summarized as follows:

- **Adaptation of recursive architectures for non-autoregressive language modeling:** We introduce TRLM, which adapts the Tiny Recursive Model idea from static puzzle-solving to text reasoning.
- **Trajectory-conditioned recursive refinement for masked language modeling:** Building on trajectory-conditioned looped computation (Jeddi et al., 2026), we adapt normalized trajectory-time and step-size conditioning to a masked non-autoregressive architecture with separate answer and latent reasoning states. These signals are injected into the shared Transformer blocks through adaptive normalization, allowing the same reasoning module to behave differently across refinement stages.
- **Shared-weight computation for parameter-efficient reasoning:** TRLM increases effective computation depth by repeatedly applying the same Transformer reasoning module rather than adding independently parameterized layers. This allows a small model to trade additional inference-time computation for effective depth without increasing its learned parameter count.
- **Controlled downstream architecture comparison:** We fine-tune TRLM on GSM8K and SVAMP and compare it with scratch-pretrained GPT-2-style (Radford et al., 2019), Gemma-4-Text-style (Team et al., 2026), Diffusion-Gemma-style (Google DeepMind, 2026), and Mamba-2-style (Dao and Gu, 2024) baselines trained with the same tokenizer, data mixture, token budget, and evaluated with the same numeric exact-match metric.
- **Analysis of inference-time recursion depth:** We evaluate multiple recursion depths under a fixed single-pass parallel decoding procedure and observe non-monotonic accuracy across depths, with different depths attaining

the highest point estimates on GSM8K and SVAMP.

2 Related Work

2.1 Parameter Sharing and Recurrent Depth

A central line of work related to TRLM obtains effective depth by repeatedly applying a shared computation module rather than stacking independently parameterized layers. The Universal Transformer recurrently applies one transition function across depth, refining token states over multiple steps and thereby decoupling computation depth from parameter count (Dehghani et al., 2019). This connects layer sharing to recurrent computation and backpropagation through time (Werbos, 1974; Lillicrap and Santoro, 2019), while inheriting the optimization difficulties of long recurrent trajectories (Pascanu et al., 2013). Adaptive halting allows a model to learn when to stop iterative computation, while conditional computation adjusts the amount or path of computation based on the input. Both approaches therefore treat depth as an input- or position-dependent inference-time resource. However, discrete or stochastic routing can complicate training (Graves, 2017; Dehghani et al., 2019; Bengio et al., 2013).

Recent recursive and looped language models extend this idea to test-time reasoning. Relaxed Recursive Transformers combine shared-depth computation with layer-specific low-rank adaptations, balancing exact weight tying with per-layer specialization (Bae et al., 2025). LoopFormer introduces elastic-depth looped Transformers trained over variable-length trajectories, conditioning each loop on normalized trajectory time and step size and using shortcut-consistency training to align representations produced by trajectories of different lengths (Jeddi et al., 2026). Other recurrent-depth approaches repeatedly apply shared Transformer modules to increase internal computation without enlarging the base model (Geiping et al., 2025; Saunshi et al., 2025). In contrast to autoregressive test-time scaling, which typically lengthens visible reasoning traces, recurrent depth increases the transformations applied before prediction (Snell et al., 2024). TRLM follows this shared-weight tradition but adapts trajectory-conditioned recursive computation to masked language modeling through separate answer and latent reasoning states.

2.2 Latent Reasoning

Latent reasoning methods move part of reasoning from explicit text tokens into continuous hidden representations. This addresses a limitation of chain-of-thought generation: intermediate reasoning must be serialized into natural language and is exposed to autoregressive error accumulation (Wei et al., 2023; You et al., 2025). Coconut reuses a final hidden state as a continuous thought representation, feeding it back into the model without first decoding a discrete token (Hao et al., 2025). Related work treats reasoning as iterative computation over hidden states, either by increasing recurrent depth at test time or by looping Transformer computation in latent space (Geiping et al., 2025; Saunshi et al., 2025; Zhu et al., 2025; Tan et al., 2026). TRLM shares this shift from visible rationales to internal refinement, but organizes the latent process differently: instead of inserting continuous thoughts into an autoregressive chain, it maintains separate answer and reasoning states, denoted by y and z, and predicts masked answer tokens in parallel after recursive refinement.

2.3 Masked and Non-Autoregressive Language Generation

Autoregressive language models factorize generation from left to right, so each new token becomes conditioning context for later decisions (Radford et al., 2019). This is natural for open-ended continuation, but it makes decoding sequential and exposes later predictions to prefix-level errors. Masked and non-autoregressive generation instead treats the target span as an unknown canvas: observed context remains fixed, and hidden positions are predicted from a bidirectional representation of the available sequence, allowing multiple target tokens to be scored in parallel.

Masked and diffusion-based language models show that text generation need not follow strict left-to-right autoregression. Discrete diffusion language models, such as DiffusionGemma, generate text through a multi-step denoising process over discrete token representations (Google DeepMind, 2026). TRLM, however, is not a diffusion language model: it does not define a diffusion/noising process or a multi-step denoising trajectory over the output tokens. Instead, TRLM uses a masked non-autoregressive interface while performing recursive computation in a continuous latent reasoning state. After the latent state has been refined for a fixed

number of recursive steps, all masked answer positions are predicted once in parallel.

3 Method

3.1 Tiny Recursive Language Model

Figure 1 provides an overview of the TRLM architecture and its recursive latent-refinement procedure. The Tiny Recursive Language Model (TRLM) is a masked non-autoregressive model that increases effective depth by recursively reusing a shared Transformer reasoning module. Increasing recursion depth adds computation but does not add parameters. For a masked input sequence

$$\mathbf{s} = (s_1, \ldots, s_L),$$

the token embedding layer produces a fixed input representation

$$\mathbf{x} = E(\mathbf{s}) \in \mathbb{R}^{B\times L\times d}.$$

TRLM also maintains an answer state $\mathbf{y}^{(0)}$ and a persistent latent reasoning state $\mathbf{z}^{(k)}$, initialized as

$$\mathbf{y}^{(0)} = \mathbf{0}, \qquad \mathbf{z}^{(0)} = \mathbf{0}.$$

During recursive reasoning, $\mathbf{x}$ remains fixed while $\mathbf{z}$ is refined; the answer state is updated only after the final reasoning state has been obtained.

At recursive step k, the model concatenates the input, answer, and reasoning states and projects them back to the model dimension:

$$\mathbf{h}_0^{(k)} = W_c \left[\mathbf{x}; \mathbf{y}^{(0)}; \mathbf{z}^{(k)}\right] + b_c.$$

This hidden sequence is processed by the shared Transformer reasoning module,

$$\mathbf{h}_\ell^{(k)} = \mathcal{T}_{\theta,\ell}\left(\mathbf{h}_{\ell-1}^{(k)}; \mathbf{c}^{(k)}\right), \qquad \ell = 1, \ldots, N.$$

The same layer stack $\{\mathcal{T}_{\theta,\ell}\}_{\ell=1}^{N}$ is reused for all recursion steps, and the reasoning state is updated residually:

$$\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} + \mathrm{Dropout}\left(\mathbf{h}_N^{(k)}\right).$$

We use externally specified recursion trajectories rather than a learned halting variable. Consequently, the number and sizes of recursive updates are known during each forward pass. This avoids the implementation overhead of a separate q_{halting} prediction head and the additional optimization burden of learning when to stop recurrent computation. The compute budget is instead controlled directly by the chosen trajectory: the number of step sizes determines recursion depth, and their values determine how the normalized refinement interval is traversed. Trajectory time t_k and step size Δt_k also tell the shared module which refinement stage it is executing, allowing the same weights to specialize their behavior across early and late updates.

Following the trajectory-conditioning formulation used in LoopFormer (Jeddi et al., 2026), each recursive step is conditioned on its normalized trajectory time t_k and step size Δt_k:

$$\mathbf{e}^{(k)} = \mathrm{MLP}_t(t_k) + \mathrm{MLP}_{\Delta t}(\Delta t_k).$$

A conditioning projection maps $\mathbf{e}^{(k)}$ to adaptive normalization parameters for the attention and feed-forward branches of the shared Transformer. In this way, the same pre-normalized Transformer stack, implemented with rotary positional embeddings, grouped-query attention, and SwiGLU feed-forward layers, can behave differently at early and late refinement stages without introducing step-specific weights. Unlike LoopFormer, which applies trajectory conditioning to the hidden states of a decoder-only autoregressive looped language model, TRLM applies it to recursive latent reasoning before a single parallel masked-token prediction step.

After R recursive reasoning steps, the final reasoning state $\mathbf{z}^{(R)}$ is used to update the answer state:

$$\mathbf{a} = W_a \left[\mathbf{y}^{(0)}; \mathbf{z}^{(R)}\right] + b_a.$$

This representation is passed once more through the shared Transformer layer stack and added residually to the initial answer state:

$$\mathbf{y}^* = \mathbf{y}^{(0)} + \mathcal{T}_\theta\left(\mathbf{a}\right).$$

The model then applies final normalization and a vocabulary projection,

$$\mathbf{o} = \mathrm{LayerNorm}\left(\mathbf{y}^*\right), \qquad \mathbf{l} = \mathbf{o}E^\top.$$

The input embedding matrix E is tied to the language-model output projection. Although logits are produced for all sequence positions, supervision is applied only to the masked target answer positions:

$$\mathcal{L}_{\text{LM}} = -\sum_{i\in\mathcal{M}} \log p_\theta\left(s_i^* \mid \mathbf{s}_{\setminus\mathcal{M}}\right),$$

where $\mathcal{M}$ denotes the masked and supervised answer positions.

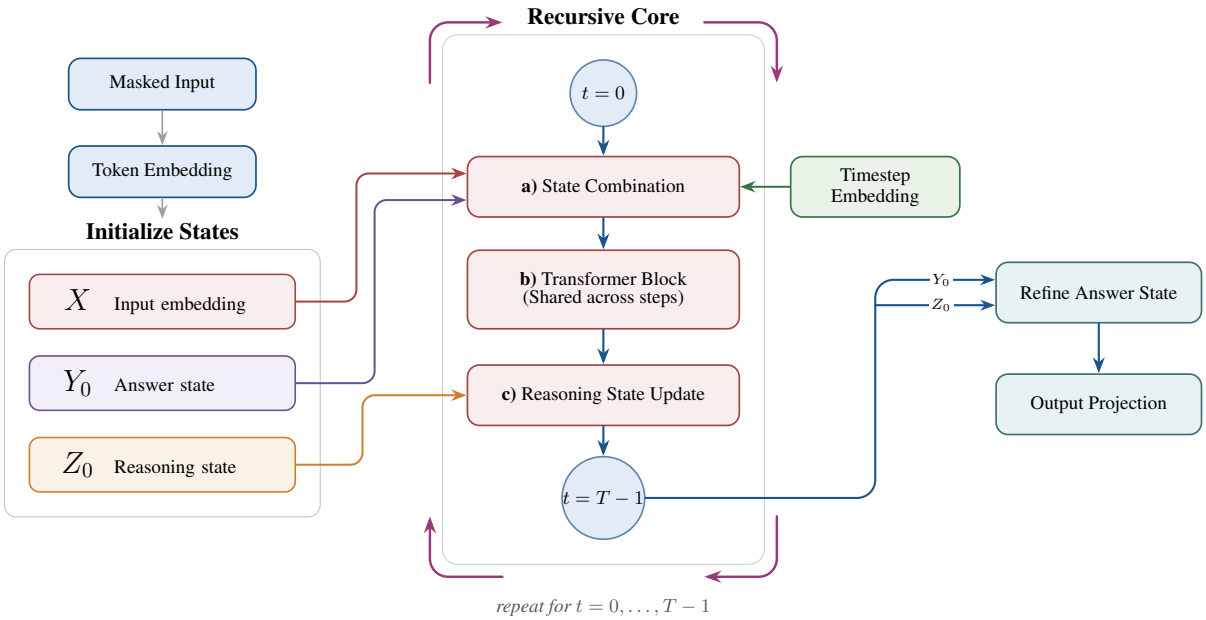


Figure 1: Overview of TRLM. The model receives a masked prompt–answer sequence, recursively refines a latent reasoning state with a shared Transformer module, and predicts the masked answer tokens in parallel.

3.2 Long–Short Trajectory Training

Our long–short trajectory training is based on the shortcut-consistency principle introduced in LoopFormer (Jeddi et al., 2026), which jointly trains long and shorter computation trajectories and encourages the shorter route to approximate a stop-gradient target produced by the longer route. We adapt this idea to masked non-autoregressive reasoning: rather than operating directly on a decoder-only language-model hidden state, TRLM applies the objective to its answer representations produced after recursive latent refinement.

Because TRLM does not learn a halting policy, it is trained with explicitly sampled long and short recursion schedules for the same masked input. A trajectory is a sequence of positive step sizes

$$\tau = (\Delta t_1, \ldots, \Delta t_K), \qquad \sum_{k=1}^{K} \Delta t_k = 1,$$

where t_k and Δt_k condition the recursive update. Applying TRLM with τ produces answer representation $\mathbf{r}_\tau$ and logits $\mathbf{l}_\tau$; the length of τ sets recursion depth while parameters remain shared.

During training, the long trajectory samples a teacher recursion length

$$K \sim \text{Uniform}\left\{K'_{\min}, \ldots, K_{\max}\right\},$$

$$K'_{\min} = \max\left(2, K_{\min}\right).$$

and uses uniform increments

$$\tau_{\text{long}} = \left(\frac{1}{K}, \ldots, \frac{1}{K}\right).$$

The short trajectory is a coarser path over the same interval: J is sampled uniformly from $\{1, \ldots, K-1\}$, and step sizes are drawn from $\{m/K\}_{m=1}^{K}$ subject to summing to one.

For trajectory τ, the masked language-modeling objective over supervised answer positions $\mathcal{M}$ is

$$\mathcal{L}_{\text{LM}}(\tau) = -\sum_{i \in \mathcal{M}} \log p_\theta\left(s_i^* \mid \mathbf{s}_{\setminus \mathcal{M}}, \tau\right).$$

The short-path language-modeling loss is

$$\mathcal{L}_{\text{short}} = \mathcal{L}_{\text{LM}}(\tau_{\text{short}}).$$

Let $\mathbf{r}_{\text{long}}$ and $\mathbf{r}_{\text{short}}$ be the final normalized answer representations produced by τ_{long} and τ_{short}. The representation-consistency term is

$$\mathcal{L}_{\text{cons}} = \left\|\mathbf{r}_{\text{short}} - \text{stopgrad}\left(\mathbf{r}_{\text{long}}\right)\right\|_2^2.$$

The resulting training objective is

$$\mathcal{L} = \mathcal{L}_{\text{LM}}(\tau_{\text{long}}) + \lambda_{\text{short}}\mathcal{L}_{\text{short}} + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}},$$

where λ_{short} and λ_{cons} weight the short-path language-modeling and representation-consistency terms. The long and short trajectories receive the same masked-token supervision. The consistency term aligns the short-path representation with a stop-gradient copy of the long-path representation, encouraging lower-compute trajectories to approximate the higher-compute solution.

3.3 Recursive Masked Generation

The prompt remains visible and the answer span is replaced by mask tokens. Let $\mathcal{M}$ be the masked answer positions. For recursion depth R, TRLM uses the uniform trajectory

$$\tau_R = \left(\frac{1}{R}, \ldots, \frac{1}{R}\right),$$

which spans the normalized interval $[0, 1]$. The model performs one forward pass with this trajectory and produces logits for all positions:

$$\mathbf{l} = f_\theta(\mathbf{s}, \tau_R).$$

Decoding is performed independently at the masked positions after invalid ids are filtered from the candidate set. The mask token is always forbidden, and other special tokens can also be suppressed when invalid. Each masked position receives the maximum-probability valid token:

$$\hat{s}_i = \arg \max_{v \in \mathcal{V}_{\text{valid}}} \text{softmax}(\mathbf{l}_i)_v, \qquad i \in \mathcal{M}.$$

Unmasked positions are copied from the input:

$$\hat{s}_i = s_i, \qquad i \notin \mathcal{M}.$$

Generation therefore consists of R latent refinement updates followed by one parallel token-selection step; R changes inference-time computation without changing parameter count or output length.

4 Experiments

4.1 Tasks and Evaluation Protocol

We evaluate all models on two mathematical reasoning benchmarks, GSM8K (Cobbe et al., 2021) and SVAMP (Patel et al., 2021). GSM8K consists of grade-school mathematical word problems that typically require multi-step arithmetic reasoning, while SVAMP evaluates arithmetic word-problem reasoning under controlled variations designed to reduce reliance on shallow linguistic or structural cues. We use evaluation sets containing 1,319 GSM8K examples and 300 SVAMP examples. Representative examples from both evaluation suites are provided in Appendix A.

Each example is formatted as a natural-language prompt followed by an answer target ending with a numeric final answer. For TRLM, prompt tokens remain visible while the answer span is replaced by mask tokens and predicted non-autoregressively. We evaluate recursion depths $r \in \{1, 2, 4, 8, 16\}$ using the uniform trajectories defined in Section 3.3. The GPT-2-style, Gemma-4-Text-style, and Mamba-2-style baselines use greedy autoregressive decoding without sampling, whereas the Diffusion-Gemma-style baseline uses its native iterative denoising procedure with a fixed evaluation seed. The same prompt formatting, numeric answer extraction, and numeric exact-match metric are used across all models. All experiments and latency measurements were conducted on a 95.6 GB G4 GPU environment.

4.2 Compared Models

We compare TRLM against four scratch-pretrained baselines representing autoregressive Transformer, Gemma-style attention, recurrent state-space, and diffusion-style generation paradigms. These are compact architecture-family baselines rather than official released checkpoints. Appendix B summarizes the controlled configurations; TRLM differs by reusing a shared Transformer reasoning module recursively and predicting masked answer tokens in parallel.

4.3 Main Results

Table 1 summarizes the fixed-length generation results on GSM8K and SVAMP. Among the non-TRLM baselines, the Gemma-4-Text-style model achieves the strongest GSM8K accuracy, reaching 2.88%, while the GPT-2-style model obtains the highest baseline result on SVAMP with 7.33%. The Diffusion-Gemma-style and Mamba-2-style baselines perform lower in this small scratch-pretrained setting, despite representing alternative non-autoregressive denoising and recurrent state-space generation paradigms.

TRLM is competitive with the strongest evaluated baseline across the two benchmarks. On GSM8K, TRLM with $r = 8$ reaches 2.88%, matching the highest baseline point estimate obtained by the Gemma-4-Text-style model, while $r = 16$ remains close at 2.81%. On SVAMP, all evaluated TRLM depths yield higher point estimates than the strongest non-TRLM baseline; the highest observed accuracy is 9.00% at $r = 4$, compared with 7.33% for the GPT-2-style baseline.

These differences should be interpreted cautiously given the relatively small number of correct predictions, particularly on SVAMP. Nevertheless, the results show that recursive masked generation remains competitive with similarly sized scratch-

Model	Params	GSM8K Acc.	SVAMP Acc.	Avg. Gen. Time
Diffusion-Gemma-style (scratch)	74.3M	0.99% (13/1319)	3.00% (9/300)	728.31 ms
Gemma-4-Text-style (scratch)	77.6M	**2.88% (38/1319)**	6.33% (19/300)	53.47 ms
GPT-2-style (scratch)	73.5M	2.58% (34/1319)	7.33% (22/300)	21.15 ms
Mamba-2-style (scratch)	76.0M	1.29% (17/1319)	1.33% (4/300)	75.19 ms
TRLM $r = 1$	74.5M	2.73% (36/1319)	8.33% (25/300)	**11.06 ms**
TRLM $r = 2$	74.5M	2.65% (35/1319)	8.67% (26/300)	16.25 ms
TRLM $r = 4$	74.5M	2.65% (35/1319)	**9.00% (27/300)**	25.86 ms
TRLM $r = 8$	74.5M	**2.88% (38/1319)**	8.00% (24/300)	46.21 ms
TRLM $r = 16$	74.5M	2.81% (37/1319)	8.67% (26/300)	85.72 ms

Table 1: Main generation results on GSM8K and SVAMP. Accuracy is numeric exact match. Average generation time is the per-example mean computed over both test sets from total generation time. Best accuracy values and the lowest average generation time are shown in bold.

pretrained architectures while providing a direct inference-time control over additional latent computation.

Accuracy varies non-monotonically with recursion depth in our experiments. The highest GSM8K point estimate occurs at $r = 8$, whereas the highest SVAMP point estimate occurs at $r = 4$. Thus, within the evaluated range, increasing recursion depth does not consistently increase accuracy. In contrast, latency increases monotonically with depth, making r a direct test-time compute control.

4.4 Analysis of Trajectory Conditioning

Table 2 compares TRLM with and without trajectory-time and step-size conditioning across the evaluated recursion depths. We remove trajectory-time and step-size conditioning while keeping the remaining architecture and evaluation protocol fixed. The conditioned variant yields higher accuracy point estimates at four of the five evaluated depths on GSM8K and matches the unconditioned model at $r = 4$. On SVAMP, conditioning yields higher point estimates at four depths, while the unconditioned model is higher at $r = 8$. These differences are generally small and should therefore be interpreted as descriptive rather than as evidence of statistically significant improvements.

Conditioning consistently increases latency because of the additional conditioning projections and adaptive-normalization computations. Overall, the ablation suggests that trajectory conditioning can be beneficial across several recursion depths, although the magnitude and consistency of the accuracy effect require further evaluation.

5 Discussion

The results suggest that the main benefit of TRLM is not a monotonic gain from increasing recursion depth, but the ability to expose a controllable compute–depth trade-off while keeping the learned parameter count fixed. On GSM8K, TRLM matches the highest baseline point estimate at $r = 8$, whereas on SVAMP all evaluated recursion depths yield higher point estimates than the strongest non-TRLM baseline. Notably, $r = 1$ already reaches 8.33% accuracy on SVAMP, compared with 7.33% for the strongest non-TRLM baseline. This indicates that the observed SVAMP advantage cannot be attributed solely to increasing recursive depth; characteristics of the shared-weight masked architecture and its training objective may also contribute. Because the present evaluation covers only two reasoning benchmarks, this interpretation should remain tentative.

The relationship between recursion depth and accuracy is clearly non-monotonic in our experiments. GSM8K reaches its highest point estimate at $r = 8$, while SVAMP peaks at $r = 4$, and deeper trajectories do not consistently improve either benchmark. One possible explanation is that additional recursive updates can eventually become redundant or perturb useful intermediate representations rather than further refining them. Another possibility is that different problems require different amounts of latent computation. The current experiments do not isolate these mechanisms, but the observed pattern suggests that a fixed recursion depth is unlikely to be optimal for every input.

The trajectory-conditioning ablation provides additional evidence that the behavior of the shared module depends on its position in the refinement

	With trajectory conditioning			Without trajectory conditioning		
r	**GSM8K Acc.**	**SVAMP Acc.**	**Avg. Time**	**GSM8K Acc.**	**SVAMP Acc.**	**Avg. Time**
1	2.73% (36/1319)	8.33% (25/300)	11.06 ms	2.20% (29/1319)	8.00% (24/300)	9.74 ms
2	2.65% (35/1319)	8.67% (26/300)	16.25 ms	2.58% (34/1319)	8.00% (24/300)	14.13 ms
4	2.65% (35/1319)	**9.00% (27/300)**	25.86 ms	2.65% (35/1319)	8.00% (24/300)	22.06 ms
8	**2.88% (38/1319)**	8.00% (24/300)	46.21 ms	2.35% (31/1319)	8.33% (25/300)	38.66 ms
16	2.81% (37/1319)	8.67% (26/300)	85.72 ms	2.50% (33/1319)	8.00% (24/300)	71.38 ms

Table 2: Trajectory-conditioning ablation for TRLM. The conditioned results are the TRLM rows from Table 1. The unconditioned model is trained and fine-tuned without trajectory-time and step-size conditioning, while keeping the same 74.5M parameter scale and evaluation protocol.

trajectory. Conditioning yields higher accuracy point estimates at four of five evaluated depths on both benchmarks, although the differences are generally small and not uniform. This suggests that providing trajectory time and step size can help a shared module distinguish early from late refinement stages, while also introducing a modest computational overhead.

Shared-weight recursion keeps the number of learned parameters fixed as recursion depth increases, but additional computation is not free. In our measurements, latency increases from 11.06 ms at $r = 1$ to 85.72 ms at $r = 16$. It therefore remains unclear whether the same accuracy–compute trade-off will persist at larger model and data scales. The non-monotonic depth results also motivate adaptive computation as a natural extension of TRLM. Rather than selecting R externally, future variants could use a learned halting mechanism or a convergence criterion based on changes in the latent reasoning state or output distribution, allowing easier examples to terminate earlier while allocating additional computation to harder ones.

6 Conclusion

We introduced TRLM, a masked non-autoregressive language model that applies a shared Transformer reasoning module recursively before predicting answer tokens in parallel. This design separates parameter count from effective computation depth: changing the recursion depth changes inference-time compute, but not the number of learned parameters.

Across GSM8K and SVAMP, TRLM is competitive with similarly sized scratch-pretrained autoregressive, state-space, and diffusion-style baselines. It matches the highest GSM8K point estimate and obtains the highest SVAMP point estimate in our comparison, while the fastest TRLM setting also gives the lowest observed average generation latency.

Accuracy varies non-monotonically with recursion depth: the highest observed GSM8K accuracy occurs at $r = 8$, whereas SVAMP reaches its highest point estimate at $r = 4$. These results suggest that additional recursive computation is not uniformly beneficial and motivate future work on adaptive recursion policies, answer-canvas selection, and larger-scale evaluation.

Limitations

This study was conducted under limited computational resources, which constrained both the scale of the evaluated models and the extent of their training. To keep pretraining, fine-tuning, and repeated evaluation computationally feasible, all models were maintained at a relatively small parameter scale and were trained with a limited token budget. Consequently, the reported results may not reflect the behavior of TRLM under substantially larger model sizes, longer training schedules, or more diverse pretraining corpora. It remains unclear whether the observed advantages of recursive parameter sharing will persist, diminish, or become stronger as model and data scale increase.

The evaluation is also restricted to GSM8K and SVAMP. These datasets were selected because they are widely used and well-established mathematical reasoning benchmarks, allowing TRLM to be compared with common baseline architectures under a consistent and reproducible evaluation protocol. However, both benchmarks consist primarily of relatively short mathematical word problems and therefore represent only a limited subset of language reasoning. The results should not be interpreted as evidence of general superiority across broader capabilities such as commonsense reasoning, long-context understanding, knowledge-

intensive question answering, or open-ended text generation. Future work should therefore evaluate larger models with more extensive training and include a broader range of benchmarks.

TRLM uses externally selected recursion depths rather than learning an input-dependent halting policy. Although this simplifies training and provides predictable compute, it may allocate too few or too many recursive updates to individual examples. A direct comparison with learned q_{halting}-based adaptive computation remains future work.

Acknowledgments

This work received support from the Secretariat of Defence Industries of the Republic of Türkiye (SSB) through the EVREN Platform.

References

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. 2025. Relaxed recursive transformers: Effective parameter sharing with layer-wise lora. *Preprint*, arXiv:2410.20672.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2019. Deep equilibrium models. *Preprint*, arXiv:1909.01377.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *Preprint*, arXiv:1308.3432.

François Chollet. 2019. On the measure of intelligence. *Preprint*, arXiv:1911.01547.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Tri Dao and Albert Gu. 2024. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *Preprint*, arXiv:2405.21060.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. *Preprint*, arXiv:1807.03819.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *Preprint*, arXiv:2502.05171.

Google DeepMind. 2026. Diffusiongemma: Open-weights discrete diffusion mixture-of-experts language model. `https://huggingface.co/google/diffusiongemma-26B-A4B-it`. Accessed: 2026-07-17.

Alex Graves. 2017. Adaptive computation time for recurrent neural networks. *Preprint*, arXiv:1603.08983.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2025. Training large language models to reason in a continuous latent space. *Preprint*, arXiv:2412.06769.

Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. 2026. Loopformer: Elastic-depth looped transformers for latent reasoning via shortcut modulation. *Preprint*, arXiv:2602.11451.

Alexia Jolicoeur-Martineau. 2025. Less is more: Recursive reasoning with tiny networks. *Preprint*, arXiv:2510.04871.

Steven George Krantz and Harold R Parks. 2002. *The implicit function theorem: history, theory, and applications*, volume 202. Springer.

Timothy P Lillicrap and Adam Santoro. 2019. Backpropagation through time and the brain. *Current opinion in neurobiology*, 55:82–89.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. *Preprint*, arXiv:1211.5063.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? *Preprint*, arXiv:2103.07191.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Paulius Rauba, Claudio Fanconi, and Mihaela van der Schaar. 2026. Tiny autoregressive recursive models. *Preprint*, arXiv:2603.08082.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. *Preprint*, arXiv:2502.17416.

KaShun Shum, Shizhe Diao, and Tong Zhang. 2024. Automatic prompt augmentation and selection with chain-of-thought from labeled data. *Preprint*, arXiv:2302.12822.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. 2026. Think silently, think fast: Dynamic latent compression of llm reasoning chains. *Preprint*, arXiv:2505.16552.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, and 282 others. 2026. Gemma 4 technical report. *Preprint*, arXiv:2607.02770.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. 2025. Hierarchical reasoning model. *Preprint*, arXiv:2506.21734.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Paul Werbos. 1974. Beyond regression: New tools for prediction and analysis in the behavioral sciences. *PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA*.

Weiqiu You, Anton Xue, Shreya Havaldar, Delip Rao, Helen Jin, Chris Callison-Burch, and Eric Wong. 2025. Probabilistic soundness guarantees in llm reasoning chains. *Preprint*, arXiv:2507.12948.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, and 14 others. 2025. Scaling latent reasoning via looped language models. *Preprint*, arXiv:2510.25741.

A Benchmark Examples

Dataset	Example problem	Gold answer
GSM8K	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?	18
SVAMP	There were 106 dollars in Olivia's wallet. After she visited a supermarket and a showroom there were 26 dollars left. If she spent 49 dollars at the showroom, how much did she spend at the supermarket?	31

Table 3: Representative examples from the GSM8K and SVAMP evaluation sets used in our experiments.

B Model Configurations

Model	Params	Key configuration	Generation paradigm
GPT-2-style (scratch)	73.5M	12 layers, hidden size 512, 8 attention heads, FFN size 2816, context length 1024	Autoregressive decoder-only Transformer
Gemma-4-Text-style (scratch)	77.6M	12 layers, hidden size 512, 8 query heads, 4 key/value heads, head dimension 64, FFN size 2304, sliding window 512	Autoregressive attention-based Transformer with grouped-query attention
Mamba-2-style (scratch)	76.0M	19 layers, hidden size 512, expansion factor 2, state size 128, 8 groups, 16 heads, head dimension 64	Autoregressive recurrent state-space model
Diffusion-Gemma-style (scratch)	74.3M	12 layers, hidden size 512, 8 query heads, 4 key/value heads, head dimension 64, one expert, canvas length 256	Text-only block diffusion with iterative denoising
TRLM (ours)	74.5M	12 shared Transformer layers, hidden size 512, 8 attention heads, grouped-query attention, SwiGLU size 2048, maximum recursion depth 16	Masked non-autoregressive generation with recursive latent refinement

Table 4: Controlled 75M-class model configurations. All models are trained from scratch on the same OpenWebText/FineWeb-Edu mixture and use the same tokenizer and pretraining token budget. Names ending in "style" denote architecture-family baselines, not official pretrained checkpoints.

Right Tool, Right Job: Native-Language Evaluation, Tokenizer Sensitivity, and Methodological Findings from a French-Only BabyLM

Adam Zachary Wasserman[1] **David Beauchemin**[2]
[1]Open Honest Foundation
[2]Group for Research in Artificial Intelligence of Laval University (GRAIL),
Université Laval, Québec, Canada
adam.wasserman@openhonest.org david.beauchemin@ift.ulaval.ca

Abstract

We submit MéTRON-FR, a 125M GPT-2 pretrained on 92.47M words of French, to the BabyLM 2026 Strict track. It scores $85.97 \pm 0.17\%$ on QFrBLiMP (a native Quebec-French benchmark of grammatical minimal pairs) and 62.80% on the BabyLM-weighted leaderboard. A cross-lingual GLUE (General Language Understanding Evaluation) protocol that combines French task-data translation with rank-16 LoRA (Low-Rank Adaptation) produces a sharp task-type gradient: relational tasks gain measurably, while world-knowledge tasks regress. Bilingual Lexicon Induction aligns the French embeddings to GPT-2 at p@1 = $68.84 \pm 8.61\%$, $18\times$ above chance, suggesting cross-lingual alignment tracks acquired grammatical competence rather than training duration. An ablation study shows that single-token zero-shot scoring is dominated by tokenizer and template artifacts at the child scale, motivating tokenizer-swap sensitivity, placebo-controlled prompting, and native-language minimal-pair benchmarks as standard diagnostics.

1 Introduction

The BabyLM challenge (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025) asks whether language models can be trained on plausible amounts of input. Submissions in the Strict track are constrained to 100M words, and are evaluated on grammatical, world-knowledge, and downstream-task probes. This budget makes data curation and architectural choices first-order determinants of performance and turns child-scale pretraining into a tractable laboratory for both cognitive-science-inspired hypotheses about acquisition and engineering questions about efficiency.

Submissions through 2025 have all been in English, raising the question of whether the lessons drawn at the child scale are properties of language acquisition in general or of English specifically. The 2026 edition lifts this restriction (Choshen et al., 2026), and BabyBabelLM (Jumelet et al., 2026a) provides a multilingual training corpus across 15 languages. We submit to this edition MéTRON-FR, a 125M-parameter GPT-2 pretrained for five epochs on the French-only MéTRON-FR Strict corpus, with sentence-level oversampling driven by a Haitian-Creole lexical oracle. We evaluate on both the official BabyLM 2025 pipeline and with the Quebec French benchmarks (Beauchemin et al., 2026b; Beauchemin and Khoury, 2025).

At the date of writing, MéTRON-FR's 85.97% on QFrBLiMP exceeds the best native-BLiMP (Benchmark of Linguistic Minimal Pairs; Warstadt et al., 2020) score posted in any of the three official languages of the 2026 multilingual track: 80.75% on English BLiMP (Strict, ≈100M words), 81.70% on Dutch BLiMP-NL (50M words), and 78.60% on Chinese ZhoBLiMP (50M words).[1] Each version of BLiMP is a separate monolingual native-language minimal-pair evaluation instrument.

Our single central message is that a French-only child-scale model attains strong *native* grammatical competence, and that at this scale single-token zero-shot scoring is dominated by tokenizer and answer-template artifacts; everything else in the paper (cross-lingual transfer via BLI and CL GLUE, and the forensic ablations) is evidence in service of that message and of the reporting diagnostics we recommend for non-English child-scale submissions. Our contributions are: **1)** We submit, to our knowledge, the first model trained from scratch under the BabyLM 100M-word constraint for evaluation on the Quebec dialect of French. **2)** We release a cross-lingual (CL) GLUE adaptation protocol that combines French task-data translation with rank-16 LoRA, and characterize a model's failure modes via three contrastive levers. **3)** We adapt Bilingual Lexicon Induction (BLI) as

[1]Leaderboard snapshot 2026-05-06

Proceedings of the Second BabyLM Workshop, pages 369–387
October 29, 2026 ©2026 Association for Computational Linguistics

a child-scale geometric diagnostic of CL structural alignment. The model, corpus, bilingual lemma bridge, and all evaluation scripts are available at github.com/adamzwasserman/babylm.

The rest of this paper is organized as follows. In Section 2, we present a review of small-scale and multilingual LM work amongst native non-English minimal-pair benchmarks. Section 3 describes the model, corpus, and evaluation protocol. Section 4 reports results on native Quebec French and on the official BabyLM suite. Section 5 presents an ablation study isolating tokenizer-swap sensitivity, placebo-controlled prompting, and a lexical-frequency confound in the translated BLiMP Supplement, then draws methodological recommendations for child-scale non-English submissions (Section 5.4). We conclude the article in Section 6.

2 Related Work

Non-English and multilingual child-scale LMs. Strict-track BabyLM entries from 2023 to 2025 (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025) are English-only and have advanced primarily through architectural innovation, with ELC-BERT (Charpentier and Samuel, 2023) and GPT-BERT (Charpentier and Samuel, 2024) as the leading lines of work. The 2026 multilingual axis (Choshen et al., 2026) is enabled by BabyBabelLM (Jumelet et al., 2026a), a developmentally plausible corpus across 15 languages, including French. Cross-linguistic studies with matched architectures document large differences in training efficiency across typologically diverse languages (Wasserman, 2026), with morphologically rich languages reaching grammatical competence with orders-of-magnitude fewer tokens than English.

Native non-English minimal-pair and acceptability benchmarks. A French-trained model evaluated on translated BLiMP conflates model competence with translation quality and with lexical-frequency artifacts of the translation pipeline. Native targeted-syntactic and acceptability corpora that avoid this confound have been released for several languages following the methodology of Warstadt et al. (2020); the Quebec-French Benchmark of Linguistic Minimal Pairs (QFrBLiMP; Beauchemin et al., 2026b) and the Quebec-French Corpus of Linguistic Acceptability (QFrCoLA; Beauchemin and Khoury, 2025) provide this instrument for Quebec French. Moreover, the COLE evaluation suite (Beauchemin et al., 2025) bundles these grammatical probes alongside lexical and cultural-knowledge tasks (Beauchemin et al., 2026a).

CL transfer and structural alignment. Two complementary methods inform our CL setup. BLI learns an orthogonal map between embedding spaces (Mikolov et al., 2013; Lample et al., 2018) and provides a geometric measurement of CL alignment that does not depend on translation. Yet, it was never adapted as a child-scale diagnostic to connect the resulting measurements to the formal-versus-functional linguistic competence distinction (Mahowald et al., 2024). For downstream CL evaluation, parameter-efficient fine-tuning via Low-Rank Adaptation (Hu et al., 2022) preserves the pretrained base by construction and lets us run a four-lever experiment grid on a single submitted checkpoint without losing the grammatical signal measured natively.

3 Methodology

We address three orthogonal experimental questions: native grammatical competence (Section 4.1), CL transfer along four levers (BLI in Section 4.3; GLUE adaptation in Section 4.4), and forensic ablations isolating tokenizer, prompt-template, and lexical-frequency confounds in benchmark scoring (Section 5). All experiments share a single submitted checkpoint; no checkpoint is selected per experiment.

3.1 Architecture and Training

MéTRON-FR (Greek *métron* for "measurement") is a 125M-parameter GPT-2 (Radford et al., 2019) pretrained for five epochs on the MéTRON-FR Strict corpus (Section 3.3 and Section 3.4). We select the best checkpoint as the empirically located grammatical-competence peak; intermediate checkpoints are released at the BabyLM-mandated cadence. Architecture hyperparameters, training schedule, hardware, GPU budget, and the released checkpoint manifest are detailed in Appendix E.

Reproduction protocol. The submitted checkpoint is the headline run; for variance estimation, we report a 5-seed reproduction ($[42–46]$) at matched corpus, tokenizer, and hyperparameters. Results in Section 4 report mean $\pm$ standard deviation across the 5 seeds, alongside the submitted-checkpoint headline values (epoch 3, seed 42); the per-seed epoch-selection rule is given in Appendix E.

3.2 Tokenizer

We use a 50,000-token byte-pair encoding (BPE) (Sennrich et al., 2016) tokenizer trained on the French training corpus itself (Appendix J). Three alternatives were compared in an ablation study: 16K/50K/64K child-directed speech-trained (CDS-trained). The 50K vocabulary yielded the lowest perplexity and the best downstream stability. The tokenizer is not a tuning parameter in our analysis. Indeed, Section 5.1 reports a forensic ablation in which only the tokenizer is swapped, with corpus, architecture, training steps, optimizer, and seed held constant.

3.3 Corpus Composition

The training corpus is French-only and combines three sources, totalling 92,469,402 words (92.47M) as counted on the released `train_french.txt`. A grammatically dense base draws on child-directed speech and accessible French text: ≈2.10M words from CHILDES French (MacWhinney, 2000), plus ≈24.90M words from the non-subtitle portion of the French component of BabyBabelLM (Jumelet et al., 2026a). The remainder, ≈ 66M words, comes from the filtered French Wikipedia (see Appendix E for details). Per-source figures are counts of the source pools before oracle-weighted assembly (Section 3.4) and therefore do not sum exactly to the released total; the 92.47M figure is the whitespace word count of the released corpus file. A hand-curated bilingual French–English lemma bridge is released alongside the corpus but is **not part of the training data**: its sole role is to supply the seed dictionary of the BLI diagnostic (Section 4.3). It consists of 73 lemma entries (30 verbs, 18 nouns, 15 adjectives, 10 function words) with full inflection tables and English glosses, enumerating 715 distinct (FR, EN) form-level pairs. The corpus builder also exposes a variant harness that interleaves the bridge into the training text; the submitted model was trained on the French-only harness, and no bridge text enters it. The corpus is pan-French rather than Quebec-specific: no Quebec-restricted corpus of comparable size exists at the BabyLM scale, and most QFrBLiMP phenomena (syntactic, morphological, agreement) are shared across French varieties; the gap with the Quebec normative prescription is isolated in the anglicism-related subscore and discussed in Section 4.1. This simple recipe outperforms a more aggressively engineered reallocation with EWoK-targeted Wikipedia (EWoK, Elements of World Knowledge; Ivanova et al., 2025) and instruction blocks; see the ablation in Section 5.1.

3.4 The Haitian-Creole Lexical Oracle

Sentences are not sampled uniformly from the three sources. We oversample by the density of French lemmas whose cognates survived into Haitian Creole. Our working hypothesis, informed by creole-genesis research (Lefebvre, 1998; Mufwene, 2001), is that because creoles form under acute communicative pressure, lemmas surviving this contact filter are plausibly disproportionately load-bearing for grammatical communication. Concretely, we map a high-frequency Haitian-Creole (HC) lemma list to a French cognate set and oversample sentences dense in these lemmas; the extraction procedure, the cognate set, and the oversampling weight are detailed in Appendix C. **No Haitian Creole text enters training**. The lemmas act as an oracle lexical filter on French source material only. Using HC as an instrument rather than a training source is consistent with prior evidence that mixing analytical languages into morphologically rich corpora degrades the training dynamics of the rich language (Wasserman, 2026). French is the morphologically richer member of the French–HC pair on every category that QFrBLiMP probes (verbal inflection, determiner-noun and subject-verb agreement); treating HC as a lexical filter rather than a training distribution preserves these French regularities while still drawing on the contact-filtered survival signal as a corpus-weighting heuristic. We present the oracle as an *exploratory* corpus-weighting heuristic, not as a validated contribution: its effect is not isolated against an unweighted baseline in this paper, and the clean-rebuild control (Section 5.1) bounds any oracle-specific gain to within the training-time noise floor (Section 6). We report it because the recipe is what we submitted, and flag the unweighted-baseline ablation as follow-up work.

3.5 Evaluation Suite

Official BabyLM 2026 leaderboard. We report scores on the four instruments of the released 2025 evaluation pipeline. BLiMP (Warstadt et al., 2020) is a zero-shot targeted-syntactic benchmark of English minimal pairs grouped into 67 paradigms (each a good/bad sentence pair), scored by comparing per-token log-probabilities. The BLiMP Supplement adds five pragmatic and

discourse paradigms (e.g. question-answer congruence, subject-aux inversion) in the same format. (Super)GLUE (Wang et al., 2018) is a set of seven classification tasks evaluated after fine-tuning. EWoK (Ivanova et al., 2025) probes world knowledge across physical, spatial, and social domains as a binary choice. The leaderboard reports both per-task accuracies and an aggregate score that averages across these instruments; a French-trained model is expected to be near chance on the English zero-shot probes, so we read the English-suite numbers as a lower-bound sanity check rather than a competitive target.

Native Quebec-French grammatical competence. Neither QFrBLiMP nor QFrCoLA is part of the official BabyLM suite; they are native Quebec-French instruments we adopt as our primary grammatical probe, on the argument (Section 2) that a native minimal-pair benchmark avoids the translation and lexical-frequency confounds of a translated one. They are complementary to, not a substitute for, MultiBLiMP (Jumelet et al., 2026b), whose French portion targets the metropolitan standard: QFrBLiMP encodes the Office québécois de la langue française (OQLF) Quebec norm, so the two measure different prescriptive targets rather than the same one at different difficulty. We map the QFrBLiMP buckets to the translated-BLiMP paradigm breakdown per phenomenon in Appendix F. We use two native Quebec-French corpora. First, the QFrBLiMP (Beauchemin et al., 2026b) a corpus of 1,761 minimal pairs annotated for 20 fine-grained linguistic phenomena of Quebec French, taken from the *Banque de dépannage linguistique* (BDL), a public normative grammar resource maintained by the OQLF, and annotated by twelve native Quebec-French annotators. The 20 phenomena span syntactic, morphological, lexical-semantic, and lexical-normative judgments. Second, the QFrCoLA (Beauchemin and Khoury, 2025) a sentence-level acceptability judgment corpus extracted from the BDL, with an additional out-of-domain split drawn from the *Académie française* normative journal. Each sentence is labelled acceptable or unacceptable, where 69.50% of sentences are labelled acceptable. The in-domain release totals 25,153 sentences, plus a separate 2,675-sentence out-of-domain set.

CL measurements. Because English benchmarks cannot be applied directly to a French-trained model without translation or CL adaptation, we also report a structural CL measurement via BLI against frozen English embedding matrices (Section 4.3) and a four-lever experiment grid for CL GLUE (Section 4.4).

3.6 Evaluation Metrics

The minimal-pair benchmarks (QFrBLiMP, BLiMP, BLiMP Supplement, EWoK) are scored zero-shot as item-level accuracy per paradigm and overall; QFrCoLA is fine-tuned and reported as accuracy and Matthews Correlation Coefficient (MCC) (Matthews, 1975); CL GLUE is fine-tuned with rank-16 Low-Rank Adaptation (LoRA) (Hu et al., 2022) on French-translated task data and reported as per-task accuracy and a five-task mean (Section 4.4); and BLI reports the orthogonal Procrustes fit and word-translation precision@k on a held-out split (Section 4.3). The exact scoring rules, the MCC reference points, and the fine-tuning setup are detailed in Appendix B.

4 Results

4.1 Native Quebec-French Evaluation

QFrBLiMP. The submitted checkpoint reaches $85.97 \pm 0.17\%$ overall on QFrBLiMP at epoch 3, the empirically located grammatical-competence peak (point estimate from the submitted run; variance from the 5-seed reproduction described in Section 3.1). We report accuracy aggregated into four broader buckets in Table 1; the epoch-1 reference checkpoint `chck_92M`, used downstream for adaptation comparisons, scores 83.53%. Figure 1 shows the multi-epoch trajectory across the 5-seed reproduction. Grammatical accuracy oscillates within a narrow band of approximately 0.40pp across epochs while training loss continues to decline, monotonically from epoch 3 onward, indicating, at the BabyLM scale, that grammatical competence saturates before perplexity saturates.

Anglicisms as a probe of normative Quebec French. On QFrBLiMP a higher score is better: it means the model assigns lower probability to the ungrammatical (here OQLF-proscribed) member of each minimal pair. The anglicism-related bucket is the lowest of the four (Table 1) because sensitivity to the prescriptive norm is the hardest of these signals to acquire at the child scale, not because the model is failing at grammar in general. The anglicism-related sub-score reflects two simultaneous mechanisms. The first is phonotactic and morphological internalization: anglicism

Bucket	Accuracy
Syntactic	$89.74 \pm 1.32\%$
Semantic	$87.19 \pm 1.70\%$
Morphological	$85.47 \pm 1.02\%$
Anglicism-related	$80.15 \pm 1.54\%$
Overall (submitted)	$\mathbf{85.97 \pm 0.17\%}$

Table 1: QFrBLiMP results on the submitted checkpoint, zero-shot, using the aggregated four high-level buckets used in Beauchemin et al. (2026b). Bucket point estimates are from the submitted run; standard deviations are from the 5-seed reproduction (Section 3.1). The epoch-1 reference checkpoint scores 83.53% overall.

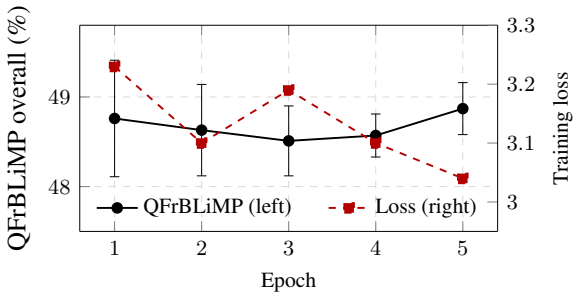


Figure 1: QFrBLiMP overall accuracy per epoch (mean $\pm$ std across 5 reproduction seeds; Section 3.1) versus training loss on the submitted run. Grammatical accuracy stays within a $\sim$0.40pp band across the 5 epochs while training loss continues to decline, monotonically from epoch 3 onward, illustrating that grammatical competence saturates before perplexity does at this scale.

candidates with non-French coda clusters or stress patterns receive lower probability under any model that has internalized French regularities, regardless of normative status. The second is sensitivity to the prescriptive norm encoded in the BDL itself: many lexical items in the underlying lexical-semantics phenomenon are phonotactically licit French formations (e.g. *matcher*) but flagged as unacceptable in the BDL because the OQLF prescribes a French-source equivalent (*correspondre*) as the standard. A high sub-score, therefore, reflects either internalization of the prescriptive distribution itself or of morphological cues correlated with it. Results on a metropolitan-French benchmark may differ on anglicism-sensitive items, where descriptive metropolitan usage is more permissive; the morphological and syntactic sub-scores are expected to generalize. Appendix K details the OQLF normative framework.

QFrCoLA. On the in-domain test set, the fine-tuned classifier achieves $69.68 \pm 0.50\%$ accuracy and an MCC of 0.13 ± 0.03 across the 5-seed reproduction. The reference upper bound is BERT-base on English CoLA at MCC 0.52, achieved with $\approx$ 110M parameters but on the order of 3.30B pretraining tokens. At the $\approx$1.3 tokens-per-word rate of English BPE, that budget is $\approx$ 2.5B words, so ours is at the same parameter scale with $\approx 27\times$ fewer words; tokens and words are not interchangeable units, and we state the comparison word-for-word. The 0.13 MCC characterizes what acceptability-judgment competence is measurable at the BabyLM word budget on a native non-English benchmark: positive and above chance, but well below that of a model trained at the standard scale.

4.2 Official BabyLM Suite

Translated BLiMP and Supplement. On translated BLiMP, the submitted checkpoint scores 76.28%. We report this strictly as a cross-lingual transfer number: our French-trained model is scored on French translations of the originally English items, so it is not directly comparable to English-native models evaluated on the original English items. For context only, and not as a like-for-like ranking, OPT-125M (a 125M-parameter open pretrained transformer language model; Zhang et al., 2022) scores 75.30% and ELC-BERT (Charpentier and Samuel, 2023) 85.30% on the English items. See Appendix F for the per-paradigm breakdown. The translated BLiMP Supplement scores 34.00%, below chance, reported as a methodological finding rather than a performance claim (Section 5.3).

GLUE. Under the submitted configuration, GLUE reaches a mean of 60.90% across the five translated tasks, on par with the 61.21% English-LoRA baseline mean (discussed in Section 4.4). We translate the official English GLUE splits into French via an LLM pass, preserving the original alignment targets; the translation script is released with the submission. We treat this as a downstream adaptation choice rather than a pretraining data violation under the 2025 rule change: no new task content is introduced, and each French example is deterministically derivable from its English source.

EWoK. EWoK falls inside the 95% chance interval $[47.90\%, 52.10\%]$ for $n = 2200$ across all four interventions we tested (Section 5.3). We report it as a clean structural null result that bounds the task definition at the child scale.

Target	Fit	p@1 (%)	p@5 (%)	p@10 (%)
GPT-2 (competent EN)	$\mathbf{0.40 \pm 0.03}$	$\mathbf{68.84 \pm 8.61}$	$\mathbf{91.87 \pm 6.07}$	$\mathbf{98.56 \pm 1.97}$
Matched-arch failed-EN	0.51	25.00	52.10	66.70
Random orthogonal (ref)	−0.20	11.50	26.90	53.80
Chance ($1/n_{\text{test}}$)	—	3.80	19.20	38.50

Table 2: BLI Procrustes alignment from the BabyLM French model to two English embedding spaces. The GPT-2 row reports mean ± std across the 5-seed reproduction (Section 3.1); the matched-arch failed-EN row is from the submitted run only. Fit $= 1 - \|E_{\text{FR}}W - E_{\text{EN}}\|_F^2 / \|E_{\text{EN}}\|_F^2$ (1.00 perfect, 0.00 no alignment); p@k is held-out word-translation precision at the k nearest neighbours under the learned map.

Weighted leaderboard score. On the BabyLM-weighted metric, the submission scores 62.80%, at or above all Strict-track reference points re-evaluated on the 2025 pipeline (GPT-BERT Charpentier and Samuel, 2024 NLP 57.8–63.0, macro 40.8–43.5; GPT-2 naive; 2025 Strict-track winners; Charpentier et al., 2025; Appendix G). We read this not as a competitive outperformance but as evidence that the language-choice axis is empirically larger than the architectural axis the field has been optimizing on.

4.3 CL Structural Alignment via BLI

The first CL question, i.e. "does a French-trained model exhibit structure recognizably aligned with that of English-trained models?", admits a direct geometric answer via BLI (Mikolov et al., 2013; Lample et al., 2018): we learn a frozen orthogonal map between the two models' embedding spaces on a seed dictionary, then measure held-out word-translation precision@k by nearest-neighbour retrieval in the mapped space. The seed dictionary is the form-level expansion of the 73-entry bilingual lemma bridge of Section 3.3, a released resource that is not part of the training corpus; its 715 enumerated pairs reduce to 242 (FR, EN) pairs (194 to fit the map, 48 held out) after a morphology-aware parser pass; the closed-form Procrustes derivation and the parser-pass details are in Appendix H. We compare the BabyLM French model against two English embedding targets (Table 2): (a) GPT-2 (Radford et al., 2019), 117M parameters trained on ≈8B WebText tokens and fully grammatically competent; (b) a matched-architecture 125M English model that plateaued at chance (40%) on English grammar probes despite being trained on 6.50B C4 tokens (Wasserman, 2026).

Alignment to GPT-2 reaches p@1 of $68.84 \pm 8.61\%$ on held-out word pairs, an order of magnitude above chance, purely via a frozen 768×768 orthogonal matrix; content-word alignments are rank-1 nearly universally (*fuir→flee*, *donner→give*, *cassé→broke*, *imaginer→imagine*, *révéler→reveal*), and morphological features map structurally (*écrivent→write*, *révélé→revealed*). We present twenty retrievals with commentary in Appendix I. The decisive contrast is the matched-architecture English model that failed to acquire English grammar: alignment drops to p@1 of 25.00% despite training on 6.50B tokens (71× more than ours, and 50% more than GPT-2 on a per-token basis). Structural alignment tracks acquired grammatical competence, not duration or architecture.

4.4 CL GLUE Experiment Grid

We test four LoRA-adapted levers for CL GLUE on each reproduction seed (Hu et al., 2022), varying the task-data language, the LoRA rank, and the number of epochs; the key contrast is `D+C` (task data translated to French, rank-16 LoRA) against the English-LoRA `Baseline`. The full lever specification is in Appendix D. An inference-time vocabulary-axiom alternative (lever `B`) is reported separately in Section 5.2, and LoRA leaves the French base model bit-identical (Section 4.4). The translation lever (`D+C`) does not dominate uniformly across tasks but produces a sharp task-type gradient (Table 3): the relational tasks BoolQ (+3.45pp), RTE (+5.13pp), and MRPC (+2.89pp) gain measurably over the English-LoRA baseline, whereas the world-knowledge task MNLI regresses (−11.10pp) and the discourse-coreference task WSC is statistically inconclusive (−1.92pp with $\sigma = 8.88$pp). The mean across the five tasks is therefore approximately flat (−0.31pp), but the per-task structure is the contribution: training-language adaptation helps where the task probes structural relations and not where it probes encyclopedic content or extended discourse. We state this gradient as conditional on two controls. First, the per-task deltas of `D+C` over the `Baseline` conflate LoRA rank (8 vs 16) with the language of the task data (English vs French-translated); lever `C` (rank-16 English-LoRA, run on all five tasks across the 5 seeds) isolates the rank effect within a single language and shows no consistent gain over the rank-8 `Baseline`, so the structure in `D+C` is attributable to training-language adaptation rather than to rank. Second, we do not run the symmetric control, i.e. the same French-translated + rank-16 LoRA pipeline applied to an English-native child-scale model; without it we cannot fully exclude that

Task	Baseline	A	C	D+C	Δ
BoolQ	62.36 ± 0.33	63.10 ± 0.72	62.89 ± 0.13	$\mathbf{65.81 \pm 0.51}$	$+3.45$
RTE	53.14 ± 2.65	53.36 ± 2.71	52.85 ± 1.69	$\mathbf{58.27 \pm 2.39}$	$+5.13$
MRPC	68.77 ± 0.93	69.22 ± 0.91	68.58 ± 0.84	$\mathbf{71.67 \pm 0.73}$	$+2.89$
WSC	58.46 ± 4.16	$\mathbf{59.42 \pm 2.19}$	58.85 ± 4.05	56.54 ± 8.88	-1.92
MNLI	63.31 ± 0.93	$\mathbf{64.47 \pm 0.74}$	64.29 ± 0.58	52.21 ± 3.83	-11.10
Mean	61.21	61.91	61.49	60.90	-0.31

Table 3: CL GLUE grid (mean ± std across 5 seeds); best per task in bold. Δ is the gain of D+C over the Baseline. MultiRC and QQP held at baseline values for budget reasons. Inference-time vocabulary axioms (lever B) are placebo-tested separately in Section 5.2.

the pipeline benefits any model regardless of pretraining language. We flag this control as follow-up work (Section 6).

LoRA preserves the base by construction. Each per-task LoRA adapter applied to the epoch-1 checkpoint yields an identical 83.53% on QFrBLiMP because the base parameters are bit-identical and the adapter weights do not enter the autoregressive head. Full fine-tuning on BoolQ drifts the QFrBLiMP score by -0.17pp to 83.36%. We adopt LoRA as the headline adaptation method on the strength of this construction-time preservation guarantee, which we argue is methodologically preferable to post-hoc verification at child scale, where every grammatical-competence parameter is hard-won. Full ablation in Appendix L.

5 Methodological Findings and Recommendations

We report four ablations whose results we read as constraints on the interpretation of child-scale zero-shot scoring. None are pre-registered; they emerged during the preparation of submissions and are presented as forensic diagnostics rather than confirmatory tests. Together they support a single observation: at 125M parameters and 100M words, single-token log-probability scoring on short-sequence benchmarks responds principally to lexical-distributional properties of the tokenizer and the answer template, with task-level competence a secondary signal at best.

5.1 Tokenizer-Swap on GLUE-Axiomatic

We initially trained a substantially reallocated 93M-word corpus ("v2") with 38M words of EWoK-targeted Wikipedia, 25M words of instruction data in GLUE format, and a 16K CDS-trained BPE tokenizer. The v2 model regressed by 7.70 percentage points on GLUE-axiomatic relative to the simpler v1 corpus (Table 4, rows v1 and v2). To

Variant	QFrBLiMP	EWoK	GLUE-ax
v1 (chck_92M)	83.53	50.95	51.00
v2 (full reallocation)	82.68	50.59	43.30
v3a (v1 recipe rebuild)	83.08	51.59	46.10
v3b (v3a + 15M Opus)	83.30	50.50	50.70
v3c (v3a + 38M EWoK Wiki)	83.08	52.32	50.70
v3d (v3a + 16K tokenizer)	82.68	51.36	**43.50**

Table 4: Single-intervention ablation isolating which v2 component caused the 7.70pp GLUE-axiomatic collapse. Only v3d, the tokenizer swap, reproduces the v2 floor.

isolate which intervention caused the regression, we trained four additional models on the v1 corpus, varying one component at a time: v3a rebuilds v1 with the same 50K Wikipedia tokenizer; v3b replaces 15M words of Wikipedia with the format-targeted Opus instructions from v2; v3c replaces 38M words of Wikipedia with the EWoK-targeted Wikipedia from v2; v3d uses the v3a corpus but the 16K CDS-trained tokenizer from v2.

v3d alone reproduces the entire v2 GLUE-axiomatic collapse: 43.50% versus v2's 43.30%, with the same corpus as v3a, v3b, and v3c. The format-targeted instructions and the EWoK-targeted Wikipedia, when introduced individually on top of the v1 recipe, do not regress GLUE-axiomatic. The 16K CDS tokenizer swap, holding all other variables constant, does.

A second observation reinforces the reading. v3a, the clean rebuild of v1 with a nominally identical recipe, scores 4.90pp lower than v1 itself. The difference was accidental, not designed: a different pre-tokenization shuffle seed reordered the sentences of an otherwise identical corpus. It changes neither the source material nor the word budget, yet it produces GLUE-axiomatic swings comparable to differences between models (Appendix J, rebuild_seed_sweep.json). The score is a property of the tokenizer's vocabulary against the scoring protocol's answer-template tokens, not of task-level competence. The fine-tuned LoRA numbers we adopt as the headline GLUE results (Section 4.4) replace single-token scoring with a learned decision boundary and are not subject to this artifact. Any method comparing log-probabilities of single-token answers across prompts is vulnerable, and a benchmark that does not report tokenizer-swap sensitivity cannot distinguish task-competence from tokenizer-vocabulary differences. Full v3 protocol and v2 details in Appendix J and Appendix M.

Task	Bare	Targeted	Placebo	Tautology
BoolQ	45.20	49.20	50.00	48.20
RTE	54.00	54.00	54.00	54.00
MNLI	32.20	35.20	35.20	34.00
Mean	43.80	46.10	**46.40**	45.40

Table 5: Dictionary-axioms placebo control. Targeted: relevant FR–EN pairs. Placebo: random, unrelated FR–EN pairs of the same number. Tautology: en=en pairs containing zero translation information.

5.2 Placebo Control on Dictionary-Axiom Prompting

An inference-time approach that prepends a French–English vocabulary axiom block to each English benchmark item produced apparent gains of 5–9 pp on three GLUE tasks (BoolQ, RTE, MNLI). A placebo control falsified the interpretation. We re-ran with three axiom variants under the same French answer template (Table 5):
Targeted minus bare is $+2.33$pp; placebo minus bare is $+2.60$pp; tautology minus bare is $+1.60$pp. The translation-specific effect (targeted minus placebo) is -0.27 pp: random, unrelated axioms produce as much improvement as targeted ones. The semantic content of the axiom block is irrelevant; the gain is structural prompting noise. The original 5–9pp effect was further inflated by an answer-template confound (English template without axioms vs. French template with axioms); with template held constant, the structural prompting effect is approximately 2pp. We retain this as a null result: structural prompting at the child scale yields a small, content-independent GLUE gain but does not constitute evidence of CL lexical transfer via vocabulary mappings.

5.3 Translated BLiMP Supplement and EWoK at Child Scale

Two further results bound interpretation. The translated BLiMP Supplement score (34%, Section 4.2) is below chance because the items contain a systematic lexical-frequency confound: in *qa-congruence-easy*, for instance, the "good" French sentence consistently has a less-frequent content word (*médecin* rather than *chaise* as the subject of an embrassée-predicate) than the "bad" sentence, and the model prefers the higher-frequency alternative regardless of grammaticality. The score measures translation-specific lexical frequency in the training distribution, not the pragmatic inference phenomena that the English original probes; we report 34% as honest evidence rather than as a competence claim. EWoK at $n = 2200$ falls inside the 95% chance interval $[47.90\%, 52.10\%]$ across all four interventions tested (raw pretraining, EWoK-targeted Wikipedia, dictionary axioms, retrieval-augmented prompting), with conditions ranging 49.86-50.95%. The right tool for world knowledge at 125M / 100M words does not exist within the BabyLM envelope; inference-time tricks cannot substitute for parameters or data.

5.4 Recommendations

The unifying observation across Section 4 and Section 5 is that no single instrument fits every evaluation decision at the child scale. Table 6 (Appendix A) summarises the mapping for the seven decisions our pipeline made: each row is backed by an experiment in this paper, and every wrong-tool entry was tested and produced a null or negative result.

Native-language evaluation as the preferred axis. The CL gap between QFrBLiMP (85.97%, native) and translated BLiMP (76.28%) is a 9.69-point gap on essentially the same kind of probe, differing only in whether the items are native or translated; translated minimal-pair benchmarks at child scale conflate model competence with translation quality and lexical-frequency artifacts (Section 5.3). Where a native-language benchmark exists, it is the cleaner instrument; otherwise a placebo-controlled translation pass is a workable second-best.

Recommended reporting metrics. For child-scale submissions scored by single-token log-probability on short-sequence benchmarks, two diagnostics should accompany headline accuracy: tokenizer-swap sensitivity (re-evaluating under a second BPE tokenizer trained on a different distribution) and placebo-controlled prompting (re-evaluating any prompt-prepended scoring under an unrelated-content prompt of matched length). Without them, a benchmark cannot separate task competence from tokenizer-vocabulary or prompt artifacts.

6 Conclusion

MéTRON-FR demonstrates that French-only child-scale pretraining produces a BabyLM submission scoring 85.97% on QFrBLiMP, above the best native-BLiMP score in any other 2026 official language (Section 1), and 62.80% on the BabyLM-weighted metric, at or above the re-evaluated Strict-

track reference points (Appendix G), while exposing measurement gaps within the suite.

We do not argue that this submission supersedes the architectural line of work (Charpentier and Samuel, 2023, 2024); we test a different axis, the choice of pretraining language, enabled by the 2025 rule change. Native-language minimal-pair benchmarks and protocol-level diagnostics belong in the standard reporting set for child-scale submissions relying on single-token zero-shot scoring.

Limitations

The submitted checkpoint is from a single pretraining seed; for variance estimation, we report a 5-seed reproduction (Section 3.1) on the headline downstream evaluations (QFrBLiMP, QFrCoLA, BLI, CL GLUE). The 4.90 percentage-point gap on GLUE-axiomatic between two nominally identical recipe rebuilds (Section 5.1, v1 vs. v3a) is a separate, training-time noise floor that bounds the resolution of any single-seed claim on this benchmark to approximately 5pp at this scale; we adopt fine-tuned LoRA numbers as the GLUE headline partly because they are not subject to this artifact.

Several quantitative sub-claims in this paper sit within or near that noise floor and should be read accordingly. The per-task gains in Table 3 (BoolQ $+3.45$pp, RTE $+5.13$pp, MRPC $+2.89$pp; std ≤ 2.70pp on the gain) are above the noise floor; the WSC and MNLI deltas are within or below it (WSC $\sigma = 8.88$pp on `D+C`; MNLI std small but the regression magnitude is large), and the relative differences between v3a, v3b, and v3c in Table 4 are of comparable magnitude to the rebuild noise. The qualitative conclusions of those experiments, that the translated-data lever (`D+C`) helps short-input relational tasks while the inference-time-axiom lever (`B`) does not, and that only the tokenizer swap (`v3d`) reproduces the v2 GLUE-axiomatic floor, do not depend on point-estimate precision and survive within the noise floor; the per-task point estimates do.

The Quebec-French normative scope of QFrBLiMP and QFrCoLA is a deliberate choice but a real limitation. Anglicism-sensitive findings do not generalize to metropolitan French without revalidation, since the metropolitan norm is more permissive on lexical anglicisms (Appendix K); morphological and syntactic findings are expected to generalize.

The single-language design cannot adjudicate whether the observed phenomena, namely the native-vs-translated minimal-pair gap, the BLI competence-vs-duration contrast, and the tokenizer and placebo sensitivities, are French-specific or general to child-scale non-English pretraining. Replicating the protocol across the BabyBabelLM languages (Jumelet et al., 2026a) with native minimal-pair instruments would test whether the right-tool mapping (Table 6) holds cross-linguistically.

The methodological findings in Section 5 are post hoc forensic diagnostics that emerged during the preparation of the submission, not preregistered confirmatory tests; we report them as constraints on interpretation rather than as confirmatory claims. The Haitian-Creole lexical oracle is a corpus-curation heuristic whose contribution is not isolated against an unweighted baseline within this paper; the v3a clean-rebuild result (Section 5.1) bounds any oracle-specific contribution to within the noise floor of the rebuild itself.

Our CL GLUE results (Section 4.4) are not directly comparable to English-trained baselines such as ELC-BERT or GPT-BERT (Charpentier and Samuel, 2023, 2024): those models report English-native fine-tuning on English task data, while ours reports rank-16 LoRA on French-translated task data. Two specific controls are missing from the present protocol. First, the per-task gains over the English-LoRA baseline conflate LoRA rank (8 vs 16) with the language of the task data (English vs French-translated); the rank-16 English-LoRA condition (Table 3, lever `C`, run on all five tasks in the 5-seed reproduction) does isolate the rank effect within the same language and shows no consistent gain over rank-8, supporting the reading that the per-task structure in lever `D+C` is driven by training-language adaptation rather than rank. Second, applying the same translated-French + rank-16 LoRA pipeline to an English-native child-scale model would test whether the protocol benefits any model regardless of pretraining language, or whether the gain is specific to having French-trained representations to adapt. This second control was outside the scope of computation for this submission and is flagged for camera-ready or follow-up work.

Ethical Considerations

The training corpus is assembled from public-domain or research-licensed French sources: CHILDES via `childes-db` (MacWhinney, 2000), French Wikipedia, and the openly released *babylm-*

fra component of BabyBabelLM (Jumelet et al., 2026a). The hand-curated bilingual lemma bridge is released as an evaluation resource for the BLI diagnostic and is not part of the training corpus. No human-subjects data is collected for this study, and no individually identifying content is introduced beyond what is already published in the source corpora; CHILDES transcripts are used within their documented research-release terms.

The Haitian-Creole lexical oracle uses a high-frequency word list as an instrument to oversample French sentences; no Haitian-Creole text enters training, and the procedure does not model, generate, or make claims about Haitian Creole as a language. We note the asymmetry: a creole derived from French serves here as an analytic tool for a French-only model, and we make no inference about Haitian Creole grammar or its speakers from the procedure.

QFrBLiMP encodes the prescriptive norm of the Office québécois de la langue française, an autonomous regulatory body whose recommendations differ from metropolitan French usage on the items the anglicism category probes (Appendix K); submissions evaluated on QFrBLiMP should report this scope alongside the score.

The released model is a 125M-parameter French language model trained on 92.47M words. At this scale, generation quality is limited, and the model is not appropriate for user-facing applications; we release it as a research artifact for the BabyLM community, not as a deployment-ready system.

Use of AI Assistants

Generative AI assistants (large language models) were used during the preparation of this submission. Their use was limited to: (i) copy-editing and language correction of author-written prose, including rephrasing for clarity and proofreading in the style of a Grammarly-type tool; and (ii) code-review and error-detection assistance applied to scripts that had been written and initially validated by the authors. All code in this submission was originally written by the authors; AI assistants were used to audit and review it, not to produce it. AI assistants were not used to generate research ideas, conduct experiments, run analyses, write substantive scientific argumentation, or perform literature search. The authors take full responsibility for all content.

Acknowledgements

This research was made possible thanks to the support of a Canadian insurance company, NSERC research grant RDCPJ 537198-18 and an FRQNT doctoral research grant. We thank the reviewers for their comments regarding our work.

References

David Beauchemin and Richard Khoury. 2025. QFrCoLA: A Quebec-French Corpus of Linguistic Acceptability Judgments. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 119–130, Suzhou, China. Association for Computational Linguistics.

David Beauchemin, Yan Tremblay, Mohamed Amine Youssef, and Richard Khoury. 2025. COLE: A Comprehensive Benchmark for French Language Understanding Evaluation. *Preprint*, arXiv:2510.05046.

David Beauchemin, Yan Tremblay, Mohamed Amine Youssef, and Richard Khoury. 2026a. Idiom Understanding as a Tool to Measure the Dialect Gap. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 505–522, San Diego, California, United States. Association for Computational Linguistics.

David Beauchemin, Pier-Luc Veilleux, Richard Khoury, and Johanna-Pascale Roy. 2026b. QFrBLiMP: A Quebec-French Benchmark of Linguistic Minimal Pairs. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 4274–4304, Rabat, Morocco. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the BabyLM Challenge: Accelerating Language Modeling Research with Cognitively Plausible Data. In *Proceedings of the BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not All Layers Are Equally as Important: Every Layer Counts BERT. In *Proceedings of the BabyLM Challenge at the Conference on Computational Natural Language Learning*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why Not Both? In *Proceedings of the BabyLM Challenge at the Conference on Computational Natural Language Learning*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and

Ethan Gotlieb Wilcox. 2026. BabyLM Turns 4 and Goes Multilingual: Call for Papers for the 2026 BabyLM Workshop. https://babylm.github.io/. *Preprint*, arXiv:2602.20092.

Jonathan Frankle and Michael Carbin. 2019. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Leshem Choshen, Ryan Cotterell, Alex Warstadt, and Ethan Gotlieb Wilcox, editors. 2024. *The BabyLM Challenge at the Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Miami, FL, USA.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of World Knowledge (EWoK): A Cognition-Inspired Framework for Evaluating Basic World Knowledge in Language Models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galván-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A Multilingual Benchmark of Developmentally Plausible Training Data. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A Massively Multilingual Benchmark of Linguistic Minimal Pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word Translation Without Parallel Data. In *International Conference on Learning Representations*.

Claire Lefebvre. 1998. *Creole Genesis and the Acquisition of Grammar: The Case of Haitian Creole*. Cambridge University Press, Cambridge.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk, Volume II: The Database*, 3rd edition. Lawrence Erlbaum Associates, Mahwah, NJ.

Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2024. Dissociating Language and Thought in Large Language Models. *Trends in Cognitive Sciences*, 28(6):517–540.

B. W. Matthews. 1975. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting Similarities Among Languages for Machine Translation. *arXiv preprint arXiv:1309.4168*.

Salikoko S. Mufwene. 2001. *The Ecology of Language Evolution*. Cambridge University Press, Cambridge.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models Are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the EMNLP Workshop BlackboxNLP*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM Challenge: Sample-Efficient Pretraining on Developmentally Plausible Corpora. In *Proceedings of the BabyLM Challenge at the Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adam Zachary Wasserman. 2026. The Scaling Hypothesis Is Language-Contingent: Evidence from Cross-Linguistic Training Dynamics. SSRN preprint 7115418.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. *arXiv preprint arXiv:2205.01068*.

A Right Tool, Right Job Summary

Job	Right tool	Wrong tool tried
Corpus weighting	Cross-linguistic evolutionary filter (Section 3.4, exploratory)	Editorial intuition
Grammatical competence	French pretraining	English at child scale
Acceptability judgment	Fine-tune classification head (Section 4.1)	Zero-shot perplexity
CL GLUE	LoRA on translated task data (Section 4.4)	Inference-time vocabulary axioms (Section 5.2)
CL structure	Procrustes on frozen embeddings (Section 4.3)	Direct task transfer
World knowledge	More parameters or data (out of constraint)	Any child-scale intervention (Section 5.3)
Adaptation preservation	LoRA (Section 4.4)	Full fine-tuning

Table 6: Right tool, right job: each row is backed by an experiment in this paper; every wrong-tool entry was tested and produced a null or negative result.

B Evaluation Metric Details

For QFrBLiMP, BLiMP, BLiMP Supplement, and EWoK, we score zero-shot by computing the per-token loss on each candidate sentence and selecting the lower-loss alternative, and report item-level accuracy aggregated per paradigm (i.e. per fine-grained linguistic phenomenon, e.g. *determiner-noun agreement*) and across the benchmark. For QFrCoLA, we fine-tune a classification head on the in-domain train set and report accuracy and the Matthews Correlation Coefficient (MCC) (Matthews, 1975); chance MCC is 0.00, and a fine-tuned BERT-base on English CoLA reaches MCC 0.52 at substantially larger pretraining scale, which we use as an upper-bound reference. For CL GLUE, we fine-tune using rank-16 LoRA (Hu et al., 2022) on French-translated task data and report per-task accuracy, averaged across the five tasks (Section 4.4). For BLI, we report the orthogonal Procrustes alignment fit and word-translation precision@k on a held-out seed-dictionary split (Section 4.3; Appendix H).

C Haitian-Creole Oracle Procedure

We extract the top 300 high-frequency Haitian-Creole (HC) lemmas and map them to a 333-lemma French cognate set with multi-form expansions such as *nou* → *nous* / *on* / *notre* / *nos*. Each French training sentence is scored by target-lemma density, and high-density sentences are oversampled at weight 3.00 against a unit-weight default (Section 3.4). No Haitian-Creole text enters training; the lemmas act as a lexical filter on French source material only.

D CL GLUE Lever Specification

The four LoRA levers of Section 4.4 (Table 3) are: 1) `Baseline` (rank-8 English LoRA, 3 epochs); 2) `A` (baseline + 5 epochs); 3) `C` (tuned rank-16 LoRA on English task data, $\alpha = 32$); 4) `D+C` ("E3": GLUE train/eval translated to French, then rank-16 LoRA). Lever `B` (inference-time vocabulary axioms) is placebo-tested in Section 5.2.

E Reproduction Checklist

Per-seed epoch selection. Each of the 5 reproduction seeds ([42–46]) is trained for 5 epochs and evaluated at each epoch on QFrBLiMP; the per-seed best epoch is selected by argmax of QFrBLiMP overall accuracy and used as input to the downstream evaluations. The submitted headline checkpoint is epoch 3, seed 42.

Table 7 summarises the model and training specification; Table 8 lists the scripts and output paths for each experiment reported in the main body. All scripts assume the submission directory as the working directory and are invoked via `uv run python`. Corpus word counts are whitespace-counted per the Strict-track convention, after deduplication. The French Wikipedia component (Section 3.3) is filtered to exclude pure list pages, infobox text, and reference dumps.

F Per-paradigm Translated BLiMP

Table 9 reports the 38-paradigm breakdown for the translated BLiMP evaluation of MéTRON-FR at epoch 3 (Section 4.2). Each paradigm has 200 unique minimal pairs; chance is 50%. Fifteen paradigms score above 80%; the lowest-scoring paradigms cluster in Principle A (anaphor binding) and gender-agreement categories, where French–English grammatical differences make the translation lossiest.

Component	Specification
Architecture	125M GPT-2
Layers / dim / heads	12 / 768 / 12
Batch / sequence	32 / 512
Optimizer	AdamW (defaults)
Learning rate	5×10^{-4}
LR warmup	1,000 steps
Epochs	5
Hardware	single RTX 4090
Time / cost per epoch	$\approx$24 min / $\approx$\$0.25
Total project cost	$\approx$\$65
Tokenizer	50K BPE, FR training corpus
Submitted checkpoint	epoch 3 (of 1–5)

Table 7: Model and training specification.

G Strict-track Reference Comparison

Table 10 reproduces the Strict-track reference points cited in Section 4.2 under the 2025 evaluation pipeline, drawn from Charpentier et al. (2025, Table 3).

H BLI Method Details

Given two frozen embedding matrices E_{FR} and E_{EN} and a seed dictionary, we learn an orthogonal transformation $W \in \mathbb{R}^{768\times768}$ minimizing $\|E_{\mathrm{FR}}W - E_{\mathrm{EN}}\|_F$ via closed-form Procrustes, and evaluate word-translation precision@k by cosine nearest-neighbour retrieval in the mapped space (Section 4.3). The reported fit is $1 - \|E_{\mathrm{FR}}W - E_{\mathrm{EN}}\|_F^2 / \|E_{\mathrm{EN}}\|_F^2$ (1.00 perfect, 0.00 no alignment).

The seed dictionary starts from the 715 form-level (FR, EN) pairs enumerated from the 73-entry bilingual lemma bridge of Section 3.3, a resource released with the submission that is not part of the training corpus. The *language-aware parser pass* is a morphology-aware normalization step: it lemmatizes and, where needed, segments each surface form on both the French and English sides so that inflected variants map onto their base form, drops pairs that survive in only one vocabulary, and deduplicates the result to the (FR, EN) pairs that are single-token in both embedding matrices (a requirement of the closed-form Procrustes map). After this pass, 242 unique pairs are retained; 194 are used to fit W and 48 are held out for precision@k evaluation.

I BLI Alignment Qualitative Examples

Table 11 shows twenty representative held-out French-to-English word-translation retrievals from the BLI experiment (Section 4.3), under the learned orthogonal map W aligning MéTRON-FR's embedding matrix to GPT-2's. *Rank* is the position of the gold English translation in the top-k cosine-similarity nearest neighbours. Content-word mappings reach rank 1 in most cases; function-word mappings and morphologically complex inflections frequently succeed at rank 2–5.

J Tokenizer Ablation Full Protocol

The v3a–v3d ablation (Section 5.1, Table 4) was run with the same architecture, batch size, sequence length, optimizer, learning-rate schedule, and random seed (42) as the v1 submission. The four v3 models were trained for one epoch each on their respective corpora; QFrBLiMP is reported on the resulting `chck_92M`-equivalent checkpoints. Tokenizer training corpora and seeds:

- `50K-FR BPE` (v1, v3a, v3b, v3c): trained on the full v1 training corpus (`corpus/final/train_french.txt`, 92.47M words), vocab size 50,000 including the two special tokens `<|endoftext|>` and `<|padding|>`, byte-level pre-tokenizer with `add_prefix_space=False` and byte-level decoder, and no Unicode normalization step. BPE training is deterministic given the corpus and the vocabulary size, so no tokenizer seed is involved.
- `16K-CDS BPE` (v2, v3d): trained on the 25M-word CHILDES + babylm-fra component of the v1 corpus, vocab size 16,000, otherwise identical to the `50K-FR` configuration.

The GLUE-axiomatic evaluation set is the inference-time axioms protocol of Lever B (Section 4.4) on the five translated GLUE tasks, scored using the lowest-loss-token-among-the-template-options heuristic. The full release includes `scripts/train_tokenizer_exp.py`, the four trained tokenizer files, the v3a–v3d corpus chunks, and the per-task scoring logs at `submission/eval_logs/full_v3_ablation/`. The 4.90pp v1-vs-v3a rebuild noise (a positive control: same recipe, different seeds in pre-tokenization shuffle) is documented in the same log directory under `rebuild_seed_sweep.json`.

Experiment	Scripts	Outputs / notes
Pretraining (Section 3)	`build_french_corpus.py`, `train.py`	`model/chck_92M_epoch3/`
QFrBLiMP (Section 4.1)	`eval_qfrblimp.py`	1,761 minimal pairs
QFrCoLA (Section 4.1)	`finetune_qfrcola.py`	3 epochs, learning rate 2×10^{-5}
Translated BLiMP / Suppl. (Section 4.2)	`eval_blimp_translated.py`	translations under `fast_eval_improved/`
BLI (Section 4.3)	`bli_procrustes.py` (GPT-2); `bli_procrustes_fractal.py` (failed-EN)	242 FR–EN pairs (194 train, 48 held out)
CL GLUE (Section 4.4)	`translate_glue_fr.py`; `finetune_glue_lora_fr_tuned.py`	7 per-task LoRA adapters, rank 16, $\alpha=32$, 3 epochs
Tokenizer-swap v3 (Section 5.1)	`train_tokenizer_exp.py`, `build_v3_corpora.py`	logs under `full_v3_ablation/`
Dict-axioms placebo (Section 5.2)	`eval_dict_axioms_*.py`	logs under `dict_axioms/`

Table 8: Scripts and outputs for each experiment.

K Quebec French Normative Framework

QFrBLiMP encodes the prescriptive norm of the *Office québécois de la langue française* (OQLF), an autonomous regulatory body created under the *Charte de la langue française* (Loi 101, 1977; expanded by Loi 96, 2022). The OQLF defines normative Quebec French through the *Grand dictionnaire terminologique* and the *Banque de dépannage linguistique*; its mandate is both administrative and descriptive; its recommendations are binding for Quebec public administration, education, and commercial signage. The norm, therefore, has political constitution, not merely lexicographic authority.

The metropolitan French standard, by contrast, is descriptive: the *Académie française* issues recommendations without administrative force, and the metropolitan written-press and dictionary tradition is more permissive on lexical anglicisms in an informal register. The two norms diverge most sharply on the items the QFrBLiMP anglicism category probes:

- *checker* (QC anglicism) vs *vérifier* (OQLF normative). Metropolitan informal usage admits *checker* freely; metropolitan formal register prefers *vérifier*.
- *matcher* (anglicism) vs *correspondre* / *aller avec* (OQLF normative). Metropolitan informal usage admits *matcher* routinely.
- *fun* (anglicism) vs *plaisir* / *amusant* (OQLF normative). Metropolitan informal usage admits *fun* as a noun and adjective.
- *cool* (anglicism) vs *bien* / *super* (OQLF normative). Metropolitan usage admits *cool* across registers.
- *courriel* (OQLF coinage, normative QC) vs *email*/*e-mail* (metropolitan accepts; OQLF prefers *courriel*).

A model evaluated on QFrBLiMP is being asked whether it has internalized the OQLF norm, not French grammaticality in some language-neutral sense. The 80.15% anglicism sub-score (Table 1) reflects two simultaneous mechanisms: (i) phonotactic and morphological internalization of French regularities, which assigns lower probability to anglicisms with non-French coda clusters or stress patterns regardless of normative status, and (ii) sensitivity to the OQLF norm itself, which flags phonotactically licit anglicisms as ungrammatical. The benchmark's strength is a well-defined ground truth backed by a regulatory body with continuous publication; its scope limitation is that performance does not translate one-to-one to a metropolitan-French

informal-register benchmark.

L LoRA Preservation Details

Table 12 reports QFrBLiMP scores after each per-task LoRA adapter is applied to `chck_92M` (epoch 1, baseline value 83.53%). Every LoRA adapter produces an identical 83.53% because the base model's parameters are bit-identical by construction and the adapter weights do not enter the autoregressive language-modelling head used for QFrBLiMP scoring. The submitted epoch-3 checkpoint exhibits identical preservation properties at its own QFrBLiMP baseline of 85.97%. Full fine-tuning on BoolQ drifts the QFrBLiMP score by -0.17pp, indicating mild catastrophic forgetting at this scale.

The result is a parameter-efficient instance of the lottery-ticket hypothesis (Frankle and Carbin, 2019) at the adaptation layer rather than the pre-training layer: 0.65% of total parameters (the rank-8 baseline LoRA) or 1.29% (the rank-16 tuned LoRA) suffices to encode per-task downstream behaviour, while the remaining $\geq 98.70\%$ serves as a fixed substrate. The construction-time guarantee replaces the empirical claim "we measured QFrBLiMP after fine-tuning, and it stayed high" with a structural one: the base parameters are bit-identical, so the QFrBLiMP score is unchanged by definition.

M Additional Methodological Details

v2 corpus reallocation (Section 5.1). The v2 model was trained on a 93M-word reallocation: 25M words of grammatical base content (CHILDES + babylm-fra, identical to v1), 38M words of EWoK-targeted Wikipedia (filtered for vocabulary relevant to the physical, social, and spatial domains targeted by EWoK), 15M words of GLUE-format Opus instructions in French, 10M words of supplement-targeted Opus instructions, and 4.60M words of diverse instruction data. v2 regressed on every metric (Table 13).

A per-item diagnosis on the BoolQ validation set showed v2 predicting "non" (no) for 97.60% of items, while v1 predicted "oui" (yes) for 66.60% (gold yes-rate is 62.20%). The v2 regression is therefore a token-calibration collapse rather than a task-level capability loss: the format-targeted instruction data shifted the output distribution toward "non", and the resulting accuracy regression is the mechanical consequence of a benchmark whose gold distribution favours "yes". The single-intervention v3 ablation (Section 5.1) localized this calibration shift to the 16K CDS-trained tokenizer.

EWoK by-domain breakdown. Table 14 shows the four interventions across the eleven EWoK domains. No domain shows statistically significant separation from chance for any intervention.

Dict-axioms placebo answer-template confound. The original 5–9pp axiom effect (Section 5.2) compared an English answer template (no axioms) against a French answer template (with axioms), conflating the template change with axiom introduction. With the template held constant, the structural prompting effect (targeted minus bare under the matched French template) is approximately 2 pp; the translation-specific effect (targeted minus placebo) is -0.27 pp.

Paradigm	Accuracy
anaphor gender agreement	43.50%
anaphor number agreement	74.00%
animate subject transitive	78.00%
coordinate structure (complex left branch)	87.50%
coordinate structure (object extraction)	56.50%
determiner-noun agreement 1	93.50%
determiner-noun agreement 2	100.00%
determiner-noun agreement (irregular 1)	85.00%
determiner-noun agreement with adjective 2	98.50%
distractor agreement (relative clause)	78.50%
ellipsis n-bar 1	77.00%
expletive it, object raising	72.50%
inchoative	77.50%
intransitive	74.00%
irregular past participle adjectives	82.50%
irregular past participle verbs	76.50%
irregular plural subject-verb agreement 1	86.50%
irregular plural subject-verb agreement 2	86.50%
left-branch island (simple question)	92.00%
matrix question negative polarity item licensor present	80.00%
negative polarity item present 1	78.00%
only-negative polarity item scope	98.50%
passive 1	73.00%
principle A (case 1)	43.00%
principle A (case 2)	82.00%
principle A (domain 3)	48.50%
principle A (reconstruction)	51.50%
sentential negation negative polarity item licensor present	100.00%
sentential subject island	69.00%
superlative quantifiers 2	86.00%
tough-vs-raising 1	65.00%
tough-vs-raising 2	55.00%
transitive	67.50%
wh-island	92.00%
wh-question subject gap	74.50%
wh-question subject gap (long distance)	93.00%
wh-vs-that, no gap, long distance	80.50%
wh-vs-that, gap, long distance	41.50%
Overall	76.30%

Table 9: Per-paradigm translated BLiMP accuracy for MéTRON-FR at epoch 3.

Model (Strict track)	NLP	Macro
GPT-BERT, 7 variants	57.8–63.0	40.8–43.5
GPT-2 naive baseline	55.4	36.2
CLASS-IT (2025 HL win.)	52.9	36.6
Simple-Diffusion (2025 NLP win.)	58.4	35.5
MéTRON-FR (2026)	**62.80 (weighted)**	

Table 10: Strict-track reference scores under the 2025 evaluation pipeline (from Charpentier et al., 2025, Table 3): NLP-task average (GLUE-style fine-tuning) and macro average. GPT-BERT (Charpentier and Samuel, 2024) is the prior challenge winner re-evaluated as the 2025 baseline; the seven variants combine {causal, mixed, masked} training objectives with {MNTP, AR} evaluation styles. MéTRON-FR's BabyLM-weighted score is a single composite reported by the same pipeline.

FR held-out	EN gold	Rank	Top-5 retrieved
écrit	writes	2	write, writes, read, articles, notice
l'	the	1	the, the, a, this, in
allé	gone	1	gone, watched, forgotten, scared, do
nous	us	2	we, us, your, the, the
ce	this	1	this, the, the, a, what
fuir	flee	1	flee, speak, scared, conceal, gone
imaginer	imagine	1	imagine, write, give, see, reveal
par	by	5	in, the, a, the, by
cassé	broke	1	broke, gone, forgotten, scared, revealed
écris	write	1	write, flee, watch, so, read
dans	in	1	in, the, the, a, by
révéler	reveal	1	reveal, revealed, give, conceal, see
regarder	watch	3	do, see, watch, give, watched
nous	we	1	we, us, your, the, the
une	a	1	a, the, the, this, in
donc	so	4	the, the, a, so, this
donner	give	1	give, do, see, speak, reveal
quoi	what	2	what, what, your, a, we
caché	concealed	8	watched, scared, revealed, gone, watch
révélé	revealed	1	revealed, reveal, forgotten, watched, investigating

Table 11: Sample BLI retrievals (MéTRON-FR → GPT-2 via learned orthogonal map). Content verbs (*fuir*, *cassé*, *donner*, *révéler*) and morphologically inflected past participles (*allé*, *révélé*) align at rank 1. Function words (*par*, *donc*) are retrieved at slightly lower ranks but within the top 5.

Configuration	QFrBLiMP	GLUE (E3 best)
`chck_92M` (no FT)	83.53	—
+ E3 BoolQ adapter	83.53	68.26
+ E3 MNLI adapter	83.53	53.79
+ E3 MRPC adapter	83.53	72.55
+ LoRA MultiRC adapter	83.53	57.55
+ LoRA QQP adapter	83.53	73.14
+ E3 RTE adapter	83.53	61.15
+ E3 WSC adapter	83.53	73.08
Full FT on BoolQ	83.36	66.20

Table 12: QFrBLiMP after downstream LoRA fine-tuning. Every adapter produces the same 83.53% because the base parameters are unchanged, and adapter weights do not enter the autoregressive head. Full FT drifts QFrBLiMP by 0.17pp.

Metric	v1	v2	Δ
QFrBLiMP	83.53	82.68	−0.85
GLUE-axiomatic	51.00	43.30	−7.70
EWoK	50.95	50.59	−0.36

Table 13: v1 vs v2 corpus reallocation results. The 7.70pp GLUE-axiomatic regression motivated the v3 ablation (Section 5.1).

Intervention	EWoK	95% CI
v1 raw pretraining	50.95	[47.90, 52.10]
v2 EWoK-targeted Wiki	50.59	[47.90, 52.10]
Dict-axioms (small bridge)	49.86	[47.90, 52.10]
Dict-axioms (big, 452 entries)	50.59	[47.90, 52.10]
Retrieval-augmented (top-3)	50.95	[47.90, 52.10]
Chance (binary)	50.00	—

Table 14: EWoK across interventions. All five conditions fall inside the 95% chance interval $[47.90\%, 52.10\%]$ for $n = 2200$.

What Helped with Huge Data Hurt with Small Data: Standardization, Representational Geometry, and Data Scale

Catherine Finegan-Dollak
University of Richmond
Department of Computer Science
cfinegan@richmond.edu

Abstract

Training models on cognitively plausible (CP) amounts of data (tens or hundreds of millions of words) offers multiple advantages over the huge-data training (hundreds of billions to trillions of tokens) typical of current models, but achieving competitive performance at CP scale remains an open challenge. Understanding differences in representational geometry between CP- and huge-data models may help us design better CP-scale models. We identify differences in two aspects of representational geometry (anisotropy and rogue dimensionality) and a difference in effect of a geometry-targeted intervention.

Although standardization—a technique to address anisotropy and rogue dimensions in LLM representational geometry—improves lexical similarity performance in huge-data models, the effect reverses for CP-data models. We also find that both rogue dimensionality and anisotropy correlate with data scale, further supporting the view that CP- and huge-scale models differ geometrically. However, statistical analysis of results across 18 models showed that neither rogue dimensions nor anisotropy significantly positively mediated the difference in standardization effect, except in one model per mechanism. This absence of mediation is itself a finding: it fails to support the two mechanisms most directly motivated by prior work on standardization, narrowing what plausibly explains the reversal. Thus, the observed qualitative difference in standardization effect may hint at as-yet undiscovered differences, geometric or otherwise, between CP-scale models and models trained on huge data.[1]

1 Introduction

While current large language models are trained on hundreds of billions or even trillions of tokens, humans learn language from a few tens or hundreds of millions of words. Training models on such *cognitively plausible*[2] (CP) data scales offers multiple benefits compared with training on typical huge data scales. First, it may help us understand what is necessary to learn language, leading to a better alignment between NLP and cognitive science. Second, it can democratize AI: reducing the FLOPs required to train allows more people outside of industry giants to experiment. Third, it offers a viable path for NLP in less-resourced languages that do not have hundreds of billions of tokens of text available for training; it likewise benefits those who train models in well-resourced languages while respecting the intellectual property of authors.

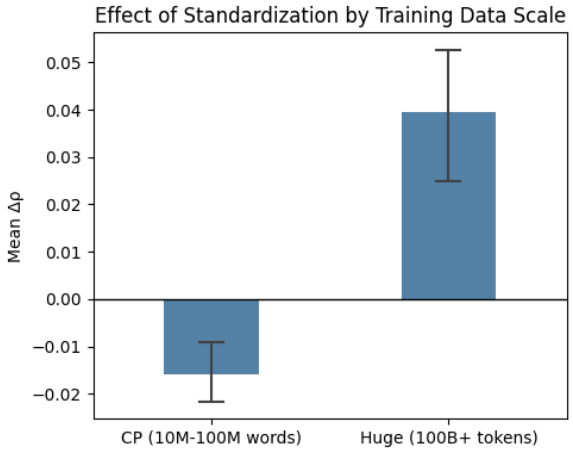


Figure 1: Standardization, a technique that helps models trained on huge data perform better at lexical similarity tasks, hurts performance for models trained on cognitively plausible (CP) data scales.

Understanding the representational geometry of models trained on CP data can help us make progress on improving these models. Lexical simi-

[1]Data and code available at https://github.com/finegan-dollak/data-scale-standardization

[2]In this work, CP refers to the scale of training data. Other characteristics with a bearing on cognitive plausibility, such as data quality or distribution, are not addressed here.

Proceedings of the Second BabyLM Workshop, pages 388–402
October 29, 2026 ©2026 Association for Computational Linguistics

larity is a useful window into representational geometry, since word-pair similarity scores are computed directly from the cosine similarity of the underlying vectors and evaluated against human similarity judgments. It's also a well-studied task with quality, human-generated benchmarks for evaluation.

With the advent of contextual embeddings and models trained on huge datasets, a number of techniques for improving lexical semantics results have emerged, with a great deal of attention paid to isotropy (uniform distribution of words within the embedding space) and its opposite, anisotropy (word vectors clustered in the same direction) (Bommasani et al., 2020; Vulić et al., 2020; Mahajan et al., 2025). Timkey and van Schijndel (2021) observed that anisotropy was often attributable to just a few *rogue dimensions*; they addressed the problem with vector standardization, which improved performance on lexical similarity for several large models.

We find, however, that **despite being effective for models trained on huge data, standardization actually *harms* lexical similarity results for models trained on cognitively plausible data**. Given the motivation for vector standardization in work with huge data models, we hypothesize that rogue dimensionality or anisotropy may explain this. Surprisingly, though, while both geometric phenomena are correlated with training data scale, our statistical analyses show that neither rogue dimensionality nor anisotropy explains the reversal in effect in the general case, though we find one positive mediation result for each mechanism within the training trajectories of the Pythia models. This raises the intriguing possibility that the geometry of models trained on CP-scale data differs from those trained on huge-scale data in ways not captured by existing measures.

Our contributions are as follows:

- Empirical results showing that standardization improves lexical similarity performance on models trained on huge data but harms it for models trained on CP data.
- Results showing that rogue dimensionality and anisotropy are correlated with how much data the model was trained on.
- Analysis showing that neither rogue dimensionality nor anisotropy explains the reversal in effect in the general case.

In short, we have discovered a qualitative difference in how a geometry-targeted intervention affects CP- and huge-data models, revealing a gap in our field's understanding of how CP-data models learn.

2 Background & Related Work

2.1 Lexical Similarity and Representational Geometry

Lexical similarity is the task of determining how similar two words are. Distributional word embeddings in the 2010s showed great improvement over previous methods (e.g., Mikolov et al., 2013). Getting further gains from LLMs has proven challenging, as the naive approach of passing a single word through an LLM and selecting a layer as its representation is ineffective (Bommasani et al., 2020; Ranjan, 2025; Vulić et al., 2020).

Some of this difficulty arises from the anisotropy of the representational space of LLMs. In an isotropic space, word representations are distributed uniformly in all directions from the origin; as a result, randomly selected pairs of words are not terribly similar. However, in an anisotropic space, representations cluster in a cone. Ethayarajh (2019) showed high anisotropy in BERT, ELMo and GPT-2; Mahajan et al. (2025) found similar results for more recent LLMs.

Timkey and van Schijndel (2021) found that a small subset of 1-5 "rogue dimensions" drive anisotropy. They found that the top three dimensions contributed up to 98% of cosine similarity between random word pairs; likewise, rogue dimensions drove the variability in cosine similarity. They found that a simple post-processing vector standardization—subtracting the mean and dividing by the standard deviation—consistently improved performance of several LLMs on multiple lexical similarity tasks.

Other work notes that anisotropy is beneficial for some purposes, and increasing isotropy is not always desirable. Raunak et al. (2020) note that although post-processing to increase isotropy had improved results for semantic similarity tasks, it harmed performance for sentence classification and machine translation tasks. Rudman and Eickhoff (2024) proposed an alternative measure of isotropy and found that increasing isotropy according to this measure led to decreased performance on several common NLP benchmarks.

2.2 Data Scales

The scale of data used to train modern LLMs is enormous and growing; for example, Llama 3 was trained on 15 trillion tokens (Grattafiori et al., 2024), while Llama 4 trained on 30 trillion tokens (AI, 2025). Even more open models, such as Pythia (Biderman et al., 2023) and Gemstones (McLeish et al., 2025) train on hundreds of billions of tokens, and Olmo 3 base trained on 6 trillion tokens (Olmo et al., 2026).

In response to this, some in the NLP community have proposed studying how a computational system can learn language from data at a *cognitively plausible* scale (Choshen et al., 2026; Charpentier et al., 2025; Hu et al., 2024; Warstadt et al., 2023)—that is, at a scale comparable to what a human child hears while learning language. Models trained at smaller scale offer many advantages: They may offer better insights into how the human brain learns language. They can be trained more quickly and on less expensive hardware than models trained on huge data. They could enable more functional NLP in less-resourced languages. The annual BabyLM challenge operationalizes cognitively plausible data scales as *strict* (100M space-separated words) and *strict-small* (10M words), roughly corresponding to the number of words a child of 12 years old (strict) or 2-5 years old (strict-small) has been exposed to in his or her lifetime (Warstadt et al., 2023).

Relatively little work has considered whether models trained on smaller scale data differ from those trained on huge scale data in kind rather than in degree. Previous work on qualitative differences at scale refers to model scale, not data scale (e.g., Wei et al., 2022). Xiao (2025) identified the phenomenon of *scaling law crossover*, where the techniques that benefited small models harmed larger models. While motivated by differences in training needs between small and large *data* scales, the experiments conflate data scale and model scale, preventing conclusions about which explains the crossover.

A few recent works have examined how LLM characteristics vary over training checkpoints. Machina and Mercer (2024) looked at trends in isotropy over 21 evenly spaced checkpoints of several Pythia models. Li et al. (2026) identified phases that the geometry of representational space passed through during training of large (160M-12B parameter) models. Müller-Eberstein et al. (2023) studied when during training various types of linguistic information was encoded in their models.

These aim to understand how and when desirable properties of models trained on large data emerge. To the best of our knowledge, no one has explored the opposite perspective: what might qualitative differences between models from small and large data regimes tell us about how best to train on cognitively plausible data scales?

3 Methods

3.1 Models

We use two main types of models for our experiments: those trained only on CP (strict- or strict-small) datasets, and models that release a sequence of checkpoints that have seen small through huge amounts of training data. For the CP-data-only models, we have a custom baseline, as well as models released as baselines for the BabyLM challenge in 2023, 2024, and 2025. For checkpoint sequences, we use Pythia and Gemstones. In selecting the checkpoint-sequence models, we focus on small versions (tens of millions of parameters), as Chinchilla-style scaling (Hoffmann et al., 2022) suggests this would be the parameter count most appropriate for models to be trained on CP-scale data, and this parameter count is most directly comparable to that of the CP-data models. Model sizes and training data scale are summarized in Table 1.

The Pythia family of decoder-only models (Biderman et al., 2023) was developed to help researchers study how LLMs change over the course of their training and how scale affects those changes. Full Pythia models are trained on 300B tokens of the Pile (Gao et al., 2020). We use two models, Pythia-14M and Pythia-70M. We report the results of the fully trained version of each, as well as all checkpoints.

Gemstones (McLeish et al., 2025) is a similar suite of models, with more variations in architectural shapes and hyperparameters. Their architecture is decoder only and based on Gemma (Team et al., 2024). We use Gemstone-256x23, (256 dimensions and 23 layers) and Gemstone-512x11 (512 dimensions and 11 layers). As with Pythia, we report results across all checkpoints of Gemstone, up to the final checkpoint trained on 350B tokens of Dolma 1.7 (Soldaini et al., 2024) data.

Our custom BaselineLM is a simple causal language model using next-token prediction as its objective function. We train versions on two 10M-word datasets: a 50-50 split of text from Local-

izedNarratives (Pont-Tuset et al., 2020) and the English subset of WIT: Wikipedia-based Image Text Dataset (Srinivasan et al., 2021),[3] and a subset of the Pile (Gao et al., 2020).[4] Our BaselineLM model was therefore trained on two different distributions of text: one more representative of what a multimodal model might encounter, and one from approximately the same distribution as Pythia models. In both cases, the model was trained on approximately 10M words (split on spaces) of text.

All remaining models were trained on the BabyLM train sets. BabyLlama (Timiryasov and Tastet, 2023) used knowledge distillation from an ensemble of GPT-2 and LLaMA models trained on the BabyLM train sets; BabyLlama outperforms its teachers and a comparable non-distilled model trained on the same data. Opt-125M (Zhang et al., 2022) is a decoder-only model used as a baseline in the first BabyLM challenge (Warstadt et al., 2023). GPT-BERT (Charpentier and Samuel, 2024) uses a hybrid masked language modeling and causal language modeling objective. It was the winner of two tracks for the BabyLM shared task in 2024 and was used as a baseline for the 2025 shared task. We use the models released on Huggingface: GPT-BERT causal focus (where the causal objective is weighted more heavily), masked focus (where the masked objective is weighted more heavily), and mixed (where the two objectives are evenly weighted). Finally, we use the versions of GPT-2 Small (Radford et al., 2019) trained as baselines for the BabyLM 2025 challenge (Charpentier et al., 2025).

When we refer to experiments as "across models," we mean that we included the final checkpoint of each model, fully trained on the data scales described in Table 1 for a total of 18 models. Experiments are "within models" when we use the published checkpoints of the Pythia and Gemstone models.

Model	Parameters	Data Scale
Pythia-14M	14M	huge
Pythia-70M	70M	huge
Gemstone 256x23	48M	huge
Gemstone 512x11	95M	huge
BaselineLM	35M	CP (strict-small)
BabyLlama	58M	CP (strict & strict-small)
OPT-125M[5]	163M	CP (strict & strict-small)
GPT-BERT 100M (causal, masked, & mixed focus)	132M	CP (strict)
GPT-BERT 10M (causal, masked, & mixed focus)	39M	CP (strict-small)
GPT-2	111M	CP (strict & strict-small)

Table 1: Models used in experiments, with their parameter counts and training data scale. Following the BabyLM challenges, "strict-small" means 10M space-delineated words of training data and "strict" means 100M words; both are cognitively plausible (CP). "Huge" here refers to hundreds of billions of tokens.

3.2 Lexical Similarity

We use SimLex-999 (Hill et al., 2015) for our evaluation. This well-established dataset contains 999 word pairs with similarity scores based on human annotations.

The established approach to lexical similarity of two words, A and B, is to take the cosine similarity of the vector representations for A and B. Performance is measured by Spearman's ρ correlation between the cosine similarities and the gold similarity scores; high ρ indicates better performance, and human inter-annotator agreement for SimLex was $\rho = 0.67$.

Large language models introduce some options for what vector representations to use. First, it is not obvious which layer or layers encode the relevant information; some work supports using layer 0, the word embedding layer (Wang et al., 2019; Conneau et al., 2020), while others support averaging across some or all layers (Liu et al., 2019; Vulić et al., 2020). Bommasani et al. (2020) reported that lower layers typically performed best; however, as they pooled over an increasing number of contexts, the best-performing layer shifted higher in the pretrained models. We therefore evaluate on each layer of each model.

Since SimLex contains whole words and language models typically use subword tokenization, a second question arises as to which subword to use or how to combine multiple tokens in a single word. We follow Bommasani et al. (2020), who found mean pooling of subword vectors to yield the best results.

[3] Although these datasets offer images associated with the text, we do not use those images in training.

[4] We used the `monology/pile-uncopyrighted` version, https://huggingface.co/datasets/monology/pile-uncopyrighted, which excludes subsets of the Pile known to contain copyrighted materials. We used stratified sampling based on the original proportions of subsets (excluding copyrighted ones) to collect 10M words of text.

[5] 125M in the name refers to parameter count without the LM head.

Finally, since language models produce contextualized embeddings, one must choose the context for the words. Prior work has established that passing a word in isolation through the network is not as effective as averaging the representations obtained from the word in multiple sentences (Bommasani et al., 2020; Vulić et al., 2020). Since Vulić et al. observed that an average over 100 contexts yielded only marginal performance gains compared with average over 10 contexts, we use 10 contexts per word to allow more efficient experiments. Our context sentences are drawn from the Wikipedia dump[6] (Foundation, 2023). No effort was made to disambiguate word sense, as SimLex itself does not specify a word sense. We confirmed with preliminary experiments (reported in Appendix A) that use of these context sentences significantly outperformed isolated word embeddings.

3.3 Operationalizing Anisotropy & Rogue Dimensionality

We follow Ethayarajh's (2019) measure of anisotropy as the average cosine similarity of random pairs of tokens. A score of 1 indicates high *an*isotropy, while a score of 0 indicates high isotropy. We measure anisotropy for each layer of each model. We sample 500k pairs of tokens from a set of Wikipedia sentences; sentences in the training data for our BaselineLM are excluded, but there could be overlap with train sets for other models.

In measuring the effect of rogue dimensions, Timkey and van Schijndel (2021) first calculated the contribution of each dimension to cosine similarity for each pair of tokens, enabling them to see how much each dimension contributed to a high anisotropy score; those that contributed a great deal were rogue. They then calculated the proportion of variance in cosine similarity of all dimensions that was accounted for by cosine similarity of all *except* those rogue dimensions. They reported r^2, the square of the Pearson correlation between cosine similarity of the tokens with all their dimensions and the cosine similarity of the tokens with the top k dimensions removed. If removing a dimension had very little impact, r^2 will be close to 1. The more a given dimension contributes to the variability in cosine similarity—that is, the more rogue it is—the closer to 0 r^2 will be when that dimension is removed. In other words, high values mean the dimensions are not very rogue. We refer to this measure as *rogue dimensionality*. Following Timkey and van Schijndel (2021), we report rogue dimensionality for $k \in \{1, 3, 5\}$.

3.4 Standardization

We also follow Timkey and van Schijndel's (2021) definition of standardization. To standardize an embedding $x \in \mathbb{R}^d$, we subtract the mean vector from the embedding and divide each dimension by its standard deviation: $z = (x - \mu)/\sigma$. See (Timkey and van Schijndel, 2021) equations 7-9 for details.

In the standardized condition, we apply this technique; in the not standardized condition, we do not. We use Spearman's ρ to measure how well each technique performs on the SimLex-999 task. We find the difference between Spearman's ρ in the two conditions:

$$\Delta\rho = \rho_{standardized} - \rho_{plain}$$

so that a positive $\Delta\rho$ indicates that standardization helped performance and negative $\Delta\rho$ means standardization harmed performance.

3.5 Mediation Analysis

Mediation analysis (Baron and Kenny, 1986) is a statistical technique to measure how much the ability of an independent variable X to predict a dependent variable Y is mediated by a third variable M, as illustrated in Figure 2. The analysis tells us how much X predicts M (path a), how much M predicts Y (path b), the *indirect effect*—how much X affects Y through M—the *direct effect*—how much X affects Y *without* going through M—and the *total effect*—the overall effect of X on Y through both direct and indirect pathways. If the indirect path is not significant or is negative, the mediator does not explain the effect of the independent variable on the dependent variable.

We use the Pingouin package's implementation of mediation analysis (Vallat, 2018), which uses a non-parametric bootstrap method. All reported mediation analyses average over layers. Reported coefficients are the unstandardized regression coefficients. We use a log-transformed count of tokens the model has seen as a continuous independent variable and $\Delta\rho$ as the dependent variable; we test whether differences in rogue dimensionality or anisotropy mediate the observed effect. For within-models analyses, we use *all* checkpoints

[6] https://huggingface.co/datasets/wikimedia/wikipedia Configuration: 20231101.en

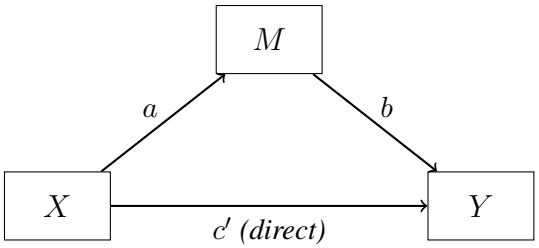


Figure 2: Mediation model. The indirect effect of X on Y through mediator M is $a \times b$; the total effect is $c = c' + ab$.

(not just the <10B and >100B ones) of the Pythia and Gemstone models.

4 Experiments

Our experiments seek to address three research questions:

1. Is the effect of standardization different for models trained on CP data vs. models trained on huge data?
2. Do rogue dimensions explain the observed differences?
3. Does anisotropy explain the observed differences?

4.1 RQ1: Effects of Standardization on CP-Data vs. Huge-Data Models

4.1.1 Across Models

Method Details. We first compare the effects of standardization across a set of different models trained on varying amounts of data. We use the BabyLM baseline models and our custom BaselineLM to represent models trained on CP data, and we use the fully trained final checkpoints of the Pythia and Gemstone models to represent models trained on huge data.

For each model, we calculate mean $\Delta\rho$ over all layers, resulting in 14 scores for the CP models and 4 for the huge models. To determine whether standardization helps, hurts, or is neutral for each group, we use a two-tailed t-test.

Results/Analysis. Results are summarized in Figure 3, which clearly shows that huge-data models benefit from standardization, but CP-data models are harmed by it. Standardization significantly *reduced* lexical similarity performance for the CP-data models (mean $\Delta\rho = -0.0158$, $p < 0.001$), indicating a modest but consistent effect. In contrast, standardization significantly *improved* lexical similarity performance for the four huge-data models (mean $\Delta\rho = 0.0369$, $p = 0.015$), despite the small number of models in this category.

We cannot rule out, based only on these experiments, the possibility that differences between models in the huge-data and CP-data groups accounts for the difference in effect of standardization. Our next experiment addresses this.

4.1.2 Within Models

Method Details. We ask whether, within the same model, the checkpoints trained on CP data are affected by standardization differently than the checkpoints trained on huge data. We evaluate each checkpoint of the Pythia and Gemstone models on SimLex-999 and calculate $\Delta\rho$ for each layer of each checkpoint. We group checkpoints trained on fewer than 10B tokens as the CP-data checkpoints and checkpoints trained on greater than 100B tokens as the huge-data checkpoints, to incorporate all scales that might be cognitively plausible and all scales that reflect how LLMs are typically trained in the other. We use a paired two-tailed t-test to determine whether mean $\Delta\rho$ over all layers of all included checkpoints for the CP-data checkpoints of each model differs significantly from the huge-data checkpoints. Given the small number of models ($n = 4$), we additionally verify using a mixed linear model regression model on all layers of *all* CP- and huge-data checkpoints with model name as a random effect.

Results/Analysis. In all four models, we observe that the early checkpoints are harmed by standardization, while the late checkpoints benefit from it. Figure 4 illustrates this pattern; graphs for the other models are in Appendix B. Average $\Delta\rho$ for the CP- and huge-data checkpoints is reported in Table 2. Conspicuously, the mean difference is negative for all models on CP-data checkpoints and positive for all models on huge-data checkpoints. These results are consistent with our observations on the across-model experiments. Despite the small n, the t-test indicates that the difference in average $\Delta\rho$ is significant ($t = 5.5980$, $p = 0.011$). The mixed-effects model findings further support this ($\text{coef} = 0.042$, $p < 0.001$).

We thus conclude that standardization of vectors harms performance for models trained on cognitively-plausible quantities of data and improves performance for models trained on huge data. Our next research questions investigate two possible mechanisms for this effect. Since standardization was proposed to mitigate the effects

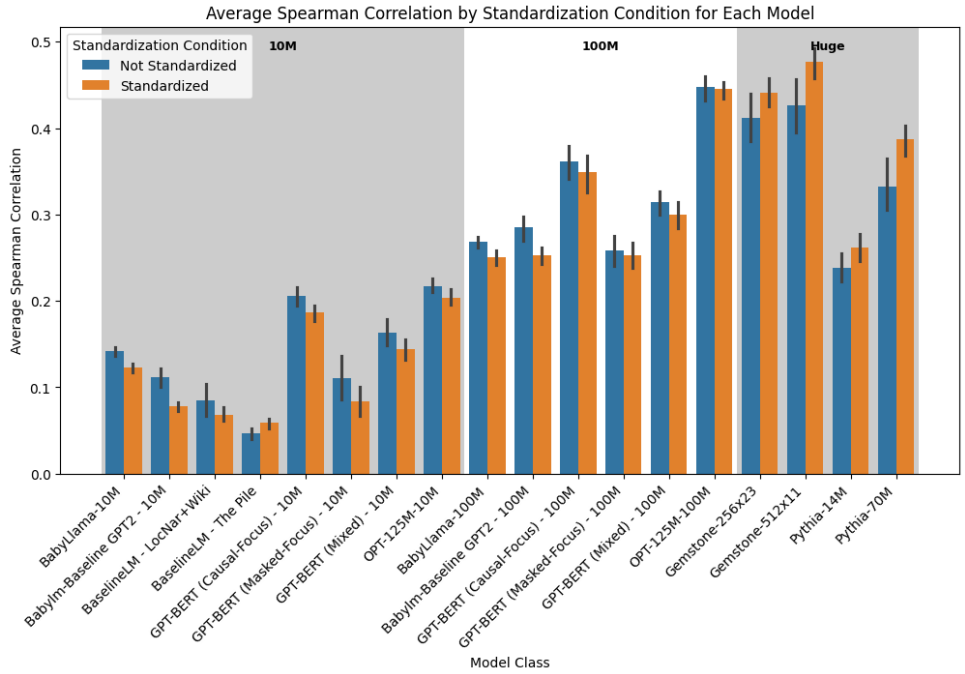


Figure 3: Models trained on 10M or 100M words performed better on a lexical similarity test without standardization (blue bars) than with it (orange bars). However, models trained on hundreds of billions of tokens benefited from standardization. Results averaged over layers for each model.

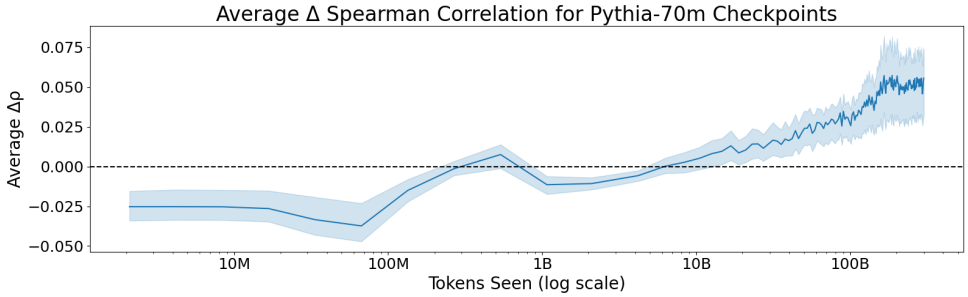


Figure 4: $\Delta\rho$, the difference between Spearman's ρ in the two standardization conditions, across number of tokens seen. Early in training, standardization harms lexical similarity performance for Pythia-70M. As the model sees more tokens, standardization becomes helpful.

Model	<10B tokens	>100B tokens
Gemstone-256x23	-0.0059	0.0232
Gemstone-512x11	-0.0002	0.0421
Pythia-14M	-0.0112	0.0219
Pythia-70M	-0.0148	0.0477

Table 2: Average $\Delta\rho$ for CP-data (<10B tokens) checkpoints was negative, while that for huge-data (>100B tokens) checkpoints was positive for all models.

of rogue dimensions, which in turn are a form of anisotropy, we investigate whether differences in rogue dimensionality or anisotropy more generally account for the observed differences in standardization.

4.2 RQ2: Does rogue dimensionality explain observed differences?

Method Details. To determine whether rogue dimensionality explains the standardization difference, we first investigate whether rogue dimensionality (operationalized as described in section 3.3) is *correlated* with greater benefits from standardization (positive $\Delta\rho$), using Spearman's ρ across models and across layers of those models. Since results from layers of the same model may not be independently distributed, we additionally perform a mixed linear model regression with model name as a random effect. Finally, we perform a mediation analysis as described in Section 3.5 to determine whether rogue dimensionality *mediates* the

correlation between data size and standardization helpfulness.

Results/Analysis. We observed a pronounced difference in rogue dimensionality at different data scales, with models trained on more data showing greater rogue dimensionality; see Appendix C for more information. Across all three values of k, Spearman's ρ between rogue dimensionality and $\Delta\rho$ was negative and significant (see Table 3). This suggests that greater rogue dimensionality (i.e., lower R^2) is associated with a greater benefit from standardization, as might be inferred from the original paper. When we apply a mixed linear model with model name as a random effect, however, the correlation is not significant at $k = 1$ ($p = 0.378$). Thus, if we focus on only one rogue dimension, we cannot conclude that rogue dimensionality is correlated with standardization benefit. The mixed linear model results do agree with the Spearman's results for $k \in \{3, 5\}$, leading to the conclusion that for these k values, rogue dimensionality is correlated with standardization benefits.

Metric	$k = 1$	$k = 3$	$k = 5$
Spearman	-0.293^{***}	-0.294^{***}	-0.307^{***}
MLM Coef.	-0.012	-0.042^{**}	-0.047^{***}

Table 3: Relationship between rogue dimensionality (R^2) and standardization benefit ($\Delta\rho$) across values of k. Lower R^2 indicates greater rogue dimensionality. Spearman correlations computed across all models and layers; mixed linear model with model name as random effect. $^{***}p < 0.001$, $^{**}p < 0.01$.

However, mediation analysis showed no significant mediation effect for rogue dimensionality ($p = 0.812$ for $k = 1$, $p = 0.972$ for $k = 3$, $p = 0.992$ for $k = 5$ in the across-model condition. Results in the within-model condition (Appendix C) likewise largely do not support a positive, significant mediation effect. We therefore conclude that **although rogue dimensionality is correlated with data size, it is *not* mediating the correlation between data size and standardization helpfulness**.

4.3 RQ3: Does anisotropy explain the observed differences?

Method Details. We begin with across-model analysis, checking if anisotropy is positively correlated with $\Delta\rho$ using Spearman's ρ across models and across layers of those models, then performing a mixed linear model regression using model name as a random effect to address the possibility that results from layers of the same model may not be independently distributed. We follow up with mediation analyses to determine whether anisotropy mediates the effect of data size on $\Delta\rho$.

Path	Coef.	p
size $\rightarrow$ anisotropy	0.0596	< 0.0001
anisotropy $\rightarrow \Delta\rho$	0.1713	0.0001
Total	0.0132	< 0.0001
Direct	0.0098	0.0107
Indirect	0.0034	0.3240

Table 4: Mediation analysis results show that train size is positively correlated with anisotropy, and anisotropy is positively correlated with $\Delta\rho$. Anisotropy does not significantly mediate the total effect.

Results/Analysis. We find a positive correlation between anisotropy and $\Delta\rho$ in the across-models setting using both methods: the Spearman correlation is $\rho = 0.296$ ($p < 0.0001$), and the mixed linear model regression yields a coefficient of 0.084 ($p < 0.001$).

Results of the across-model mediation analysis are summarized in Table 4. We again see a positive correlation between anisotropy and $\Delta\rho$. We observe a total effect of 0.0132. The indirect path is not significant ($p = 0.32$), so we cannot conclude that anisotropy mediates the relationship of training data size and $\Delta\rho$.

We next performed mediation analysis of the within-model checkpoints. As seen in Table 5, only Pythia-70M showed a significant positive mediation effect in the within-model experiments. In the Gemstone models, train size was negatively correlated with anisotropy, leading to a *negative* indirect effect in Gem-512 and no significant indirect effect in Gem-256. Thus, in the Gemstone models, anisotropy does not account for the observed effect.

5 Discussion

From our experiments, we see a qualitative difference between models trained on cognitively plausible data scales and models trained on huge data: the effect of standardization flips its sign. One might expect this to be due to differences in rogue dimensionality or anisotropy, as those increase with data scale;[7] that is, perhaps standardization is sim-

[7]The exception is Gemstone-512's anomalous negative correlation between data size and anisotropy in Table 5, which

Model	Path	Coef.	p
Pythia-14M	size → anisotropy	0.0165	< 0.0001
Pythia-14M	anisotropy → $\Delta\rho$	0.1957	< 0.0001
Pythia-14M	Total	0.0114	< 0.0001
Pythia-14M	Direct	0.0094	< 0.0001
Pythia-14M	Indirect	0.0021	0.0520
Pythia-70M	size → anisotropy	0.0678	< 0.0001
Pythia-70M	anisotropy → $\Delta\rho$	0.2112	< 0.0001
Pythia-70M	Total	0.0200	< 0.0001
Pythia-70M	Direct	0.0132	< 0.0001
Pythia-70M	Indirect	0.0068	< 0.0001
Gem-256	size → anisotropy	−0.0133	0.1388
Gem-256	anisotropy → $\Delta\rho$	−0.1107	< 0.0001
Gem-256	Total	0.0300	< 0.0001
Gem-256	Direct	0.0289	< 0.0001
Gem-256	Indirect	0.0011	0.5080
Gem-512	size → anisotropy	−0.0782	< 0.0001
Gem-512	anisotropy → $\Delta\rho$	−0.0483	< 0.0001
Gem-512	Total	0.0228	< 0.0001
Gem-512	Direct	0.0240	< 0.0001
Gem-512	Indirect	−0.0012	0.0160

Table 5: Within-model mediation analysis results. All models show a positive coefficient for total (path from train size to $\Delta\rho$). The indirect path for Gemstone-256 and Pythia-14M is not significant, so we cannot conclude that anisotropy mediates for those models. For Gemstone-512, the mediation effect is slightly *negative*. Pythia-70M shows a positive mediation effect.

ply unnecessary for CP models because they do not exhibit substantial rogue dimensionality in the first place. However, this interpretation is inconsistent with the mediation analyses. If the difference in rogue dimensionality between CP- and huge-scale models accounted for the difference in the standardization effect, the indirect (mediated) path would carry a significant, positive coefficient; it does not, with the single outlier exception of Pythia-14M noted in Appendix C. The same is true for anisotropy; only in Pythia-70M did anisotropy mediate positively and significantly. We additionally tested whether non-standardized (baseline) performance itself explains the reversal, using the same mediation framework as our anisotropy and rogue dimensionality analyses (log-transformed tokens seen as X, $\Delta\rho$ as Y). Baseline performance is positively correlated with training data scale ($a = 0.049$, $p < 0.001$) and, on its own, positively correlated with $\Delta\rho$ (coef. $= 0.073$, $p < 0.001$). However, when we control for training data scale, the relationship reverses: among models/checkpoints with comparable amounts of training data, higher baseline performance is associated with smaller standardization benefit ($b = -0.089$, $p < 0.001$, verified via an independent OLS regression). This produces a significant negative indirect effect ($a \times b = -0.0044$, $p < 0.001$), meaning baseline performance acts as a suppressor rather than an explanatory mediator here: its unique contribution once data scale is held fixed runs counter to the marginal relationship.[8] This lack of explanation suggests there may be other properties, geometric or otherwise, of the learned representations that we do not yet understand.

Future work should explore these and other differences between models trained on cognitively plausible data and models trained on huge data, with a special focus on differences of type rather than degree. Where previous work asks what properties emerge over the course of training that make LLMs so effective, we propose that understanding the differences from the opposite side is also important: what do these differences tell us about the best ways to develop new models for cognitively plausible data scales? Rather than borrowing and adjusting the techniques of huge scale data, an understanding of qualitative differences in the two can help identify CP-data-specific techniques.

Future work focused on the representational geometry of models trained on CP data should explore whether a similar reversal occurs when using other techniques to address anisotropy, such as whitening (Su et al., 2021). Using other methods of measuring anisotropy, like singular value decomposition (Razzhigaev et al., 2024) and IsoScore (Rudman et al., 2022), might provide further insights.

6 Conclusion

Standardization helps for models trained on over 100 B tokens, but hurts for models trained on fewer than 10 B tokens. In addition to the application to this particular task (lexical similarity), it raises the possibility that techniques developed for LLMs may not generalize well to models trained on cognitively plausible data. Our results show that this is unlikely to be simply a matter of models trained on less data having lower anisotropy or rogue dimensionality. By finding no support for these two leading candidates, we narrow the space of explanations future work needs to consider in seeking the mechanism of this effect.

warrants further investigation in future work.

[8] We verified this was not an artifact of collinearity between the two predictors ($r = 0.67$, $VIF = 1.84$, well below standard concern thresholds (O'Brien, 2007)).

Limitations

All models were transformer-based and at least partly causal language models. GPT-BERT models do additionally incorporate a masked language modeling objective, but the architectural diversity remains somewhat limited. This is, we think, a symptom of a larger lack of model diversity in the broader NLP community at present, and something we should collectively find ways to address.

We evaluated using only lexical similarity, and only one benchmark. While SimLex-999 is a well-respected, carefully designed evaluation set, results could differ using other benchmarks or other tasks. Future work should evaluate on additional datasets.

The difference between CP (<10B tokens) and huge (>100B tokens) data for the within-models paired two-tailed t-test in Section 4.1.2 is treated as a clean boundary, but these values are somewhat arbitrary. They were chosen to incorporate all scales that might be cognitively plausible in one group and all scales that reflect how LLMs are typically trained in the other and omit a middle section that does not clearly belong in either group. Moreover, the Gemstone checkpoints start at 2B tokens, much more than the 100M of the CP-scale models.

The across-model and within-model analyses have complementary confounds: across-model comparisons confound data scale with architecture, tokenizer, objective, and parameter count; checkpoint analyses hold these constant but are not equivalent to independently trained models. That the reversal appears under both designs is suggestive but not dispositive; fully isolating data scale would require training multiple, independently initialized models at both scales for each architecture, which we leave to future work.

Although one goal of CP-scale models is to benefit less-resourced languages, the present work looked only at English models. This allowed us to compare CP-scale models with huge scale models, but future work should verify that the findings extend to other languages.

Ethical Considerations

Data. We used existing datasets (the Pile, Wikipedia, WIT, and Localized Narratives) and benchmarks (SimLex-999). Web-based corpora are known to have issues of author consent and bias. We attempted to reduce the former by relying on the uncopyrighted subset of the Pile.

Training. Training LLMs demands significant computation, leading to use of electricity and potential harm to the environment. We attempt to mitigate this by using primarily models pretrained by other researchers, effectively amortizing the environmental cost by getting additional scientific findings from it. The two models we trained were limited to 10M words of input and 35M parameters, making them unusually small and reducing the environmental impact of training. Moreover, we hope that studying models trained on smaller data will lead to future improvements in training efficiency, for a net benefit.

Acknowledgments

We gratefully acknowledge helpful suggestions from Sean S. Trott. We thank the anonymous reviewers for their constructive feedback, which strengthened the paper. This work was supported in part by a midcourse leave from the University of Richmond. We used Claude and Claude Code to assist with implementation of analysis code, to suggest candidate additional analyses and relevant literature (all reviewed and verified by the authors), and to help polish the language. All AI-generated code, content, and suggestions were critically evaluated, edited, and verified by the authors, who take full responsibility for the final text and analysis.

References

Meta AI. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Reuben M Baron and David A Kenny. 1986. The Moderator-Mediator Variable Distinction in Social Psychological Research: Conceptual, Strategic, and Statistical Considerations. *Journal of personality and social psychology*, 51(6).

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. *Preprint*, arXiv:2304.01373.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the Third BabyLM Challenge: Accelerating Language Modeling Research with Cognitively Plausible Data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? *Preprint*, arXiv:2410.24159.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM Turns 4 and Goes Multilingual: Call for Papers for the 2026 BabyLM Workshop. *Preprint*, arXiv:2602.20092.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging Cross-lingual Structure in Pretrained Language Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Wikimedia Foundation. 2023. Wikimedia/wikipedia.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *Preprint*, arXiv:2101.00027.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 Herd of Models. *Preprint*, arXiv:2407.21783.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation. *Computational Linguistics*, 41(4):665–695.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, and 3 others. 2022. An empirical analysis of compute-optimal large language model training. *Advances in Neural Information Processing Systems*, 35:30016–30030.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the Second BabyLM Challenge: Sample-Efficient Pretraining on Developmentally Plausible Corpora. *Preprint*, arXiv:2412.05149.

Melody Li, Kumar Krishna Agrawal, Arna Ghosh, Komal Teru, Adam Santoro, Guillaume Lajoie, and Blake Richards. 2026. Tracing the Representation Geometry of Language Models from Pretraining to Post-training. *Advances in Neural Information Processing Systems*, 38:54691–54724.

Qianchu Liu, Diana McCarthy, Ivan Vulić, and Anna Korhonen. 2019. Investigating Cross-Lingual Alignment Methods for Contextualized Embeddings with Token-Level Evaluation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 33–43, Hong Kong, China. Association for Computational Linguistics.

Anemily Machina and Robert Mercer. 2024. Anisotropy is Not Inherent to Transformers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4892–4907, Mexico City, Mexico. Association for Computational Linguistics.

Yash Mahajan, Matthew Freestone, Naman Bansal, Sathyanarayanan N. Aakur, and Santu Karmaker. 2025. Revisiting Word Embeddings in the LLM Era. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 2686–2717, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.

Sean Michael McLeish, John Kirchenbauer, David Yu Miller, Siddharth Singh, Abhinav Bhatele, Micah Goldblum, Ashwinee Panda, and Tom Goldstein. 2025. Gemstones: A Model Suite for Multi-Faceted Scaling Laws. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Max Müller-Eberstein, Rob van der Goot, Barbara Plank, and Ivan Titov. 2023. Subspace Chronicles:

How Linguistic Information Emerges, Shifts and Interacts during Language Model Training. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13190–13208, Singapore. Association for Computational Linguistics.

Robert M. O'Brien. 2007. A Caution Regarding Rules of Thumb for Variance Inflation Factors. *Quality & Quantity*, 41(5):673–690.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, and 49 others. 2026. Olmo 3. *Preprint*, arXiv:2512.13961.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. 2020. Connecting Vision and Language with Localized Narratives. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V*, pages 647–664, Berlin, Heidelberg. Springer-Verlag.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Rajeev Ranjan. 2025. One Word Is Not Enough: Simple Prompts Improve Word Embeddings. *Preprint*, arXiv:2512.06744.

Vikas Raunak, Vaibhav Kumar, Vivek Gupta, and Florian Metze. 2020. On Dimensional Linguistic Properties of the Word Embedding Space. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 156–165, Online. Association for Computational Linguistics.

Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Ivan Oseledets, Denis Dimitrov, and Andrey Kuznetsov. 2024. The Shape of Learning: Anisotropy and Intrinsic Dimensions in Transformer-Based Models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 868–874, St. Julian's, Malta. Association for Computational Linguistics.

William Rudman and Carsten Eickhoff. 2024. Stable Anisotropic Regularization. *International Conference on Learning Representations*, 2024:23594–23609.

William Rudman, Nate Gillman, Taylor Rayne, and Carsten Eickhoff. 2022. IsoScore: Measuring the Uniformity of Embedding Space Utilization. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3325–3339, Dublin, Ireland. Association for Computational Linguistics.

Skipper Seabold and Josef Perktold. 2010. Statsmodels: Econometric and Statistical Modeling with Python. In *Python in Science Conference*, pages 92–96, Austin, Texas.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, and 17 others. 2024. Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788, Bangkok, Thailand. Association for Computational Linguistics.

Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. 2021. WIT: Wikipedia-based Image Text Dataset for Multimodal Multilingual Machine Learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2443–2449, Virtual Event Canada. ACM.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening Sentence Representations for Better Semantics and Faster Retrieval. *Preprint*, arXiv:2103.15316.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024. Gemma: Open Models Based on Gemini Research and Technology. *Preprint*, arXiv:2403.08295.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby Llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 279–289, Singapore. Association for Computational Linguistics.

William Timkey and Marten van Schijndel. 2021. All Bark and No Bite: Rogue Dimensions in Transformer Language Models Obscure Representational Quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Raphael Vallat. 2018. Pingouin: Statistics in Python. *Journal of Open Source Software*, 3(31):1026.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, and 15 others. 2020. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing Pretrained Language Models for Lexical Semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. 2019. Cross-Lingual BERT Transformation for Zero-Shot Dependency Parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5721–5727, Hong Kong, China. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM Challenge: Sample-Efficient Pretraining on Developmentally Plausible Corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent Abilities of Large Language Models. *Transactions on Machine Learning Research*.

Lechao Xiao. 2025. Rethinking Conventional Wisdom in Machine Learning: From Generalization to Scaling. *Preprint*, arXiv:2409.15156.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. *Preprint*, arXiv:2205.01068.

A Isolated vs. Contextualized Word Vectors

In preliminary experiments, we compared performance of Opt-125M, BabyLlama, Pythia-14M, Pythia-70M, and the baseline LM in the isolated and contextualized conditions. In the isolated condition, each word was passed through the model with no context. In the contextualized condition, ten sentences from Wikipedia containing the word were passed through the model; the word was represented using the average of its embeddings in those ten contexts. Averaged over all models and all layers, contextualized embeddings significantly outperformed isolated embeddings on SimLex-999 (mean Spearman's ρ 0.231 vs. -0.005; paired t-test t=18.61, p<0.001). We therefore used contextualized vectors for all subsequent experiments.

B Difference in Spearman's ρ Across Checkpoints

As described in Section 4.1.2 above, within each of the Pythia and Gemstone models, early checkpoints were harmed by standardization (negative $\Delta\rho$), and later checkpoints were helped (positive $\Delta\rho$). Results over all layers, all checkpoints, are shown in Figure 5.

C Additional Rogue Dimensionality Results

C.1 Distribution of Rogue Dimensionality Results

We observed that models trained on huge data were more impacted by rogue dimensionality than models trained on 100M-word data, which in turn were more impacted than the 10M-word models (Table 6). The distribution was also quite different, as illustrated in Figure 6. Models trained on strict-small data universally had almost no rogue dimensionality, as demonstrated by R^2 close to 1, while models trained on strict-scale or huge data had greater variance.

Data Scale	$k=1$	$k=3$	$k=5$
10M	0.999	0.996	0.994
100M	0.802	0.738	0.717
Huge	0.551	0.390	0.374

Table 6: Mean R^2 by data scale and k across models. Lower R^2 indicates greater rogue dimensionality.

C.2 Within-Model Mediation Analyses of Rogue Dimensionality

We applied mediation analysis to the within-model condition to further study whether the difference in rogue dimensionality in models trained on CP versus huge data accounts for the difference in standardization effect. Consistent with our observations in the across-model mediation analysis (Section 4.2), our results do not support a positive mediation by rogue dimensionality.

As shown in Table 7, most of these analyses showed no significant indirect effect (that is, no sig-

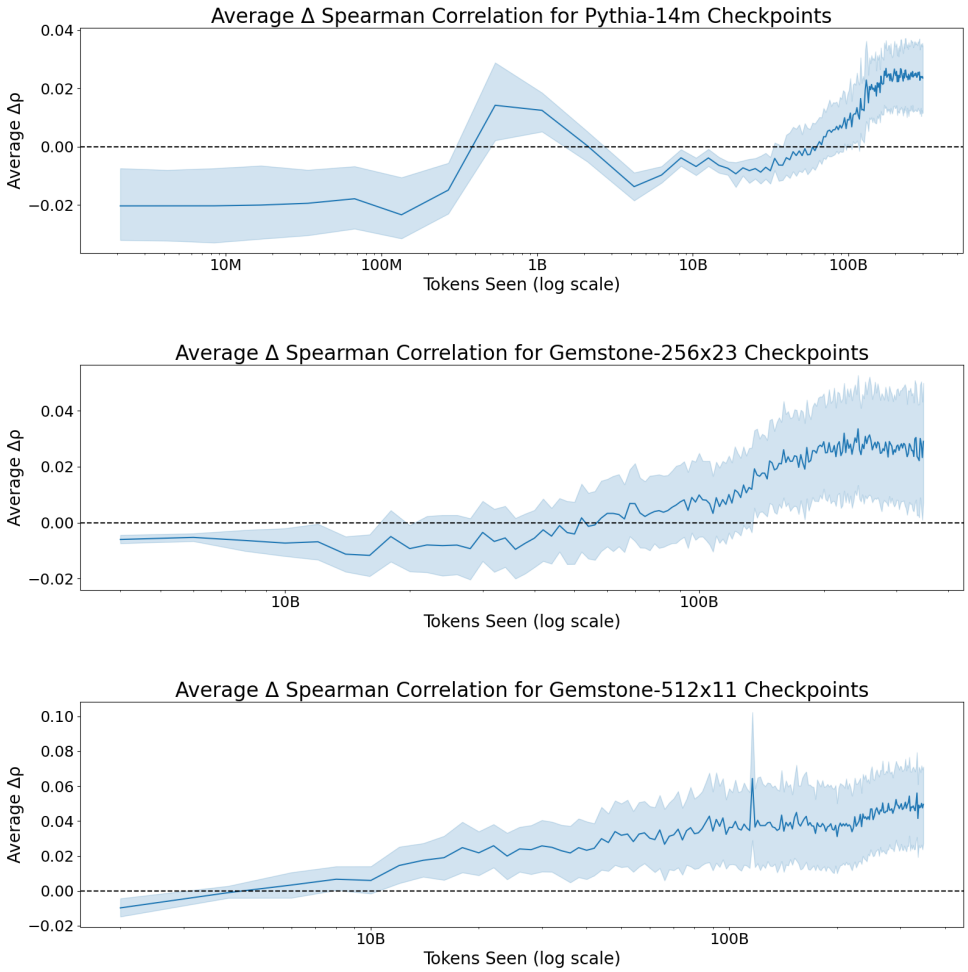


Figure 5: $\Delta\rho$, the difference between Spearman's ρ in the two standardization conditions, for Pythia-14M and the two Gemstone models. As observed in Pythia-70M (Figure 4), early in training, standardization harms lexical similarity performance. As each model sees more tokens, standardization becomes helpful.

nificant mediation). Among those with significant mediation, all but one showed a *negative* mediation: that is, the path mediated by rogue dimensionality actually *reduced* the total effect of data size on $\Delta\rho$. We interpret the one positive mediation among the twelve analyses as an outlier.

D Statistical Packages Used

- SciPy (Virtanen et al., 2020)
- Statsmodels (Seabold and Perktold, 2010)
- Pingouin (Vallat, 2018)

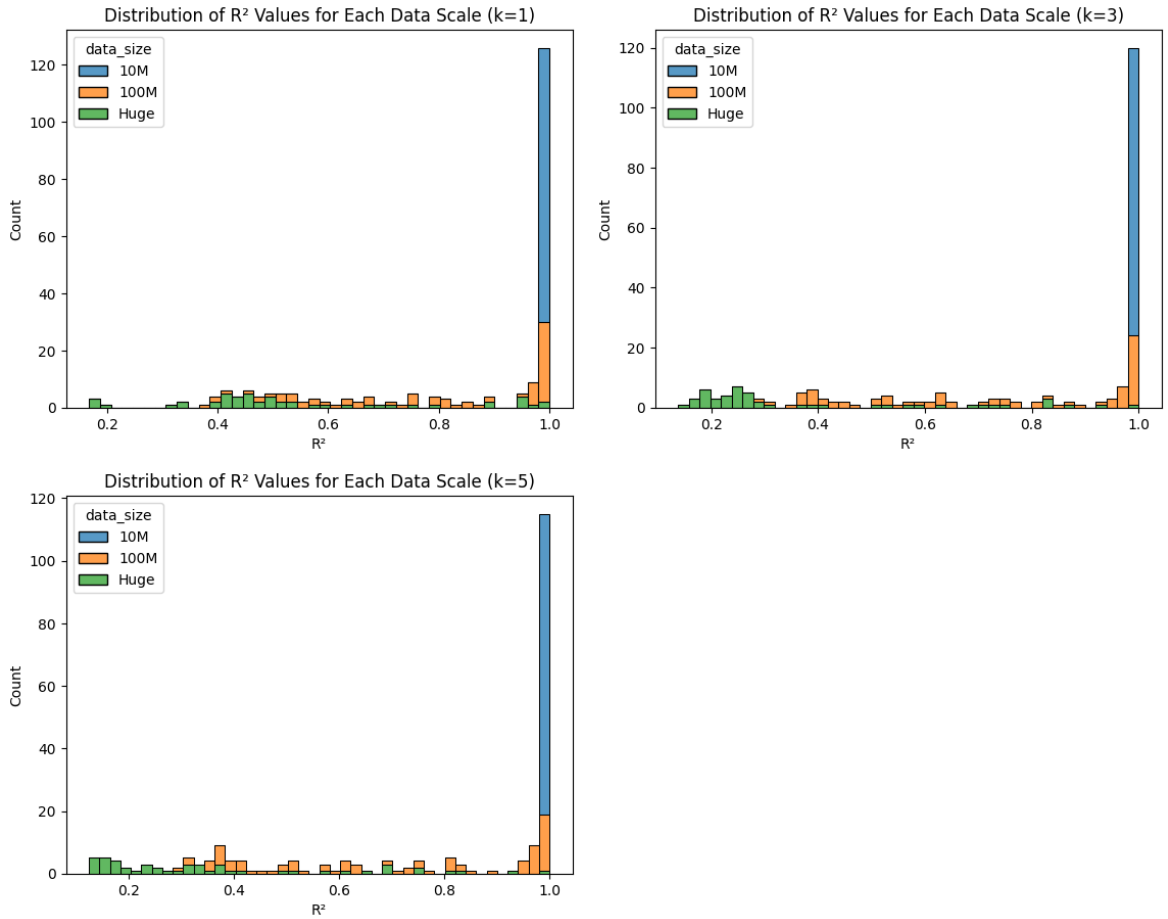


Figure 6: Distributions of rogue dimensionality results for all layers of all fully-trained models for $k \in \{1, 3, 5\}$. High R^2 indicates low rogue dimensionality. All layers of all models trained on 10M words had the lowest possible rogue dimensionality. Models trained on 100M words had less rogue dimensionality than huge models, though both showed a wider distribution.

Model	k	Total Coeff.	Indirect Coeff.	% Mediated	p-value
Gemstone-256x23	k=1	0.0300	-0.0108	-35.9665	<0.0001
Gemstone-512x11	k=1	0.0228	0.0007	–	0.0720
Pythia-14M	k=1	0.0114	0.0034	29.7397	0.0400
Pythia-70M	k=1	0.0200	-0.0031	-15.6364	<0.0001
Gemstone-256x23	k=3	0.0300	-0.0020	–	0.4560
Gemstone-512x11	k=3	0.0228	0.0011	–	0.1320
Pythia-14M	k=3	0.0114	0.0018	–	0.2200
Pythia-70M	k=3	0.0200	-0.0035	-17.2936	<0.0001
Gemstone-256x23	k=5	0.0300	-0.0031	-10.4552	0.0360
Gemstone-512x11	k=5	0.0228	0.0006	–	0.1800
Pythia-14M	k=5	0.0114	0.0006	–	0.4440
Pythia-70M	k=5	0.0200	-0.0035	-17.6214	<0.0001

Table 7: Mediation analysis results from within-model experiments. Total reflects the effect of data size on $\Delta\rho$; indirect reflects the effect mediated by rogue dimensionality.

Compact Multilingual Pre-training under the BabyLM 100M-Word Budget with Regex-Guided Tokenization and Attention Residuals

Yihan Liu
Lund University
yi7510li-s@student.lu.se

Tongjing Zhang
Lund University
tongjing.zhang.1246@student.lu.se

Pierre Nugues
Lund University
pierre.nugues@cs.lth.se

Abstract

Multilingual pre-training under severe data constraints can suffer from limited representational capacity and uneven tokenization across languages. Addressing the BabyLM 2026 Multilingual Track, we study a compact pre-training recipe under the 100M-word budget. Our main contribution is a Calibrated Regex-Guided Byte-Level BPE (Regex-BBPE) tokenizer that combines script-aware pre-tokenization with byte-premium-adjusted tokenizer training in a 16K vocabulary. We pair this tokenizer with the previously proposed Attention Residuals mechanism and the Muon optimizer, and evaluate three English/Dutch/Chinese mixing ratios. Among the tested mixtures, the Chinese-dominant configuration (20% English, 20% Dutch, 60% Chinese) gives the strongest aggregate fine-tuning results. Zero-shot results are mixed and remain slightly below the official GPT-2 trilingual baseline on average, while fine-tuning improves the language-level averages by 8.55, 9.55, and 10.09 accuracy points for English, Dutch, and Chinese, respectively. The mixing-ratio experiments also show different sensitivity patterns for the two West Germanic languages and Chinese; we treat this as an observation within the tested language triple rather than a general cross-lingual law. Our code is publicly available at [code].

1 Introduction

Modern large language models (LLMs) achieve impressive linguistic competence through training on trillions of tokens. This scale stands in stark contrast to human language acquisition, which yields proficiency from only 10–100 million words (Warstadt et al., 2023). The BabyLM 2026 Multilingual Track (Choshen et al., 2026a), with texts in English, Dutch, and Chinese, addresses this gap by promoting sample-efficient pre-training under developmentally plausible constraints. However, low-data multilingual pre-training introduces severe bottlenecks: the *Curse of Multilinguality* (Conneau et al., 2020), which triggers negative transfer or catastrophic forgetting (French, 1999) within restricted parameter spaces, and *tokenization pathologies*, where standard frequency-based algorithms, like byte-pair encoding (BPE), over-segment non-Latin scripts, fracturing semantic units and bloating the token budget (Rust et al., 2021, Ahia et al., 2023).

This paper addresses these challenges through a compact recipe spanning tokenization, architecture, and data mixture. Our main methodological contribution is *Calibrated Regex-guided Byte-level BPE* (Regex-BBPE). Script-aware pre-tokenization itself is not new; our contribution is to combine it with byte-premium-adjusted tokenizer training under a strict 16K vocabulary. We compare this design with SentencePiece Unigram (Kudo, 2018, Kudo and Richardson, 2018) and the larger-vocabulary tiktoken tokenizer (OpenAI, 2023).

Second, we evaluate the recently proposed *Attention Residuals* (AttnRes) mechanism (Kimi Team et al., 2026) in this compact multilingual setting. AttnRes uses learned, input-dependent weighting over preceding layer outputs rather than only uniform residual accumulation. We pair the backbone with the previously proposed *Muon* optimizer (Jordan et al., 2024) and report the training setup explicitly in Section 4.1.

Finally, we study the **language mixing ratio** between English, Dutch, and Chinese as a hyperparameter. Across balanced (V1: 33/33/33), ZH-dominant (V2: 20/20/60), and EN-dominant (V3: 50/25/25) mixtures, V2 provides the best Chi-

Proceedings of the Second BabyLM Workshop, pages 403–412
October 29, 2026 ©2026 Association for Computational Linguistics

nese and Dutch fine-tuning averages, while V3 is marginally best for English. The zero-shot and fine-tuning trends suggest different sensitivity to data allocation across the three languages. Because English and Dutch are both West Germanic and only three ratios are tested, we interpret this pattern conservatively and do not claim a general Indo-European "syntax–semantics dissociation."

Our contributions are fourfold:

1. We propose **Calibrated Regex-BBPE**, combining script-aware pre-tokenization, byte-premium-adjusted tokenizer training, and byte-level fallback within a 16K vocabulary.

2. We evaluate **Attention Residuals** (Kimi Team et al., 2026) in a compact multilingual BabyLM setting and provide a controlled ablation against standard residual connections.

3. We report an **efficient low-data pre-training recipe** using the Muon optimizer and a compact LLaMA-style backbone.

4. We conduct a **language-mixing ablation** over three EN/NL/ZH ratios and characterize the resulting task- and language-specific trade-offs.

2 Related Work

2.1 Multilingual Tokenization Bottlenecks

Tokenization is a critical bottleneck in data-constrained multilingual pre-training. Standard frequency-based byte-pair encoding (BPE) (Sennrich et al., 2016) exhibits two severe pathologies: *cross-lingual token pollution*, where single merges span morpheme boundaries of distinct scripts, and *over-segmentation of low-resource languages*, where infrequent scripts consume a disproportionate token volume per semantic unit (Rust et al., 2021, Ahia et al., 2023). Under the BabyLM 100M-word limit, such over-segmentation severely shrinks the effective learning signal.

Regex-based pre-tokenization (Radford et al., 2019) can enforce script boundaries, but it does not by itself determine how a compact multilingual vocabulary is allocated. Both BPE and Unigram-style tokenization are frequency-driven, and their multilingual behavior depends on the training distribution and vocabulary budget. SentencePiece provides implementations of these approaches (Kudo and Richardson, 2018); for the Unigram algorithm we follow Kudo (2018). We use language-specific byte premiums (Arnett et al., 2024) to adjust the tokenizer-training mixture and examine whether this improves compression balance under a 16K vocabulary.

2.2 Compact Transformer Architectures

Recent small language models largely converge on the LLaMA paradigm (Touvron et al., 2023), which enhances standard Transformers (Vaswani et al., 2017) via Rotary Position Embeddings (RoPE) (Su et al., 2024), SwiGLU activations (Shazeer, 2020), and Pre-RMSNorm (Zhang and Sennrich, 2019). While these components stabilize training, directly downscaling LLaMA to strict parameter constraints (e.g., under 100M parameters) exacerbates representational bottlenecks. In such severely limited regimes, the traditional uniform residual connections impose a structural rigidity that restricts the network's capacity to dynamically route information across varying depths, leading to severe representation dilution. This highlights the critical need for deeper architectural interventions to maximize representation utilization at a small scale.

2.3 Attention Residuals

Standard residual connections (Vaswani et al., 2017) employ a uniformly-weighted recurrence, $h_l = h_{l-1} + f_{l-1}(h_{l-1})$. While this unit-weight accumulation stabilizes deep training, it imposes an inflexible depth-wise aggregation. Every layer receives an identical sum of prior outputs with no mechanism to selectively weight individual layer contributions, causing uncontrolled hidden-state growth and "PreNorm dilution" (Kimi Team et al., 2026). To address this, Kimi Team et al. (2026) introduce *Attention Residuals* (AttnRes), replacing standard residuals with a softmax attention mechanism over preceding layer outputs. Each layer maintains a learnable pseudo-query to compute input-dependent weights, enabling dynamic retrieval of prior representations. Validated on the Kimi architecture at a trillion-token scale, AttnRes achieved a $1.25\times$ compute advantage with negligible parameter overhead.

AttnRes was introduced and validated at much larger scale than the setting studied here. Its behavior in compact multilingual models is therefore comparatively underexplored. We evaluate whether its depth-wise routing remains useful under the BabyLM 100M-word constraint, where both data and model capacity are limited.

2.4 Optimization in Low-Data Regimes

While AdamW (Loshchilov and Hutter, 2019) is the standard for LLM pre-training, maximizing step-wise sample efficiency is paramount under strict data constraints. Structurally constrained updates, particularly orthogonal optimization, can significantly accelerate early convergence by preserving the spectral properties of weight matrices (Anil et al., 2020, Zhao et al., 2024). Based on this, the Muon optimizer (Jordan et al., 2024) orthogonalizes update momentum via Newton-Schulz iterations. This maximizes spectral utilization without the prohibitive memory overhead typical of full second-order methods.

Although Muon accelerates standard large-scale pre-training, its efficacy in severely constrained, multilingual regimes remains unexplored. In the BabyLM setting, where models must rapidly align disparate linguistic representations within a rigid 100M-word budget, the steep loss reduction promised by Newton-Schulz orthogonalization presents a compelling avenue for maximizing low-data sample efficiency.

3 Methodology

Our framework for the data-constrained 100M-word regime rests on three pillars: a byte-premium-calibrated tokenization pipeline, a compact LLaMA backbone augmented with Attention Residuals, and an efficient orthogonal optimization recipe.

3.1 Data Composition and Tokenization Pipeline

Our training corpus is sampled from the Baby-BabelLM repositories (Jumelet et al., 2025) spanning English (`babylm-eng`), Dutch (`babylm-nld`), and Chinese (`babylm-zho`). Following the challenge setup, the language quotas used to construct tokenizer-training text are adjusted using the reported byte-premium values (Arnett et al., 2024): 1.0000 (EN), 1.0516 (NL), and 0.9894 (ZH).

Regex-BBPE procedure. The tokenizer is trained in three steps. First, we construct the multilingual tokenizer-training text using the byte-premium-adjusted language quotas. Second, a regex pre-tokenizer separates Latin/alphanumeric spans, Han characters, punctuation, and whitespace so that BPE merges are learned only within compatible pre-tokenized spans. Third, standard byte-level BPE merges are learned until the 16K vocabulary budget is reached, with byte-level fallback guaranteeing coverage. The script split itself is a standard technique; our contribution is its combination with byte-premium calibration under the compact vocabulary budget. We do not apply a separate post-hoc "high-frequency-only" vocabulary filter.

The resulting tokenizer is used to process the training corpus, which is packed into a single binary sequence of exactly 100M tokens. For V1, the token budget is distributed equally (33.3% per language); V2 and V3 modify only this language mixture for the ablation in Section 5.2.

3.2 LLaMA-based Backbone

Our structural foundation is a compact, decoder-only LLaMA-style Transformer (Touvron et al., 2023) with approximately 42M parameters. It uses 8 layers, $d_{\text{model}} = 512$, 8 attention heads, 8 KV heads, a 2048-dimensional FFN, a 16K vocabulary, and a maximum sequence length of 512, together with RMSNorm, SwiGLU, and RoPE. These choices define our submitted backbone; the official GPT-2 system is used as the BabyLM task baseline rather than as an architecture-matched control.

3.3 Mitigating Representation Dilution via Attention Residuals

We incorporate the previously proposed *Attention Residuals* (AttnRes) mechanism (Kimi Team et al., 2026) into the compact backbone. In our implementation, each layer has access to a pool of earlier hidden states and uses a learnable pseudo-query to obtain a softmax weighting over those states. The resulting depth-wise mixture is supplied to the block while the usual within-block residual path is retained. This follows the AttnRes idea of replacing fixed depth-wise accumulation with learned retrieval; we evaluate its contribution directly through the ablation in Section 5.1.

4 Experimental Setup

4.1 Model and Training Configurations

All models are trained from scratch for 10 epochs using `bfloat16` mixed precision with PyTorch on an NVIDIA 5090 GPU. The compact LLaMA backbone is configured with a 16,000 vocabulary size, $d_{\text{model}} = 512$, 8 layers, 8 attention heads, 8 KV heads, a 2048 FFN intermediate dimension, and a maximum sequence length of 512. Regulariza-

Task	English		Dutch		Chinese	
	Ours	Baseline	Ours	Baseline	Ours	Baseline
BLiMP	64.55	**69.18**	74.99	**79.27**	75.34	**75.44**
MultiBLiMP	78.70	**87.14**	87.73	**91.85**	—	—
HellaSwag	26.48	**26.66**	**26.47**	26.37	26.98	**27.05**
Winogrande	**50.37**	48.80	**49.46**	49.30	**50.62**	49.71
XStoryCloze	48.30	**49.92**	**48.22**	47.37	48.92	**50.62**
XCOMPS	—	—	51.93	**52.68**	**54.01**	53.90
Avg.	53.68	**56.34**	56.47	**57.81**	51.17	**51.34**

Table 1: Zero-shot evaluation of our LLaMA + Attention Residuals model (with ZH-dominant (20/20/60)) versus the official GPT-2 `en_nld_zho_equal` trilingual baseline (Choshen et al., 2026a). Bold indicates per-cell winner.

Task	English		Dutch		Chinese	
	Ours	Baseline	Ours	Baseline	Ours	Baseline
ARC	**25.62**	24.79	**26.25**	24.38	24.17	**25.00**
Belebele	**23.86**	22.16	23.86	**30.11**	18.75	**22.73**
BMLAMA	**18.63**	9.35	**14.57**	10.68	**14.16**	9.35
MNLI	50.68	**51.63**	**52.76**	43.92	**54.90**	45.83
SIB-200	**66.00**	21.50	**61.50**	21.50	**78.00**	21.50
TruthfulQA	**22.32**	**22.32**	**32.14**	23.21	**25.89**	23.21
XNLI	**51.15**	46.65	—	—	**45.35**	43.45
Avg.	**36.89**	28.34	**35.18**	25.63	**37.32**	27.23

Table 2: Finetuning accuracy of our main V2 (ZH-dominant, 20/20/60) LLaMA + Attention Residuals model versus the official GPT-2 `en_nld_zho_equal` trilingual baseline. Bold indicates per-cell winner (ties are both bolded).

tion utilizes a 0.05 dropout probability and an RMSNorm epsilon of 1×10^{-5}.

We use the Muon optimizer (Liu et al., 2025) for hidden two-dimensional weight matrices; parameters excluded from Muon, including embeddings and normalization/scalar parameters, are optimized with AdamW. Muon orthogonalizes momentum updates using Newton–Schulz iterations. Following a progressive hyperparameter search, the final recipe uses a base learning rate of 3.36×10^{-4}, a Muon learning rate of 2.38×10^{-4}, and weight decay of 0.0867. Training cross-entropy converges to 2.3–2.5. We did not run a matched Muon-vs.-AdamW ablation, so the optimizer's isolated contribution should not be inferred from the end-to-end comparison.

4.2 Evaluation Protocol

We use the official BabyLM 2026 Multilingual Track evaluation suite and the official `en_nld_zho_equal` GPT-2 baseline (Choshen et al., 2026b). The task set and baseline were provided by the organizers rather than selected by us. The same official task-specific fine-tuning pipeline is applied to both our model and the baseline.

- **Zero-shot evaluation**: BLiMP (Warstadt et al., 2020), MultiBLiMP (Jumelet et al., 2026), ZhoBLiMP, HellaSwag, Winogrande, XStoryCloze, and XCOMPS, with no task-specific fine-tuning.
- **Fine-tuning evaluation**: ARC, Belebele, BMLAMA, MNLI, SIB-200, TruthfulQA, and XNLI under the official task-adaptation pipeline.

All reported languages (EN/NL/ZH) are present during pre-training; therefore, "zero-shot" here is task-level zero-shot and does not demonstrate transfer to an unseen language.

4.3 Results and Analysis

We compare the ZH-dominant V2 configuration (20/20/60), selected from the three tested mixtures, with the official trilingual GPT-2 baseline (`en_nld_zho_equal`) across the two evaluation regimes.

4.3.1 Zero-Shot Evaluation

As shown in Table 1, the zero-shot results are mixed. V2 improves over the baseline on Winogrande in all three languages and on a small number of other cells, including Dutch HellaSwag/XStoryCloze and Chinese XCOMPS. However, its average zero-shot scores remain below the baseline for English (53.68 vs. 56.34), Dutch

(56.47 vs. 57.81), and Chinese (51.17 vs. 51.34). ZhoBLiMP is nearly tied (75.34 vs. 75.44). We therefore treat V2 as competitive on selected zero-shot tasks rather than as uniformly better than the baseline.

4.3.2 Fine-Tuning Evaluation

Fine-tuning results (Table 2) are stronger. Relative to the official baseline under the same fine-tuning pipeline, V2 improves the language-level averages from 28.34 to 36.89 in English (+8.55 points), 25.63 to 35.18 in Dutch (+9.55), and 27.23 to 37.32 in Chinese (+10.09). The largest margins occur on SIB-200 and BMLAMA, while the baseline remains better on several individual tasks such as Dutch Belebele. Because the systems differ in multiple components, these end-to-end differences should not be attributed to a single architectural or optimization choice.

5 Experiments

To isolate architectural and tokenization effects from data-skew biases, we sequentially ablate components under the balanced V1 (33/33/33) ratio before optimizing the language mixing ratio.

5.1 Ablation of Attention Residuals

Tables 3 and 4 isolate the effect of AttnRes under V1. Removing AttnRes reduces the zero-shot averages from 54.44 to 50.40 (EN), 54.19 to 52.40 (NL), and 49.05 to 47.45 (ZH). The fine-tuning averages fall from 35.24 to 29.30, 33.30 to 25.60, and 36.62 to 27.67, respectively. These results support a positive contribution from AttnRes in this configuration, although individual tasks do not improve uniformly.

5.2 Language Mixing Ratio Ablation

The BabyLM budget leaves the per-language token allocation open. To determine the optimal distribution, we ablate three configurations under an identical backbone and optimization recipe: balanced (V1: 33/33/33 for EN/NL/ZH), ZH-dominant (V2: 20/20/60), and EN-dominant (V3: 50/25/25).

Tables 5 and 6 show that language allocation affects tasks differently. Reducing English from 33% in V1 to 20% in V2 changes English BLiMP only slightly (64.41 to 64.55), while Dutch BLiMP rises from 70.90 to 74.99. Increasing Chinese from 33% to 60% raises ZhoBLiMP from 67.46 to 75.34. In fine-tuning, V2 has the highest average for Dutch and Chinese, whereas V3 is marginally highest for English.

We use "syntax–semantics dissociation" only as a descriptive shorthand for this pattern within the tested EN/NL/ZH setting. English and Dutch are both West Germanic, so the experiment does not justify a general claim about Indo-European languages. Shared structure between English and Dutch is one plausible explanation for the relative stability of their grammatical scores, but we do not isolate that mechanism. Chinese sensitivity also cannot be explained solely by tokenizer fragmentation: Regex-BBPE has the best Chinese compression among the three tested tokenizers (0.81 chars/token), yet Chinese scores still vary strongly with the data mixture.

Among the three tested mixtures, V2 provides the strongest aggregate result for the ZH-focused setting, but denser sampling of the mixture space and more language families would be required to establish a general allocation rule.

5.3 Comparison of Tokenization Paradigms

We evaluate our Regex-BBPE (16K vocabulary) against SentencePiece Unigram (16K) and the BPE-based tiktoken cl100k_base (100K) using the LLaMA backbone under the balanced V1 ratio.

Table 7 shows a clear trade-off: tiktoken provides the best English compression, whereas Regex-BBPE is strongest on Dutch and Chinese. On Chinese, tiktoken reaches 0.52 chars/token versus 0.81 for Regex-BBPE, a 35% relative decrease. The downstream tables show corresponding differences across tokenizers, but these comparisons also change embedding-vocabulary size (16K vs. 100K), so we treat vocabulary allocation, sequence fragmentation, and parameter allocation as plausible contributing factors rather than isolated causes.

5.4 Comparison of LLM Backbones

We isolate architectural impacts by comparing a Baby-Qwen prototype against our LLaMA backbone under the balanced V1 ratio, with both variants incorporating AttnRes and Muon optimization.

As detailed in Tables 10 and 11, the LLaMA variant gives the higher English zero-shot average and slightly higher fine-tuning averages in all three languages, while results are mixed across individual tasks. Because both backbone variants use AttnRes and Muon, this comparison informs our backbone choice but does not isolate the contribution of AttnRes; that effect is evaluated separately in Sec-

Task	English			Dutch			Chinese		
	Ours	**w/o AttnRes**	**BL**	**Ours**	**w/o AttnRes**	**BL**	**Ours**	**w/o AttnRes**	**BL**
BLiMP	**64.41**	56.20	69.18*	**70.90**	64.79	79.27*	**67.46**	61.14	75.44*
MultiBLiMP	**82.73**	71.82	87.14*	**78.40**	77.52	91.85*	—	—	—
HellaSwag	**26.40**	26.21	26.66*	**26.06**	25.80	26.37*	**26.75**	26.19	27.05*
Winogrande	49.30	**49.71***	48.80	**50.56***	47.82	49.30	**50.10***	49.38	49.71
XStoryCloze	**49.38**	48.07	49.92*	**47.70***	47.14	47.37	47.99	**48.30**	50.62*
XCOMPS	—	—	—	**51.54**	51.32	52.68*	**52.94**	52.23	53.90*
Avg.	**54.44**	50.40	56.34*	**54.19**	52.40	57.81*	**49.05**	47.45	51.34*

Table 3: AttnRes ablation study. **Ours** is compared to the ablated version (**w/o AttnRes**) and the GPT-2 baseline (**BL**). **Bold** indicates the better score between our variants; asterisk (*) marks the global maximum.

Task	English			Dutch			Chinese		
	Ours	**w/o AttnRes**	**BL**	**Ours**	**w/o AttnRes**	**BL**	**Ours**	**w/o AttnRes**	**BL**
ARC	23.54	**24.50**	24.79*	**26.25***	24.50	24.38	**26.67***	26.00	25.00
Belebele	21.02	**21.80**	22.16*	24.43	**27.50**	30.11*	20.45	**22.50**	22.73*
BMLAMA	**12.91***	9.80	9.35	**19.54***	12.50	10.68	**13.33***	9.50	9.35
MNLI	47.47	**50.00**	51.63*	**45.21***	44.10	43.92	**48.70***	46.20	45.83
SIB-200	**77.00***	29.50	21.50	**66.50***	22.00	21.50	**80.50***	23.00	21.50
TruthfulQA	17.86	**22.80***	22.32	17.86	23.00	23.21*	**24.11***	23.50	23.21
XNLI	**46.85***	46.70	46.65	—	—	—	42.60	**43.00**	43.45*
Avg.	**35.24***	29.30	28.34	**33.30***	25.60	25.63	**36.62***	27.67	27.23

Table 4: AttnRes finetuning ablation. **Ours** is compared to the ablated version (**w/o AttnRes**) and the GPT-2 baseline (**BL**). **Bold** indicates the better score between our variants; asterisk (*) marks the global maximum.

Task	English				Dutch				Chinese			
	V1	**V2**	**V3**	**BL**	**V1**	**V2**	**V3**	**BL**	**V1**	**V2**	**V3**	**BL**
BLiMP	64.41	64.55	**65.06**	69.18*	70.90	**74.99**	72.67	79.27*	67.46	**75.34**	74.18	75.44*
MultiBLiMP	**82.73**	78.70	82.34	87.14*	78.40	87.73	**88.03**	91.85*	—	—	—	—
HellaSwag	26.40	**26.48**	26.46	26.66*	26.06	26.47	**26.60***	26.37	26.75	**26.98**	26.76	27.05*
Winogrande	49.30	**50.37***	50.21	48.80	**50.56***	49.46	49.46	49.30	50.10	**50.62***	48.80	49.71
XStoryCloze	**49.38**	48.30	48.76	49.92*	47.70	48.22	**48.53***	47.37	47.99	**48.92**	47.21	50.62*
XCOMPS	—	—	—	—	51.54	51.93	**52.16**	52.68*	52.94	**54.01***	53.24	53.90
Avg.	54.44	53.68	**54.57**	56.34*	54.19	**56.47**	56.24	57.81*	49.05	**51.17**	50.04	51.34*

Table 5: Zero-shot evaluation of mixing ratios (V1/V2/V3) vs. the GPT-2 baseline (**BL**). **Bold** marks the best among our variants; asterisk (*) denotes the global maximum.

Task	English				Dutch				Chinese			
	V1	**V2**	**V3**	**BL**	**V1**	**V2**	**V3**	**BL**	**V1**	**V2**	**V3**	**BL**
ARC	23.54	**25.62***	23.33	24.79	26.25	26.25	**27.08***	24.38	**26.67***	24.17	24.79	25.00
Belebele	21.02	**23.86***	**23.86***	22.16	**24.43**	23.86	19.32	30.11*	20.45	18.75	**21.59**	22.73*
BMLAMA	12.91	**18.63***	15.73	9.35	**19.54***	14.57	16.14	10.68	13.33	14.16	**14.24***	9.35
MNLI	47.47	**50.68**	48.48	51.63*	45.21	**52.76***	45.50	43.92	48.70	**54.90***	49.38	45.83
SIB-200	**77.00***	66.00	72.50	21.50	**66.50***	61.50	66.00	21.50	**80.50***	78.00	75.00	21.50
TruthfulQA	17.86	22.32	**27.68***	22.32	17.86	**32.14***	23.21	23.21	24.11	25.89	**26.79***	23.21
XNLI	46.85	**51.15***	47.40	46.65	—	—	—	—	42.60	**45.35***	44.20	43.45
Avg.	35.24	36.89	**36.99***	28.34	33.30	**35.18***	32.88	25.63	36.62	**37.32***	36.57	27.23

Table 6: Finetuning accuracy of mixing ratios (V1/V2/V3) vs. the GPT-2 baseline (**BL**). **Bold** marks the best among our variants; asterisk (*) denotes the global maximum.

tion 5.1. We therefore use LLaMA as the final backbone for the remaining experiments.

Tokenizer	Vocab	Eng	Nld	Zho	Mix	Avg
Regex-BBPE *(ours)*	16K	3.14	**4.27**	**0.81**	3.17	2.85
SentencePiece	16K	2.44	3.13	0.68	2.71	2.49
tiktoken cl100k	100K	**4.40**	3.13	0.52	**4.07**	**3.03**

Table 7: Compression efficiency (chars/token) across three tokenizers. **Mix** is measured on a 1:1:1 concatenation of held-out English, Dutch, and Chinese text. **Bold** indicates the highest efficiency.

6 Conclusion

We present a compact multilingual pre-training recipe for the BabyLM 2026 100M-word setting, centered on a 16K Regex-BBPE tokenizer and evaluated with Attention Residuals, Muon optimization, and three language mixtures. The revised analysis separates the original tokenizer contribution from previously proposed architectural and optimization components. Regex-BBPE improves Dutch and Chinese compression relative to the tested 16K SentencePiece tokenizer, while the AttnRes ablation shows consistent average gains over standard residuals in our V1 setup. Across the three language mixtures, the ZH-dominant V2 configuration substantially improves Chinese grammatical accuracy while preserving competitive English/Dutch results. Against the official GPT-2 baseline, zero-shot averages are slightly lower, whereas the official fine-tuning pipeline yields gains of 8.55, 9.55, and 10.09 accuracy points for English, Dutch, and Chinese. These findings support the recipe within this specific trilingual, low-data setting; broader linguistic and optimizer-specific conclusions require additional controlled experiments.

Limitations

Our experiments are intentionally scoped to the BabyLM 2026 Multilingual Track, and several limitations constrain the conclusions.

Language and mixture scope: We study only English, Dutch, and Chinese and only three mixing ratios. English and Dutch are both West Germanic, so their similar response to data redistribution should not be generalized to Indo-European languages as a whole. More languages and denser mixture sampling are needed to test whether the observed pattern extends beyond this setting.

Zero-shot and transfer scope: All three evaluation languages also occur in pre-training. Our "zero-shot" results therefore refer to task-level evaluation without task-specific fine-tuning, not language-level transfer to unseen languages. The current results do not establish such held-out cross-lingual transfer.

Component attribution: The official GPT-2 baseline differs architecturally from our system, so end-to-end comparisons cannot isolate the contribution of each component. We include a direct AttnRes ablation, but we did not run a matched Muon-vs.-AdamW ablation. Muon's specific effect therefore remains an open question.

Evaluation breadth: The official suite emphasizes syntactic probes and discriminative downstream tasks. We do not extensively evaluate open-ended generation, instruction following, or alignment behavior, which limits conclusions about broader model capability.

Ethical considerations

In alignment with the ACL Ethics Policy, we explicitly discuss the potential risks and ethical considerations associated with our low-data pre-training framework:

- **Amplification of societal biases:** Pre-training compact models on raw, uncurated multi-source corpora risks encoding and propagating implicit societal stereotypes, toxic language, and cultural biases present in the training text. Because sub-100M models lack the parameter capacity for robust internal safety alignment, they are highly susceptible to reproducing harmful distributions found in web data.

- **Curriculum-induced performance disparities:** Our exploration of heavily skewed language-mixing ratios (such as the ZH-dominant 20/20/60 configuration) deliberately introduces cross-lingual trade-offs. Intentionally restricting the token budget of certain languages to maximize global benchmarks introduces the risk of functional inequity; the model exhibits a higher propensity for factual hallucinations and unstable behavioral alignment in under-represented scripts (e.g., English and Dutch in V2).

We do not anticipate any malicious dual-use applications of this work, as it is strictly designed for sample-efficient linguistic research. However, we urge caution and the implementation of rigorous post-hoc safety filtering before adapting these data-

Task	English				Dutch				Chinese			
	Regex	**SP**	**Tik**	**BL**	**Regex**	**SP**	**Tik**	**BL**	**Regex**	**SP**	**Tik**	**BL**
BLiMP	64.41	63.90	**64.80**	69.18*	**70.90**	69.80	70.10	79.27*	**67.46**	66.80	66.20	75.44*
MultiBLiMP	**82.73**	82.10	82.50	87.14*	**78.40**	77.10	77.50	91.85*	—	—	—	—
HellaSwag	26.40	25.80	**26.60**	26.66*	**26.06**	25.60	25.80	26.37*	**26.75**	26.10	25.90	27.05*
Winogrande	**49.30**	48.70	49.10	48.80	**50.56***	49.80	50.20	49.30	**50.10***	49.20	48.50	49.71
XStoryCloze	49.38	48.50	**49.75**	49.92*	**47.70***	46.50	46.90	47.37	**47.99**	47.20	46.80	50.62*
XCOMPS	—	—	—	—	**51.54**	50.40	51.10	52.68*	**52.94**	52.20	51.60	53.90*
Avg.	54.44	53.80	**54.55**	56.34*	**54.19**	53.20	53.60	57.81*	**49.05**	48.30	47.80	51.34*

Table 8: Zero-shot tokenizer ablation on V1: **Regex**-BBPE vs. SentencePiece (**SP**) vs. **Tik**token, alongside the GPT-2 baseline (**BL**). **Bold** marks the best tokenizer; asterisk (*) denotes the global maximum.

Task	English				Dutch				Chinese			
	Regex	**SP**	**Tik**	**BL**	**Regex**	**SP**	**Tik**	**BL**	**Regex**	**SP**	**Tik**	**BL**
ARC	23.54	23.00	**24.10**	24.79*	**26.25***	25.50	24.80	24.38	**26.67***	26.00	24.50	25.00
Belebele	21.02	20.50	**21.50**	22.16*	**24.43**	23.80	22.10	30.11*	**20.45**	19.80	18.50	22.73*
BMLAMA	**12.91***	12.50	12.20	9.35	**19.54***	18.90	17.50	10.68	**13.33***	12.80	11.20	9.35
MNLI	47.47	46.80	**48.10**	51.63*	**45.21***	44.50	43.10	43.92	**48.70***	47.90	45.20	45.83
SIB-200	77.00	76.10	**77.80***	21.50	**66.50***	65.20	62.80	21.50	**80.50***	79.10	74.30	21.50
TruthfulQA	17.86	17.30	**18.10**	22.32*	**17.86**	17.10	16.50	23.21*	**24.11***	23.50	21.80	23.21
XNLI	46.85	46.10	**47.20***	46.65	—	—	—	—	**42.60**	41.80	39.50	43.45*
Avg.	35.24	34.61	**35.57**	28.34	**33.30***	32.50	31.13	25.63	**36.62***	35.84	33.57	27.23

Table 9: Finetuning tokenizer ablation on V1: **Regex**-BBPE vs. SentencePiece (**SP**) vs. **Tik**token, alongside the GPT-2 baseline (**BL**). **Bold** marks the best tokenizer; asterisk (*) denotes the global maximum.

Task	English			Dutch			Chinese		
	Qwen	**LLaMA**	**BL**	**Qwen**	**LLaMA**	**BL**	**Qwen**	**LLaMA**	**BL**
BLiMP	58.06	**64.41**	69.18*	68.42	**70.90**	79.27*	**70.80**	67.46	75.44*
MultiBLiMP	74.42	**82.73**	87.14*	**83.78**	78.40	91.85*	—	—	—
XCOMPS	—	—	—	51.16	**51.54**	52.68*	52.65	**52.94**	53.90*
HellaSwag	26.34	**26.40**	26.66*	25.86	**26.06**	26.37*	26.49	**26.75**	27.05*
Winogrande	**49.88***	49.30	48.80	49.96	**50.56***	49.30	48.64	**50.10***	49.71
XStoryCloze	48.07	**49.38**	49.92*	47.52	**47.70***	47.37	47.37	**47.99**	50.62*
Avg.	51.35	**54.44**	56.34*	**54.45**	54.19	57.81*	**49.19**	49.05	51.34*

Table 10: Zero-shot backbone ablation: Baby-Qwen vs. LLaMA, alongside the GPT-2 baseline (**BL**). **Bold** marks the better backbone; asterisk (*) denotes the global maximum.

Task	English			Dutch			Chinese		
	Qwen	**LLaMA**	**BL**	**Qwen**	**LLaMA**	**BL**	**Qwen**	**LLaMA**	**BL**
ARC	23.10	**23.54**	24.79*	25.80	**26.25***	24.38	25.50	**26.67***	25.00
Belebele	**21.80**	21.02	22.16*	23.00	**24.43**	30.11*	**21.00**	20.45	22.73*
BMLAMA	12.50	**12.91***	9.35	18.20	**19.54***	10.68	12.50	**13.33***	9.35
MNLI	46.50	**47.47**	51.63*	44.80	**45.21***	43.92	47.00	**48.70***	45.83
SIB-200	75.50	**77.00***	21.50	65.00	**66.50***	21.50	78.50	**80.50***	21.50
TruthfulQA	17.00	**17.86**	22.32*	**18.50**	17.86	23.21*	23.00	**24.11***	23.21
XNLI	45.10	**46.85***	46.65	—	—	—	41.50	**42.60**	43.45*
Avg.	34.50	**35.24**	28.34	32.55	**33.30***	25.63	35.57	**36.62***	27.23

Table 11: Finetuning backbone ablation: Qwen vs. LLaMA, alongside the GPT-2 baseline (**BL**). **Bold** marks the better backbone; asterisk (*) denotes the global maximum.

constrained recipes to user-facing or real-world multilingual production environments.

Acknowledgements

Supported in part by the Swedish Research Council (*Vetenskapsrådet*), grant 2021-04533.

References

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah A. Smith, and Yulia Tsvetkov. Do all languages cost the same? tokenization in the era of commercial language models. In *Proceedings of EMNLP 2023*, pages 9904–9923, 2023.

Rohan Anil, Vineet Gupta, Tomer Koren, Kevin Regan, and Yoram Singer. Scalable second order optimization for deep learning. *arXiv preprint arXiv:2002.09018*, 2020.

Catherine Arnett, Tyler A. Chang, and Benjamin K. Bergen. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages at LREC-COLING 2024*, 2024.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. Babylm turns 4: Call for papers for the 2026 babylm workshop. *arXiv preprint arXiv:2602.20092*, 2026a.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *arXiv preprint arXiv:2602.20092*, 2026b.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzman, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*, pages 8440–8451, 2020.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4): 128–135, 1999.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, et al. Babybabellm: A multilingual benchmark of developmentally plausible training data. *arXiv preprint arXiv:2510.10159*, 2025.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. Multiblimp 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216, 2026.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, et al. Attention residuals. *arXiv preprint arXiv:2603.15031*, 2026.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of ACL*, pages 66–75, 2018.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP: System Demonstrations*, pages 66–71, 2018.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Phillip Rust, Jonas Pfeiffer, Ivan Vulic, Sebastian Ruder, and Iryna Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of ACL-IJCNLP 2021*, pages 3118–3135, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725, 2016.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Jianlin Su, Murtuza Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yuwei Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Roziere, Naman Goyal, Eric Hambro, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, 2017.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. Findings of the babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at CoNLL*, 2023.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. In *International Conference on Machine Learning*, 2024.

Different Corpora, Different Skills? A Paired Study of Small Language Models

Edoardo Vaira and Georgios Chavales and Pierre Nugues

Department of Computer Science
Faculty of Engineering
Lund University
edoardo.vaira@icloud.com
geochaval@gmail.com
pierre.nugues@cs.lth.se

Abstract

Small language models are often trained under tight data budgets, where the choice of pretraining corpus may matter as much as model size or training recipe. We compare two contrasting corpora at matched 10-million- and 37-million-word budgets: the 1911 *Encyclopaedia Britannica* and the BabyLM 2026 strict pretraining corpus. For each corpus, we train teacher models, distilled 58-million-parameter Baby-Llama students, and no-distillation student ablations, using the same model families, training recipes, and evaluation protocol across corpora. Across BLiMP, BLiMP Supplement, EWoK, Entity Tracking, and COMPS, the corpora produce different skill profiles rather than a single quality ranking. BabyLM 2026 is consistently stronger on linguistic acceptability benchmarks, while Britannica is often stronger on Entity Tracking, a synthetic state-tracking task. The contrast appears across teacher and student models and at both budgets, with the cleanest cross-seed replication in the distilled student comparison. Targeted controls show that the central contrast persists after matching training exposure and at a longer diagnostic context; direct teacher ensembling also does not reproduce the 10M Britannica student's Entity Tracking behavior. Entity Tracking magnitudes nevertheless remain sensitive to training dynamics. For this corpus pair, corpus choice can therefore produce different relative skill profiles at small scale, motivating multi-benchmark evaluation when comparing pretraining data.

1 Introduction

Frontier language models are trained on hundreds of billions to trillions of tokens (Hoffmann et al., 2022; Touvron et al., 2023; Brown et al., 2020). At that scale, corpus effects are entangled with large heterogeneous mixtures, filtering pipelines, and training recipes. In our setting, by contrast, a model's entire pretraining history comprises only tens of millions of words from one corpus. Recurring properties of that corpus—such as register, discourse structure, and sentence length—therefore make up a larger share of what the model observes and may have a more visible effect. Yet prior small-model work (Warstadt et al., 2023; Timiryasov and Tastet, 2023) often varies the corpus alongside the budget, tokenizer, or evaluation pipeline, making it difficult to attribute performance differences to the source data itself.

Our objective is to test whether changing the pretraining corpus under a matched procedure produces a uniform performance difference or changes the model's relative strengths across skills. We use small models as controlled experimental probes of this question, not as proposed deployment systems. We compare two contrasting corpora at matched 10-million- and 37-million-word budgets: the 1911 *Encyclopaedia Britannica*, a single-source formal encyclopedia, and the BabyLM 2026 strict corpus, a heterogeneous mixture of child-directed speech, dialogue, simple Wikipedia, and transcribed speech. For each corpus, we first train larger *teacher* language models directly on that corpus. We then train 58-million-parameter Baby-Llama *student* models (Timiryasov and Tastet, 2023), either with knowledge distillation (KD)—learning from both the text and the teachers' predictions—or without it. "Matched" means that corresponding models across the two corpora use the same architecture, tokenizer-training procedure, optimization recipe, and evaluation protocol; only the source data and the vocabulary learned from it differ. Consequently, the comparison matches source-word budget and procedure, but not the resulting vocabulary or token count.

We evaluate the models zero-shot, without task-specific training, on five benchmarks. The Benchmark of Linguistic Minimal Pairs (BLiMP)

Proceedings of the Second BabyLM Workshop, pages 413–425
October 29, 2026 ©2026 Association for Computational Linguistics

and its Supplement test grammatical acceptability (Warstadt et al., 2020; BabyLM Challenge Organizers, 2026); Elements of World Knowledge (EWoK) probes basic world knowledge (Ivanova et al., 2024); Entity Tracking tests state tracking and reference resolution in synthetic discourse (Kim and Schuster, 2023); and Conceptual Minimal Pair Sentences (COMPS) probes semantic property knowledge (Misra et al., 2023). Across this suite, neither corpus dominates (Figure 1). BabyLM 2026 is consistently stronger on BLiMP and its Supplement, while Britannica is often stronger on Entity Tracking. This pattern appears in the teachers before distillation, persists in the students, and holds most clearly in the matched distilled-student comparisons at both budgets.

Our contributions are threefold. First, we provide a paired corpus comparison across two data budgets, two training seeds, teacher and student models, and distilled and non-distilled students, supplemented by a matched-exposure control. Second, we document a replicated cross-benchmark ranking reversal: BabyLM 2026 is stronger on linguistic acceptability, whereas Britannica is often stronger on synthetic state tracking. This shows, for this corpus pair, that changing the pretraining data can alter relative strengths rather than producing a uniform performance shift. Third, by observing the contrast in teachers before distillation and in students trained without it, while separately measuring corpus-dependent KD effects, we distinguish the central corpus result from a distillation artifact. Reporting only BLiMP would therefore produce a clean but partial ranking. This is especially relevant to cognitively motivated, data-limited modeling, where acquisition should be evaluated as a profile of emerging capabilities rather than a single aggregate score. Our code and configurations are available at `https://github.com/EdoardoVaira/babylm-corpora-skills`, and the Britannica corpus at `https://huggingface.co/datasets/EdoVaira/Encyclopedia-Britannica`.

2 Related work

The effect of pretraining data composition on model behavior has been studied extensively at large scale. Li et al. (2024) introduce a controlled benchmark for dataset experiments with 412M–7B parameter models; Penedo et al. (2024) show that aggressive quality filtering yields large gains for billion-token models; Longpre et al. (2024) systematically vary data age, domain coverage, quality, and toxicity; and Albalak et al. (2024) survey the broader data-selection literature. These studies share a common finding—data quality and composition matter—but they operate at web-corpus scale with models far larger than those studied here, where corpus identity is entangled with filtering, mixing, and scaling choices.

At smaller scale, targeted data has supported strong results. Eldan and Li (2023) show that a constrained synthetic story corpus can train models below 10M parameters to generate coherent English. Gunasekar et al. (2023) train competitive models from 7B tokens of textbook-style data, and the Phi family (Abdin et al., 2024) extends this curated-data approach to billion-scale models. The BabyLM Challenge (Warstadt et al., 2023) established matched-budget developmentally motivated pretraining as a testbed for small language models, and BabyLlama (Timiryasov and Tastet, 2023) showed that a 58M student can perform strongly after distillation from corpus-trained teachers. We use the BabyLM 2026 strict corpus (BabyLM Challenge Organizers, 2026) as one side of our comparison. However, prior small-model work typically trains on a single corpus or varies the corpus alongside the budget, tokenizer, or evaluation pipeline, so corpus-specific effects are confounded with other design choices.

Knowledge distillation (KD) is a standard model-compression technique (Hinton et al., 2015; Bucila et al., 2006); in our setting, it is part of the BabyLlama-style training recipe rather than the main object of study. We adopt the BabyLlama distillation recipe and 58M student architecture (Timiryasov and Tastet, 2023). Within each matched comparison, we substitute the source corpus while keeping the model family, tokenizer-training procedure, training recipe, and evaluation protocol fixed. Our contribution is not the generic observation that language models depend on their training corpus, which is long established. Rather, we use a paired small-model design to characterize a replicated reversal in the relative corpus ranking across acceptability and state-tracking evaluations.

3 Datasets

We use two pretraining corpora at matched word budgets: the 1911 *Encyclopaedia Britannica* and the BabyLM 2026 strict pretraining corpus (BabyLM Challenge Organizers, 2026). Table 1 summarizes their key properties.

3.1 Corpora

We chose Britannica because it is expert-edited with a consistent formal register, covers a wide range of topics, and provides a clear stylistic contrast to heterogeneous small-model pretraining corpora. We use the article volumes of the 11th edition (1911), scraped as volumes 1–28 and excluding the index volume. The source is digitally accessible, has a clear temporal boundary on its knowledge scope, and is public domain.

The BabyLM 2026 strict corpus provides the contrasting side of the comparison. It is a heterogeneous mixture of child-directed speech, dialogue, simple Wikipedia, transcribed speech, and other text. We use the larger `Strict` variant rather than `Strict-Small` because only `Strict` is large enough to support both a 10M-word and a 37M-word matched track from a single source.

The two corpora differ sharply in discourse structure. Britannica consists of long formal articles, for example:

> *"QUINSY, a common term for acute suppurative tonsillitis..."*

BabyLM 2026 contains many short utterances and dialogue turns, for example:

> *"*MOT: does the monkey know how to drive the train?"*

These differences in register, document length, sentence length, and topical continuity are the intended source of variation in our comparison.

3.2 Preparation and splits

We scrape Britannica articles from Wikisource, convert the scraped JSON to plain text, and apply a Britannica-specific cleanup step that normalizes Unicode, whitespace, quotes, dashes, and punctuation spacing, and removes bracketed numeric reference markers and *q.v.*/*cf.* parentheticals. For BabyLM 2026, we use the official Hugging Face release, whitespace-normalize each source row, and apply the same document-boundary token convention as for Britannica.

Both corpora are split into matched 10M-word and 37M-word training tracks, with approximately 2M-word and 1M-word validation splits respectively. Splits are formed by shuffling source documents with data-split seed 42, accumulating whole documents until the target word count is reached, and inserting explicit boundary tokens (`<s>` and `</s>`) between documents. Because whole documents are preserved, actual word counts can slightly exceed the nominal targets. The 10M training split is nested inside the 37M training split for each corpus; validation documents for the separate 10M-budget run can therefore appear in the 37M training split. Validation splits are used only for training monitoring; all reported scores come from external zero-shot benchmarks. Appendix E reports the BabyLM 2026 split manifests, and Appendix C reports token counts for all tracks.

3.3 Tokenization and sequencing

We train a separate 16,000-token byte-pair encoding (BPE) tokenizer on each track's training split, following the BabyLlama recipe (Timiryasov and Tastet, 2023). The tokenizer-training procedure is fixed, but vocabularies and token counts are corpus-specific; we therefore treat this as a matched-procedure comparison rather than a shared-vocabulary study. Because training uses fixed 128-token slices, matched source-word budgets do not imply matched token budgets. In these splits, BabyLM 2026 yields about 1.37× as many BPE training tokens as Britannica at both budgets, reflecting both corpus-specific BPE segmentation and BabyLM's many short source segments, which introduce far more document-boundary markers.

The tokenized text is segmented into 128-token slices. Longer documents span multiple slices, while shorter documents can share a slice with neighbors. During training, the slice start is randomly offset within a 128-token window; during validation, slices are deterministic and contiguous. Any final incomplete slice is discarded rather than padded.

4 Model architecture and training

To separate corpus effects from artifacts of a single architecture or distillation pipeline, we train three kinds of models per corpus: (i) larger teacher

Statistic	Brit. 10M	Brit. 37M	BabyLM 10M	BabyLM 37M
Train documents	9,181	34,295	1.17M	4.30M
Mean document words	1,089	1,079	8.6	8.6
Median document words	318	318	5	5
Mean sentence words	20.3	20.2	7.5	7.5
Word types	239k	532k	170k	344k
Tokenizer tokens / word	1.42	1.42	1.95	1.95

Table 1: Corpus properties for the matched 10M- and 37M-word training splits. Britannica is a single-source formal encyclopedia corpus; BabyLM 2026 is a heterogeneous mixture of child-directed speech, dialogue, simple Wikipedia, transcribed speech, and other text. Sentences are segmented on terminal punctuation. Token counts use each track's corpus-specific tokenizer and include document-boundary tokens.

	Llama	GPT-2	GPT-J
Layers	24	24	24
Attention heads	8	16	16
Hidden dimension	1024	1536	1536
Intermediate dimension	3072	6144	6144
Parameters	360M	705M	705M
Epochs	4	6	4
Effective batch size	128	256	128
Peak learning rate	3×10^{-4}	2.5×10^{-4}	2.5×10^{-4}

Table 2: Teacher architectures and training recipes. All teachers share a cosine learning-rate schedule with 300 warm-up steps. GPT-J is trained only at the 10M budget.

models trained directly on the corpus, (ii) 58M-parameter Baby-Llama students distilled from an ensemble of corpus-matched teachers, and (iii) 58M students trained without distillation on the same data. All models are trained from scratch on 128-token sequences with the corpus-specific tokenizer for that track. Although the architectures support a maximum context of 256 tokens, we use this extended capacity only for the diagnostic context-length check in Section 5.1. We run the full matched sweep with training seeds 2023 and 43, separate from the data-split seed, and evaluate all models zero-shot. Absolute-score tables use seed 2023 for readability, while Table 3 reports the paired corpus effects for both seeds and their across-seed variability.

4.1 Teacher models

We train multiple teacher architectures on each corpus to verify that corpus effects are not specific to a single architecture. At the 10M budget, we use three teachers (Llama-360M, GPT-2-705M, GPT-J-705M); at 37M, we use two (Llama-360M, GPT-2-705M). Table 2 summarizes the architectures and training recipes. All teachers use a cosine learning-rate schedule with 300 warm-up steps; validation loss is computed after each epoch on a fixed random subset of 8,192 sequences.

4.2 Student models

All 58M students share a single Llama-based architecture: 16 transformer layers, 8 attention heads, hidden dimension 512, and intermediate dimension 1024 (∼58M parameters). Each is trained from scratch for 6 epochs with effective batch size 32, peak learning rate 2.5×10^{-4}, a cosine schedule with 200 warm-up steps, and weight decay 0.1. As with the teacher runs, validation loss is computed after each epoch on a fixed subset of 8,192 validation sequences.

Distilled students. We follow the BabyLlama recipe (Timiryasov and Tastet, 2023; Hinton et al., 2015; Bucila et al., 2006). The student is trained against both next-token labels and soft targets obtained by averaging the raw logits of two corpus-matched teachers (Llama + GPT-2 by default, or Llama + GPT-J for the alternate-teacher 10M variant), then applying temperature $T = 2$. Teacher logits are computed on the same training batches seen by the student. The loss is

$$L = \alpha L_{\text{CE}} + (1 - \alpha) T^2 L_{\text{KL}},$$

with $\alpha = 0.5$, where L_{CE} is cross-entropy on the next-token labels and L_{KL} is the KL divergence between the student's softened distribution and the softened distribution of the averaged teacher logits. In the implementation, the model-provided cross-entropy is averaged over non-masked tokens, while the KL term uses PyTorch's `batchmean` reduction (summed over token positions and divided by batch size).

No-distillation baseline. At both word budgets, we additionally train a student with the same architecture, optimizer, schedule, and number of epochs, but using plain cross-entropy on next-token labels with no teacher signal. This lets us separate corpus effects from distillation effects.

4.3 Compute

All runs use a single NVIDIA A40 graphics processing unit (GPU) with 16-bit floating-point (fp16) training. At 10M words, training takes roughly 1–2 hours for a Llama teacher, 3–4.5 hours for a GPT-style teacher, and 4–6 hours for a distilled 58M student; no-distillation students omit teacher forward passes and take roughly 0.5–0.8 hours. At 37M words, training scales to roughly 4–6 hours for a Llama teacher, 11–16 hours for a GPT-2 teacher, about 15–20 hours for a distilled 58M student, and 2–2.5 hours for a no-distillation student.

5 Experimental results

5.1 Evaluation suite

We evaluate every trained model on a zero-shot suite drawn from the BabyLM 2026 strict evaluation pipeline (BabyLM Challenge Organizers, 2026), which extends the BLiMP-only protocol used in earlier BabyLM Challenge editions. The main text focuses on five metrics:

- **BLiMP** (Warstadt et al., 2020) – 67 minimal-pair tasks targeting syntax, semantics, and morphosyntax.
- **BLiMP Supplement** (BabyLM Challenge Organizers, 2026) – an extension of BLiMP with additional and more diverse minimal-pair tasks.
- **EWoK** (Ivanova et al., 2024) – minimal-pair world-knowledge probes.
- **Entity Tracking** (Kim and Schuster, 2023) – a synthetic state-tracking benchmark in which a context describes a sequence of operations on entities (e.g. boxes containing objects), and the model must select the continuation that correctly describes the resulting state.
- **COMPS** (Misra et al., 2023) – minimal-pair semantic plausibility.

For all benchmarks, higher scores indicate better performance. Because our main protocol uses 128-token training sequences and a tokenizer-declared context window of 128 tokens, we apply left-truncation to the zero-shot evaluation inputs. To check whether the Entity Tracking results are driven by left-truncation (since prompts at higher operation counts can exceed 128 tokens), we additionally re-evaluate all models at 256-token context—the maximum supported by the models' position embeddings.[1] The full 256-token Entity Tracking results are reported in Appendix B; the Britannica-vs-BabyLM ranking and the student-vs-ensemble patterns reported below hold at both context lengths. We also run the BabyLM reading benchmark, which reports correlations against eye-tracking (Eye) and self-paced reading (SPR) data; scores are low overall and do not change the main corpus-tradeoff conclusion, so we report them only in Appendix A.

5.2 Main results: different corpora, different skills

Figure 1 gives the seed-2023 skill profiles. We next test whether the paired corpus effects replicate under seed 43. Table 3 reports BabyLM 2026 minus Britannica for every matched model family under both seeds, so positive BLiMP/Supplement values indicate a BabyLM advantage, while negative Entity Tracking values indicate a Britannica advantage.

All 20 matched BLiMP comparisons favor BabyLM 2026 across the two seeds. Eighteen of 20 Entity Tracking comparisons favor Britannica; the two reversals are the seed-43 37M GPT-2 teacher and 10M no-KD student. Every matched KD-student comparison preserves both directions. For the default Llama+GPT-2 KD students, the paired effects computed from the unrounded benchmark-subtask accuracies (mean $\pm$ sample standard deviation over two seeds) are $+7.64 \pm 0.79$ BLiMP, $+10.83 \pm 0.86$ Supplement, and -19.63 ± 3.73 Entity Tracking at 10M; at 37M they are $+10.55 \pm 0.33$, $+7.08 \pm 1.89$, and -4.88 ± 0.25, respectively. Because these summaries use unrounded accuracies, they can differ by 0.01 from averages reconstructed from the rounded table entries. These standard deviations summarize two runs and therefore provide only a limited estimate of training variability. Separately, paired hierarchical-bootstrap 95% confidence intervals over benchmark-subtask groups and paired items exclude zero for all three metrics at both budgets and both seeds (Appendix F); these in-

[1] The 256-token setting is diagnostic rather than the main protocol: all models were trained on 128-token slices. The Llama-based models use rotary position embeddings (RoPE), while GPT-2 teachers use learned absolute position embeddings; some GPT-2 teacher rows, especially at 10M, are therefore sensitive to positions 128–255.

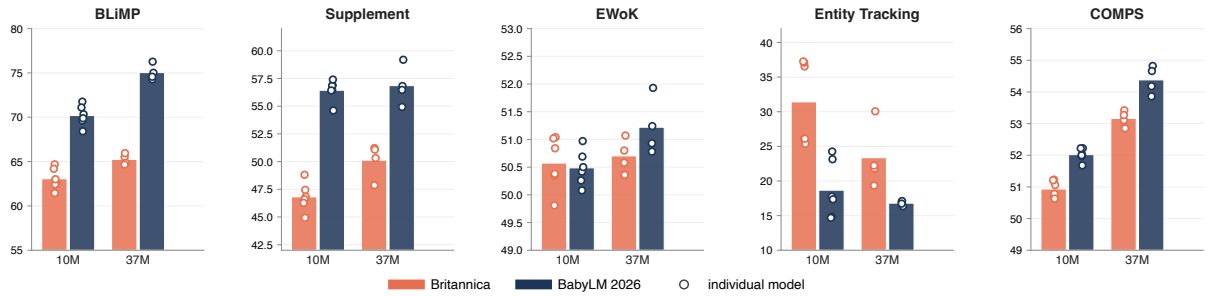


Figure 1: Skill profiles of the seed-2023 experimental sweep on five zero-shot metrics. Each panel shows one metric; bars give the mean accuracy per (corpus, budget) combination, and circles mark individual models (3 teachers + 3 students at 10M; 2 teachers + 2 students at 37M). The Britannica corpus produces higher Entity Tracking in the main matched comparisons; the BabyLM 2026 corpus produces higher BLiMP, BLiMP Supplement, and (slightly) COMPS; EWoK is essentially tied. Table 3 reports the paired effects for both training seeds.

		ΔBLiMP		ΔSupp		ΔEntity	
Budget	**Matched model family**	**s2023**	**s43**	**s2023**	**s43**	**s2023**	**s43**
10M	Llama-360M teacher	+8.19	+7.97	+7.65	+10.46	−8.38	−8.05
10M	GPT-2-705M teacher	+7.73	+8.19	+7.78	+5.21	−8.02	−16.49
10M	GPT-J-705M teacher	+5.99	+7.22	+11.96	+5.53	−12.29	−4.59
10M	Baby-Llama-58M student (KD, Llama+GPT-2)	+7.07	+8.19	+10.23	+11.44	−22.27	−17.00
10M	Baby-Llama-58M student (KD, Llama+GPT-J)	+6.91	+6.41	+10.15	+11.75	−22.58	−5.49
10M	Baby-Llama-58M student (no KD)	+6.88	+8.97	+9.95	+11.09	−2.97	+6.49
37M	Llama-360M teacher	+8.76	+8.83	+3.71	+4.72	−2.34	−1.52
37M	GPT-2-705M teacher	+10.16	+9.61	+8.87	+4.90	−13.70	+7.44
37M	Baby-Llama-58M student (KD, Llama+GPT-2)	+10.32	+10.79	+5.74	+8.42	−4.70	−5.06
37M	Baby-Llama-58M student (no KD)	+9.91	+9.66	+8.60	+5.36	−5.49	−12.23

Table 3: Paired corpus effects under both training seeds. Deltas are BabyLM 2026 minus Britannica for the same model family and source-word budget. Positive BLiMP/Supplement values favor BabyLM 2026; negative Entity Tracking values favor Britannica.

tervals quantify evaluation-set uncertainty conditional on each checkpoint, not training variability. Appendix A reports the seed-2023 absolute scores.

Because the BabyLM split yields 1.37× as many BPE tokens per matched word budget, we also run a 10M matched-exposure control covering the Llama and GPT-2 teachers used by the default KD student, that KD student, and the no-KD student. We retain the BabyLM split and tokenizer but cap each model at its Britannica counterpart's optimizer-step count, thereby matching schedule length and approximately matching 128-token sequence presentations. BabyLM remains ahead on BLiMP and Supplement in all four controlled models. Britannica remains ahead on Entity Tracking in both controlled teachers and the KD student; the no-KD comparison reverses to a small 1.28-point BabyLM advantage, consistent with that row's seed sensitivity above. Appendix D gives the design and results. This control matches total training exposure, not the number of unique tokens in the two corpora.

Tables 4 and 5 zoom in on the 58M students at both budgets.

Three further points are visible in these tables. First, the 58M students are competitive with the larger teachers on BLiMP: at 10M, the no-KD student exceeds all three Britannica teachers (63.01 vs. 61.45–62.61) and is within 0.45 points of the best BabyLM 2026 teacher (69.89 vs. 70.34). Thus, the corpus comparison remains informative even though the students are substantially smaller than the teachers.

Second, within the two 10M KD configurations, swapping GPT-J for GPT-2 as the second teacher changes student BLiMP by less than 0.7 points on either corpus and does not alter the BLiMP/Supplement versus Entity Tracking pattern. In this comparison, the corpus-associated differences are larger than those between the two teacher pairs.

Third, the cross-over persists at 37M. In the KD students, BabyLM 2026 gains a further ∼5 BLiMP points over its 10M student (71.76 →

Model	BLiMP	Supp	EWoK	Entity	COMPS
Britannica 10M					
Baby-Llama-58M (KD, Llama+GPT-2)	64.69	46.60	49.81	37.10	51.23
Baby-Llama-58M (KD, Llama+GPT-J)	64.19	46.27	**51.02**	**37.26**	51.21
Baby-Llama-58M (no KD)	63.01	47.45	50.84	26.11	50.63
BabyLM 2026 10M					
Baby-Llama-58M (KD, Llama+GPT-2)	**71.76**	56.83	50.08	14.83	52.00
Baby-Llama-58M (KD, Llama+GPT-J)	71.10	56.42	50.26	14.68	**52.22**
Baby-Llama-58M (no KD)	69.89	**57.40**	50.97	23.14	51.68

Table 4: Seed-2023 zero-shot evaluation of 58M students at matched 10M-word budget on the Britannica and BabyLM 2026 corpora. KD labels name the two-teacher ensemble used for distillation; "no KD" is the matched non-distilled baseline. Bold marks the best entry per metric across both corpora.

Corpus	Model	BLiMP	Supp	EWoK	Entity	COMPS
Britannica 37M	KD, Llama+GPT-2	65.95	51.08	50.36	21.80	53.42
Britannica 37M	no KD	64.66	47.87	50.58	**22.22**	53.27
BabyLM 2026 37M	KD, Llama+GPT-2	**76.27**	**56.82**	**51.24**	17.10	**54.66**
BabyLM 2026 37M	no KD	74.57	56.47	50.93	16.73	54.18

Table 5: Seed-2023 zero-shot evaluation of 58M Baby-Llama students at the larger 37M-word budget. KD rows use the Llama+GPT-2 two-teacher distillation ensemble; no-KD rows use the same architecture and training schedule without teacher logits. Bold marks the best entry per metric across both corpora.

76.27) while Britannica gains $\sim$1 point (64.69 $\rightarrow$ 65.95). On Entity Tracking, the Britannica KD student drops from 37.10 to 21.80, while the BabyLM 2026 KD student improves slightly (14.83 $\rightarrow$ 17.10), leaving Britannica ahead by $\sim$5 points. The 37M no-KD students show the same corpus direction (22.22 vs. 16.73 on Entity Tracking), confirming that the larger-budget contrast is not created by distillation alone. EWoK is essentially tied across corpora and budgets; COMPS shows a small consistent BabyLM 2026 advantage.

5.3 Ablations and analysis

Effect of distillation. Table 6 reports the paired effect of distillation: KD score minus no-KD score on the same corpus and budget.

For the default Llama+GPT-2 KD rows at 10M, distillation improves BLiMP by $\sim$1.7–1.9 points on both corpora and helps COMPS slightly. At 10M it is also mildly negative on Supplement and EWoK regardless of corpus. The striking result is Entity Tracking, where the effect is large but opposite-signed: $+10.99$ on Britannica versus -8.31 on BabyLM 2026. Distillation therefore amplifies the corpus-level Entity Tracking contrast rather than reducing it: a $\sim$3-point Britannica advantage between the no-KD students (26.11 vs. 23.14) grows to a 22-point gap once distillation is applied (37.10 vs. 14.83). At 37M, the KD effect on Entity Tracking is much smaller on both corpora (-0.42 on Britannica, $+0.37$ on BabyLM 2026), so the dramatic Entity Tracking amplification is specific to the 10M setting.

On BabyLM 2026, the no-KD baseline scores 23.14 on Entity Tracking (above the 20% chance level), but KD pulls the student down to 14.83 (below chance), suggesting that the distillation signal can move the student's preferences away from the correct continuation on this corpus. On Britannica, by contrast, the KD student reaches 37.10, roughly 11 points higher than either individual teacher (Llama 26.13, GPT-2 25.38).

Ensemble baseline and the role of data budget. A natural hypothesis is that the Britannica KD student simply inherits the predictions of the teacher ensemble: even when each teacher individually scores around 26, their logit mean might concentrate more reliably on the correct option. We test this directly by scoring Entity Tracking with the per-token logit mean of each KD student's two corpus-matched teachers, using the same evaluation pipeline. Table 7 reports the result.

At 10M, the ensemble scores well below the KD student in every configuration: the Llama+GPT-2 ensemble reaches 25.15 versus 37.10 for the stu-

Corpus	ΔBLiMP	ΔSupp	ΔEWoK	ΔEntity	ΔCOMPS
Britannica 10M	+1.68	-0.85	-1.03	+10.99	+0.60
BabyLM 2026 10M	+1.87	-0.57	-0.89	-8.31	+0.32
Britannica 37M	+1.29	+3.21	-0.22	-0.42	+0.15
BabyLM 2026 37M	+1.70	+0.35	+0.31	+0.37	+0.48

Table 6: Effect of knowledge distillation on the 58M student, expressed as KD score minus no-KD score on the same corpus and word budget. Positive values indicate that distillation helps.

Seed	Budget	Teacher pair / context	Teacher ensemble	KD student	Student − Ensemble
2023	10M	Llama+GPT-2 @ 128 tokens	25.15	37.10	+11.95
43	10M	Llama+GPT-2 @ 128 tokens	29.15	35.58	+6.43
2023	10M	Llama+GPT-2 @ 256 tokens	15.83	38.96	+23.13
2023	10M	Llama+GPT-J @ 128 tokens	31.29	37.26	+5.97
2023	10M	Llama+GPT-J @ 256 tokens	26.94	37.71	+10.77
2023	37M	Llama+GPT-2 @ 128 tokens	23.85	21.80	−2.05
2023	37M	Llama+GPT-2 @ 256 tokens	27.02	22.05	−4.97

Table 7: Direct ensemble comparison for the Britannica Entity Tracking models. All entries are macro averages: accuracy is first computed for each benchmark subtask (UID group) and then averaged. For the ensemble, each candidate completion is scored by the per-token logit mean of the two Britannica-trained teachers used during KD training. Main rows use seed 2023; the seed-43 row repeats the default 10M Llama+GPT-2 check. At 10M, the student exceeds the ensemble in every configuration; at 37M, the ensemble exceeds the student at both context lengths.

dent (a +11.95 gap), and the Llama+GPT-J ensemble reaches 31.29 versus 37.26 (+5.97). Repeating the default Llama+GPT-2 check under seed 43 gives the same direction (29.15 versus 35.58, a +6.43 gap). Direct logit averaging at evaluation time therefore does not explain the behavior learned by the KD student; the simple "KD inherits the ensemble" account is insufficient.

The picture reverses at 37M. The Llama+GPT-2 ensemble scores 23.85 at 128-token context and 27.02 at 256-token context, exceeding the KD student (21.80 and 22.05). The student-over-ensemble lift observed at 10M is therefore budget-specific rather than a generic property of Britannica + KD.

The 37M no-KD rows confirm a ∼5.5-point Britannica Entity Tracking advantage without distillation at the larger budget. Unlike at 10M, where KD amplifies the Entity Tracking gap from ∼3 to ∼22 points, at 37M the gap is comparable with and without KD (22.22 vs. 16.73 without KD; 21.80 vs. 17.10 with KD). The KD advantage on BLiMP also narrows slightly at 37M: the no-KD students score 74.57 (BabyLM) and 64.66 (Britannica), versus KD students at 76.27 and 65.95, yielding KD lifts of +1.70 and +1.29. These are comparable to the 10M lifts (+1.87 and +1.68), so the KD benefit on linguistic acceptability is stable across budgets, while the dramatic Entity Tracking amplification is specific to 10M.

More broadly, comparing the 10M and 37M students across both KD and no-KD conditions, the larger-budget setup improves BLiMP and COMPS on both corpora. The main skill-profile contrast is preserved at the larger budget and is visible in the no-KD baseline, supporting an association with the pretraining data rather than an effect created solely by distillation.

What Entity Tracking measures. Because Entity Tracking is the strongest single metric favoring Britannica, it is worth being precise about what it tests. The benchmark measures state-tracking and reference resolution over synthetic discourse—not encyclopedic factual knowledge. The Britannica advantage should not be read as stronger world knowledge. A more plausible interpretation is that the structural regularities of formal encyclopedia prose (long descriptive passages, frequent third-person reference, stable topical focus) better match the discourse-level patterns probed by the benchmark. We do not have a mechanistic account and treat this as an observation worth further investigation.

6 Conclusion

Our paired comparison reveals a replicated ranking inversion rather than a single corpus ordering: BabyLM 2026 is stronger on linguistic acceptability, while Britannica is often stronger on Entity Tracking. The contrast appears in teachers and distilled students, at both budgets, and—for both teachers and the distilled student—after matching total training exposure at 10M.

Distillation improves BLiMP but is not uniformly beneficial, and the no-distillation baselines show that it does not create the broad corpus contrast. Reporting only BLiMP would therefore favor BabyLM 2026 throughout, whereas Entity Tracking would often favor Britannica. Multi-benchmark evaluation is necessary to reveal these different skill profiles. For cognitively motivated small-model research, limited pretraining experience should therefore be evaluated through the profile of capabilities that emerges, not only a single aggregate score.

Entity Tracking measures synthetic state tracking, not encyclopedic knowledge. With only two corpora, we cannot identify whether register, discourse structure, repetition, or another property causes the differences; targeted corpus manipulations are needed. All code and configurations are available at https://github.com/EdoardoVaira/babylm-corpora-skills; the Britannica corpus is at https://huggingface.co/datasets/EdoVaira/Encyclopedia-Britannica.

Limitations

Our comparison is between two corpora; with $n = 2$, the relationship between corpus structure and skill profile is suggestive rather than conclusive. The student-side ablations use a single 58M-parameter architecture, a single distillation recipe inherited from BabyLlama, and zero-shot evaluation; whether the skill-profile contrast carries over to other student sizes, distillation setups, or evaluation paradigms is open. The full matched sweep uses two training seeds. Table 3 reports both runs and their paired effects; with only two seeds, the reported sample standard deviations remain limited estimates of training variability. The matched distilled-student comparisons preserve the acceptability-versus-Entity-Tracking direction in every case, while two seed-43 Entity Tracking rows outside the KD comparisons reverse direction, so we treat Entity Tracking magnitudes as seed-sensitive. The no-KD baseline was run at both 10M and 37M; at 37M the Entity Tracking corpus gap is similar with and without KD (∼5.5-point Britannica advantage), suggesting that the dramatic 10M KD amplification (Section 5.3) is budget-specific rather than a generic property of distillation. Source-word matching does not imply token-budget matching under corpus-specific BPE tokenization; Appendix C reports the difference. Our seed-2023 matched-exposure control equalizes optimizer steps and total sequence presentations at 10M but retains the larger unique BabyLM corpus; it is therefore not a trim-matched corpus study, and its training variability is not separately estimated. The Entity Tracking benchmark is synthetic, and we caution against reading its scores as evidence about open-world knowledge.

Ethics Statement

The 1911 *Encyclopaedia Britannica* contains historically dated viewpoints and outdated factual content; models trained on it may reproduce these characteristics, and the comparison reported here should not be taken as endorsement of the corpus as training data for downstream user-facing applications. The small models studied here are not intended for deployment as general-purpose language models. The skill-profile finding is observational; we caution against using it to motivate corpus selection in safety-relevant applications without further targeted evaluation.

Acknowledgments

This work was partially supported by *Vetenskapsrådet*, the Swedish Research Council, registration number 2021-04533.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 68 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas

Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A survey on data selection for language models. *Preprint*, arXiv:2402.16827.

BabyLM Challenge Organizers. 2026. Babylm 2026 strict pretraining corpus and evaluation pipeline. https://babylm.github.io/. Pretraining corpus available at https://huggingface.co/datasets/BabyLM-community/BabyLM-2026-Strict; evaluation pipeline available at https://github.com/babylm-org/babylm-eval.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Knowledge Discovery and Data Mining*.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *Preprint*, arXiv:2305.07759.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *Preprint*, arXiv:2306.11644.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, Pramod RT, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (ewok): A cognition-inspired framework for evaluating basic world knowledge in language models. *Preprint*, arXiv:2405.09605.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. *Preprint*, arXiv:2305.02363. ACL 2023.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, and 40 others. 2024. DataComp-LM: In search of the next generation of training sets for language models. *Preprint*, arXiv:2406.11794.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. 2024. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, and toxicity. *Preprint*, arXiv:2305.13169. NAACL 2024.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2023. Comps: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. *Preprint*, arXiv:2210.01963. EACL 2023.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. 2024. The FineWeb datasets: Decanting the web for the finest text data at scale. *Preprint*, arXiv:2406.17557.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *Preprint*, arXiv:2308.02019.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell, editors. 2023. *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Singapore.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

A Full Results and Split Manifests

The main text reports paired corpus deltas and the 58M student results. Tables 8 and 9 give the corresponding absolute zero-shot scores for the main teacher, KD-student, and no-KD student models. Reading columns report correlation scores for eye-tracking (Eye) and self-paced reading (SPR); reading scores were not collected for the 37M no-KD rows.

B Entity Tracking at 256-Token Context

Entity Tracking prompts vary in length depending on the number of operations: at 4–5 ops, prompts can reach ∼200–250 tokens and so are left-truncated when scored against the models' 128-token tokenizer setting. To check whether the corpus and student-vs-teacher patterns reported in Sections 5.2 and 5.3 are artifacts of this truncation, we re-evaluated every Entity Tracking row at 256-token context—the maximum supported by the models' position embeddings. Table 10 reports the comparison.

C Training Token Counts

Table 11 reports the number of BPE tokens produced by each track-specific tokenizer on the corresponding training split. Word budgets count source words before inserting document boundary markers. Token counts include the <s> and </s> markers that are present during training.

D Matched-Exposure Control

To test whether BabyLM 2026's greater number of token presentations explains the main 10M result, we retrain its seed-2023 Llama teacher, GPT-2 teacher, Llama+GPT-2 KD student, and no-KD student with the optimizer-step count capped at the corresponding Britannica run. All other settings, including the BabyLM split, tokenizer, architecture, effective batch size, warm-up, and loss, are unchanged. The caps are 3,472 steps for Llama, 2,604 for GPT-2, and 20,826 for each student. This exactly matches optimizer steps and schedule length and matches sequence presentations to within 0.03%; because the BabyLM corpus contains more tokens, it yields 2.91 effective epochs for Llama and 4.37 for the other models rather than 4 and 6. This is a matched-exposure control, not a trim-matched corpus: the number of unique tokens available during training still differs. Table 12 reports the resulting scores.

Relative to the corresponding Britannica models in Table 8, the matched-exposure BLiMP differences (BabyLM minus Britannica) are $+7.46$, $+6.53$, $+5.53$, and $+5.81$ for the Llama teacher, GPT-2 teacher, KD student, and no-KD student, respectively; Supplement differences are $+8.78$, $+6.83$, $+9.79$, and $+8.91$. Entity Tracking differences are -5.61, -11.14, -20.70, and $+1.28$. Thus the acceptability advantage remains in all four families, while the Britannica Entity Tracking advantage remains in both teachers and the central KD comparison. The no-KD Entity Tracking row is the exception and reverses slightly, consistent with its reversal under seed 43 in Table 3.

E BabyLM 2026 Split Details

Table 13 reports the manifests produced by the BabyLM 2026 split-preparation script. Word counts are counted before adding article/document boundary tokens. Because the preparation script keeps whole source documents, actual word counts can slightly exceed the requested targets.

F Paired Evaluation-Set Uncertainty

For the default Llama+GPT-2 KD-student comparisons, we estimate evaluation-set uncertainty with a paired hierarchical bootstrap. Each of 10,000 replicates first resamples benchmark subtasks (the evaluation pipeline's UID groups) and then resamples paired items within each selected group. We recompute the official macro-average—the mean of subtask accuracies—and report the BabyLM 2026-minus-Britannica difference. Pairing ensures that both models are compared on the same resampled examples. These intervals are conditional on a trained checkpoint and are distinct from the across-seed variability reported in Table 3. Table 14 reports the full intervals.

Corpus	Model	BLiMP	Supp	EWoK	Entity	COMPS	Eye	SPR
Britannica 10M	Llama-360M teacher	61.45	48.81	50.35	26.13	50.69	0.35	0.01
Britannica 10M	GPT-2-705M teacher	62.61	46.83	51.04	25.38	51.06	0.12	0.01
Britannica 10M	GPT-J-705M teacher	62.43	44.93	50.38	36.54	50.78	0.25	0.03
Britannica 10M	Baby-Llama-58M student (KD, Llama+GPT-2)	64.69	46.60	49.81	37.10	51.23	0.13	0.01
Britannica 10M	Baby-Llama-58M student (KD, Llama+GPT-J)	64.19	46.27	51.02	37.26	51.21	0.17	0.00
Britannica 10M	Baby-Llama-58M student (no KD)	63.01	47.45	50.84	26.11	50.63	0.26	0.07
BabyLM 2026 10M	Llama-360M teacher	69.64	56.46	50.42	17.75	51.97	0.17	0.00
BabyLM 2026 10M	GPT-2-705M teacher	70.34	54.61	50.50	17.36	52.22	0.18	0.03
BabyLM 2026 10M	GPT-J-705M teacher	68.42	56.89	50.69	24.25	52.02	0.34	0.02
BabyLM 2026 10M	Baby-Llama-58M student (KD, Llama+GPT-2)	71.76	56.83	50.08	14.83	52.00	0.14	0.02
BabyLM 2026 10M	Baby-Llama-58M student (KD, Llama+GPT-J)	71.10	56.42	50.26	14.68	52.22	0.15	0.02
BabyLM 2026 10M	Baby-Llama-58M student (no KD)	69.89	57.40	50.97	23.14	51.68	0.10	0.08

Table 8: Full zero-shot results for the matched 10M-word corpus comparisons. KD student labels name the two-teacher ensemble used for distillation.

Corpus	Model	BLiMP	Supp	EWoK	Entity	COMPS	Eye	SPR
Britannica 37M	Llama-360M teacher	65.52	51.22	50.80	19.34	53.11	0.41	0.02
Britannica 37M	GPT-2-705M teacher	64.86	50.32	51.07	30.06	52.85	0.39	0.01
Britannica 37M	Baby-Llama-58M student (KD, Llama+GPT-2)	65.95	51.08	50.36	21.80	53.42	0.31	0.02
Britannica 37M	Baby-Llama-58M student (no KD)	64.66	47.87	50.58	22.22	53.27	–	–
BabyLM 2026 37M	Llama-360M teacher	74.28	54.93	50.78	17.00	53.86	0.24	0.01
BabyLM 2026 37M	GPT-2-705M teacher	75.02	59.19	51.93	16.36	54.82	0.19	0.02
BabyLM 2026 37M	Baby-Llama-58M student (KD, Llama+GPT-2)	76.27	56.82	51.24	17.10	54.66	0.18	0.01
BabyLM 2026 37M	Baby-Llama-58M student (no KD)	74.57	56.47	50.93	16.73	54.18	–	–

Table 9: Full zero-shot results for the matched 37M-word corpus comparisons. Reading scores for the 37M no-KD runs were not collected; these models were evaluated only on the five core metrics.

Corpus	Model	ET @ 128	ET @ 256	Δ
Britannica 10M	Llama-360M teacher	26.13	23.00	−3.13
Britannica 10M	GPT-2-705M teacher	25.38	13.48	−11.90
Britannica 10M	GPT-J-705M teacher	36.54	32.08	−4.46
Britannica 10M	Baby-Llama-58M (Llama+GPT-2)	37.10	38.96	+1.86
Britannica 10M	Baby-Llama-58M (Llama+GPT-J)	37.26	37.71	+0.45
Britannica 10M	Baby-Llama-58M (no KD)	26.11	23.34	−2.77
BabyLM 2026 10M	Llama-360M teacher	17.75	16.54	−1.21
BabyLM 2026 10M	GPT-2-705M teacher	17.36	13.21	−4.15
BabyLM 2026 10M	GPT-J-705M teacher	24.25	25.30	+1.05
BabyLM 2026 10M	Baby-Llama-58M (Llama+GPT-2)	14.83	15.48	+0.65
BabyLM 2026 10M	Baby-Llama-58M (Llama+GPT-J)	14.68	14.74	+0.06
BabyLM 2026 10M	Baby-Llama-58M (no KD)	23.14	18.92	−4.22
Britannica 37M	Llama-360M teacher	19.34	22.91	+3.57
Britannica 37M	GPT-2-705M teacher	30.06	30.30	+0.24
Britannica 37M	Baby-Llama-58M (Llama+GPT-2)	21.80	22.05	+0.25
Britannica 37M	Baby-Llama-58M (no KD)	22.22	22.59	+0.37
BabyLM 2026 37M	Llama-360M teacher	17.00	15.96	−1.04
BabyLM 2026 37M	GPT-2-705M teacher	16.36	14.67	−1.69
BabyLM 2026 37M	Baby-Llama-58M (Llama+GPT-2)	17.10	19.42	+2.32
BabyLM 2026 37M	Baby-Llama-58M (no KD)	16.73	16.44	−0.29

Table 10: Entity Tracking accuracy at the 128-token (tokenizer-declared) and 256-token (position-embedding maximum) context lengths. The 256-token setting is a diagnostic rather than the main protocol because all models were trained on 128-token slices. Most student rows move modestly, and the Britannica-vs-BabyLM ranking holds in every matched row at both context lengths. Some teacher rows, especially the 10M GPT-2 rows with learned absolute position embeddings, are more sensitive to the diagnostic setting.

Corpus	Budget	Source words	Boundary markers	BPE train tokens	Tokens / word
Britannica	10M	10,000,066	18,362	14,216,855	1.42
BabyLM 2026	10M	10,000,092	2,331,378	19,518,278	1.95
Britannica	37M	37,002,217	68,590	52,559,454	1.42
BabyLM 2026	37M	37,002,217	8,605,126	72,172,588	1.95

Table 11: Training-token counts after corpus-specific BPE tokenization. The matched-word design follows the BabyLM convention, but it does not match the number of BPE tokens seen per epoch. In these splits, BabyLM 2026 yields about 1.37× as many training tokens as Britannica at both budgets, reflecting both corpus-specific BPE segmentation and BabyLM's many short source segments, which introduce far more document-boundary markers.

BabyLM 2026 model	Steps	Eff. ep.	BLiMP	Supp	EWoK	Entity	COMPS
Llama-360M teacher	3,472	2.91	68.91	57.59	50.60	20.52	51.51
GPT-2-705M teacher	2,604	4.37	69.14	53.66	50.12	14.24	51.65
Baby-Llama-58M student (KD)	20,826	4.37	70.22	56.39	50.95	16.40	51.66
Baby-Llama-58M student (no KD)	20,826	4.37	68.82	56.36	50.40	27.39	51.64

Table 12: Seed-2023 zero-shot results for the BabyLM 2026 10M matched-exposure control. Total optimizer steps and approximately total 128-token sequence presentations match the corresponding Britannica models.

Split	Train words	Train docs	Val words	Val docs	Source rows consumed
BabyLM 2026 10M	10,000,092	1,165,689	2,027,481	236,073	1,401,762
BabyLM 2026 37M	37,002,217	4,302,563	1,003,428	116,690	4,419,253

Table 13: BabyLM 2026 strict split manifests. Both splits are deterministic random document-level cuts from the same shuffled `Strict` source pool using seed 42.

Budget	Training seed	ΔBLiMP [95% CI]	ΔSupp [95% CI]	ΔEntity [95% CI]
10M	2023	+7.07 [+3.01, +11.11]	+10.23 [+3.54, +16.49]	−22.27 [−24.96, −19.22]
10M	43	+8.19 [+4.05, +12.36]	+11.44 [+3.32, +19.58]	−17.00 [−19.77, −13.96]
37M	2023	+10.32 [+7.21, +13.75]	+5.74 [+0.47, +11.67]	−4.70 [−7.07, −2.36]
37M	43	+10.79 [+7.46, +14.53]	+8.42 [+2.29, +14.24]	−5.06 [−7.36, −2.89]

Table 14: Paired corpus effects and percentile 95% hierarchical-bootstrap confidence intervals for the default KD students. Positive values favor BabyLM 2026; negative Entity Tracking values favor Britannica. All intervals exclude zero. Bootstrap seed: 2026.

Modeling the Developmental Shift in Telicity Acquisition

Ellie Xia[1] **Parisa Kordjamshidi**[1] **Alan Hezao Ke**[2]
[1]Michigan State University [2]University of South Carolina
{xiayili, kordjams}@msu.edu hezao@mailbox.sc.edu

Abstract

Acquiring telicity, which is the distinction between bounded (e.g., *ate an apple*) and unbounded (e.g., *ate apples*) events, requires first language (L1) learners to map surface-level and semantic cues to abstract event structures, but the computational trajectory of this mapping is not well understood. We introduce a **Difference in Surprisal** method that uses GPT2 token surprisal over paired temporal adverbial diagnostics (*in an hour* versus *for an hour*) to automatically label telicity across English CHILDES corpora, validated against expert linguist judgments. Using these labels, we train diagnostic logistic regression classifiers on 12 syntactic and lexical semantic features to compare how child speech and child-directed speech encode telicity. The two models diverge: the child model reaches near perfect accuracy through a single deterministic cue, the presence of a post-verbal determiner, while the adult model relies more heavily on verb class and other lexical semantic features, with the determiner cue neutralized. This trajectory supports Syntactic Bootstrapping: learners first exploit high-frequency structural cues as a scaffold to bootstrap, before developing fully compositional, verb-based event structures.

1 Introduction

Telicity is a core concept in the event structure of predicates, which distinguishes whether an event is bounded and culminates ('telic') or is characterized by ongoing duration without an inherent endpoint ('atelic') (Levin and Rappaport Hovav, 1994; Krifka, 1998; MacDonald, 2008; Portelance et al., 2023; Kim et al., 2024). For language learners, this distinction is crucial yet complex. Infants must learn to map surface-level morphosyntactic, syntactic, and semantic features, such as determiners, resultative particles, and verb classes, to their corresponding aspectual semantics (Wagner, 2006). Telicity is formally diagnosed by the compatibility of predicates with temporal adverbials (Dowty, 1979; Smith, 1991; MacDonald, 2010). More precisely, "in an hour" phrases are semantically felicitous with telic predicates as they denote the time span of culmination, whereas "for an hour" phrases select atelic predicates, expressing duration. For example, Figure 1 illustrates the culmination point of the event *eating an apple* is where the final state of *only the core remaining* serves as the telic boundary. The incremental stages, such as *one bite*, *half eaten* are intermediate process leading up to this final change of state.

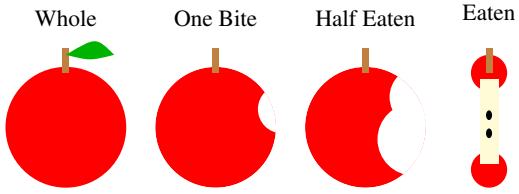


Figure 1: Apple Incremental Consumption

To give a concrete natural language example, consider the following contrast in (1) when the sentence is appended with different temporal phrases:

(1) I ate **an** apple {in an hour / #for an hour}.

Example (1) is not compatible with "for an hour" because the event *ate an apple* is terminated/completed when *an apple* is being eaten. Thus "in an hour" captures the termination of the event but "for an hour" does not. The symbol # indicates semantic anomaly or infelicity for appending "for an hour". This means that although the sentence follows the rules of grammar, the meaning sounds infelicitous or odd in that context.

More telicity related cues including plurality (2), resultative (3), partial (4), aspectual (5), and stative (6) readings, are as follows:

(2) I ate apple**s** {#in an hour / for an hour}.

(3) I ate **up** all the apples {in an hour / #for an

Proceedings of the Second BabyLM Workshop, pages 426–437
October 29, 2026 ©2026 Association for Computational Linguistics

hour}.

(4) I ate **a piece of** the apple {in an hour / #for an hour}.

(5) I **finished** eating all the apples {in an hour / #for an hour}.

(6) I **like** eating apples {#in an hour / for an hour}.

Since these distinctions (1)-(6) rely on the interface between syntax (determiner, Structural Case (Chomsky, 1957)) and semantics (boundedness, verb class), acquiring them requires L1 learners to track complex distributional cues.

We investigate the acquisition of these different cues in early English-speaking children by connecting linguistic theory with computational modeling. Specifically, we test the following hypotheses regarding the developmental trajectory of telicity. These hypotheses are potentially compatible with one another and are not necessarily mutually exclusive:

H_1: Telicity is based on the boundedness of the object (semantics of NP) (Schmitt, 1996; Krifka, 1998; MacDonald, 2010; Xia and Ke, 2026).

H_2: Telicity is primarily a property of the verb. (Vendler, 1967; Verkuyl, 1972; Dowty, 1979; Naigles, 1990; Wagner, 2006; Ramchand, 2008).

H_3: Telicity is induced by prepositional phrases or resultative particles (e.g., `up`, `off` in English) (Cheng and Huang, 1994; Sybesma, 1997; Lin, 2003; Kratzer, 2005; Song, 2018; Tay, 2024).

H_4: Telicity is a mixed semantic property of different cues (Borer, 2005; Zhao et al., 2021; Hu, 2024).

The current study addresses L1 acquisition on telicity, specifically targeting the gap between formal linguistic theories of event boundedness and the empirical distribution of telicity cues in naturalistic child and child-directed speech by applying natural language processing methods, based on CHILDES data (MacWhinney, 2000) and recent developments in large language models (LLM) (Radford et al., 2019). Using LLM-based token surprisal and probability-based analysis within a computational pipeline, this project tracks how the use of telic predicates by children develops over time compared to adults' speech. The results will inform theories of the acquisition of aspectual semantics by highlighting developmental patterns and potential avoidance behaviors, demonstrating how computational linguistics can enrich traditional acquisition research. The pipeline is shown in Figure 2. It consists of three primary stages designed to test our core hypotheses (H_1 to H_4). First, in Stage I, we partition the English CHILDES corpora into child utterances and adult child-directed speech. Second, in Stage II, we extract 12 features, while an LLM-based Labeling Engine calculates the diagnostic surprisal difference (ΔS) to determine the telicity of each utterance. Finally, Stage III applies diagnostic probing via logistic regression. By comparing the feature weights relied upon by the child model versus the adult model, this framework allows us to quantitatively track the developmental shift from syntactic bootstrapping to semantic composition.

In framing these hypotheses, we specifically evaluate the mechanism of the Syntactic Bootstrapping Hypothesis (Gleitman, 1990) under the learnability constraints of the Transparency Principle (Wagner, 2006; Lightfoot, 2017), which posits that learners initially prioritize transparent, one-to-one surface mappings before mastering complex compositional operations. Furthermore, we clarify that our empirical scope targets language *production* via naturalistic corpus transcripts rather than experimental comprehension. While experimental studies probing sentence acceptability test comprehension, observational corpus studies reflect production. Prior acquisition literature notes a marked asymmetry where children may demonstrate adult-like competence in comprehension while lagging in production, or vice versa (Wagner, 2001; Conroy and Lidz, 2007; Hendriks, 2016).

Contributions: **(a)** Introduce a novel "Difference in Surprisal" method to automatically annotate telicity in large corpora using open-weight LLM. **(b)** Provide quantitative computational evidence supporting the Syntactic Bootstrapping Hypothesis in early L1 acquisition. **(c)** Identify a measurable developmental shift where reliance on syntactic cues (determiners) transitions to lexical semantic cues (verb class).

2 Related Work

Telicity lies at the interface of the verb class, morphology, and syntax, etc. Prior research has examined this interface through two lenses: developmental psycholinguistics and computational modeling (Bloom et al., 1980; Shirai and Andersen, 1995; van Hout, 1998; Zhao et al., 2021; Kim et al., 2024).

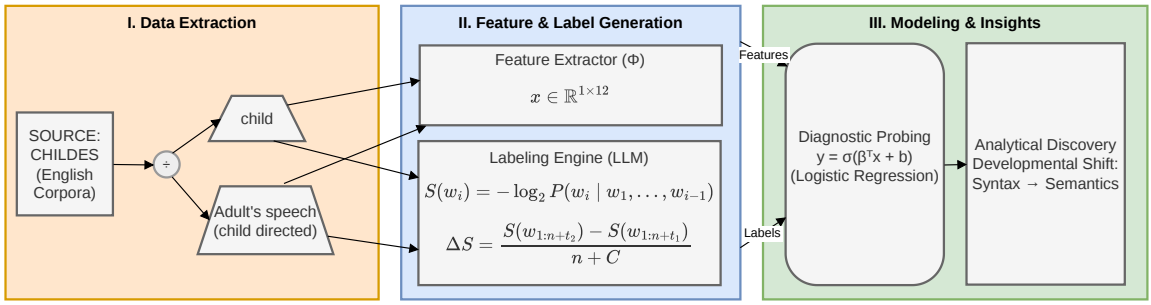


Figure 2: L1 Telicity Acquisition Pipeline

2.1 Theoretical Acquisition of Telicity

Theoretical work suggests that children acquire aspectual mappings in a non-uniform manner, heavily influenced by the **Transparency Principle** (Wagner, 2006; Lightfoot, 2017). This principle predicts that one-to-one form-meaning mappings are acquired earlier than complex compositional ones.

Overt vs. Compositional Cues. Children as young as age 3 reliably interpret verb-particle constructions (e.g., *up* in English, *op* in Dutch) as encoding telic events (Wagner, 2006). These particles serve as transparent markers of completion. In contrast, children show a marked delay in acquiring telicity induced by object quantization (e.g., *drink milk* vs. *drink a glass of milk*). Adult-like judgments for these compositional structures often do not emerge until after age 5 (van Hout, 1998).

Tense-aspect Morphology Selection. Early production data reveals that young children initially restrict tense-aspect morphology to specific semantic classes. Bloom et al. (1980) and Shirai and Andersen (1995) observe that children predominantly apply past-tense marking to telic verbs (e.g., *broke*, *made*) and progressive marking to atelic verbs (e.g., *riding*, *laughing*). This suggests that early grammar is heavily lexicalized before becoming fully productive.

Cross-Linguistic Variation. The strength of these mappings varies by languages. Dutch adults, for instance, assign telic interpretations to bounded objects far more consistently ($\approx 78\%$) than English adults ($\approx 25\%$), suggesting that English relies more heavily on pragmatic inference than on object quantization alone in the experimental work (van Hout, 1998).

2.2 Computational Modeling of Telicity

Recent computational work has begun to probe whether neural networks capture the structural and lexical nuances required to model aspectual interpretation and clause-level situation types (Siegel and McKeown, 2000; Zarcone and Lenci, 2008; Zhao et al., 2021; Friedrich et al., 2023; Kim et al., 2024).

Automatic Aspectual Classification. Early machine learning frameworks approached aspectual detection as a supervised task optimized over type-level linguistic markers. The foundational work of Siegel and McKeown (2000) introduced methods to combine sparse, rule-based linguistic indicators–such as the frequency of progressive markers or temporal adjuncts–to classify whether a verb root function natively behaves as stative or dynamic. Building upon these type-based baselines, Friedrich et al. (2016) extended the task to the token-clausal level, using sequential labeling to capture clause-level situation entities (e.g., episodic events, habits, or states). More recently, Friedrich et al. (2023) provided a survey mapping these historical feature-engineering baselines to modern contextualized spaces, emphasizing that tracking compositional aspect remains a critical, non-trivial bottleneck for modern NLP.

Distributional Semantics. To relax the need for hand-crafted syntactic rules, Kim et al. (2024) explored aspectual boundaries through distributional semantics. Using GloVe embeddings, they modeled telicity as a function of lexical vector co-occurrence, testing whether the raw statistical probability of structural complements appearing alongside specific verbal heads could predict human telicity judgments. Their findings demonstrate that while static distributional signals correlate with coarse telic environments, they inherently lack the hierarchical structural dimensions necessary to compute complex compositional telicity or capture fluid aspectual boundaries in context.

Aspectual Composition and Coercion. Another line of structured computational research evalu-

ates semantic coercion—where an inherently atelic predicate is forced into a bounded interpretation by its arguments. Structured distributional models and distributional memory frameworks (Chersoni et al., 2019) have been deployed to quantify the cognitive processing load of these aspectual transitions. These models show that tracking thematic fit and argument-driven expectations can capture semantic composition, though they often sctruggle to scale outside strictly constrained, experimental sentences.

Transformer Sensitivity. Most recently, researchers have turned to deep, pretrained language models to probe implicit semantic boundaries. Zhao et al. (2021) demonstrated that pretrained transformers (e.g., BERT, RoBERTa) can infer certain telicity patterns, particularly those anchored to explicit lexical cues. However, their findings reveal that models do not fully match human sensitivity; while humans integrate a wide range of pragmatic, flexible, and contextual cues, transformers rely disproportionately on shallow, surface-level artifacts like post-verbal determiners or explicit temporal units.

3 Methodology

Our approach combines LLM-based surprisal with classical machine learning (i.e., linear regression model) to understand how telicity cues are used by the learners. First, we calculate the difference in token surprisal between telic and atelic diagnostic contexts to automatically generate binary telicity labels across the corpus. Second, using these LLM-derived annotations as our target labels, we train diagnostic classifiers to determine which linguistic features best predict telicity labels. Last, by comparing the feature weights of a model trained on child speech against one trained on adult child-directed speech, we can quantitatively map the developmental trajectory of these cues. The details of this process are explained in the following sections.

3.1 Data and Preprocessing

To investigate how L1 speakers, specifically English speaking children, learn the concept of *telicity*, we utilize the CHILDES database (MacWhinney, 2000). In particular, we extract English corpora including English-NA (North American) and English-UK (British) (MacWhinney, 2010). We partition the data into Child Speech (utterances by the child) and Parent/Caregiver(Adult) Speech (utterances directed at the child). To process the raw corpus transcripts for computational modeling, we first clean the data by removing non-linguistic transcriber annotations (e.g., pause markers, phonetic notes). Utterances are filtered to exclude extreme length outliers. Short fragments under three tokens are removed. Utterances that do not contain a verb are removed because they lack the core predicate-argument that is necessary, given that adjectival predicate is always stative (atelic). Conversely, utterances exceeding 30 tokens are excluded to prevent transcription artifacts from dependency parsers (de Marneffe et al., 2021). This yields a dataset optimized for information-theoretic evaluation. The processed datasets are used to extract the 12 linguistic features described in Section 3.3. The use of adult child-directed speech (CDS) as the adult benchmark is grounded in extensive developmental literature demonstrating that CDS input constitutes the primary linguistic environment shaping L1 distribution and constraint induction (Snow, 1972; Farwell et al., 1979). The full filtering breakdown—from raw counts to post-filtering totals across subsets—is detailed in Appendix.

3.2 Telicity Labels via LLM-Based Surprisal

To generate consistent telicity annotations in a large corpus without manual effort, we propose a novel method, namely the "Difference in Surprisal" method. Previous research has established that LLM-based surprisal serves as a robust diagnostic for recovering latent grammatical structure and identifying semantic infelicity (Kwiatkowski et al., 2012; Goodkind and Bicknell, 2018; Fan and Reilly, 2020; Wilcox et al., 2020; Cong, 2024; Gambi et al., 2024). We define telicity based on the compatibility with temporal adverbials: *telic* predicates prefer t_1 ="in an hour" (bounded), while *atelic* predicates prefer t_2 ="for an hour" (durative).

We employ LLM (GPT-2) to calculate the token-level surprisal $S(w_i)$. The surprisal of a token is defined as the negative log probability of its occurrence given the context:

$$S(w_i) = -\log_2 P(w_i \mid w_1, \ldots, w_{i-1}) \quad (1)$$

For a given sentence context (e.g., "He ate the apple"), we append the diagnostic phrases and compute the mean surprisal over the total phrase tokens ($n + C$= the number of tokens in the original sentence plus the three-token diagnosis phrase "for an

hour" or "in an hour"). The diagnostic signal ΔS is defined as:

$$\Delta S = \frac{S(w_{1:n+t_2}) - S(w_{1:n+t_1})}{n + C} \quad (2)$$

A positive ΔS indicates that sentence with "for an hour" is more surprising (less likely) than sentence with "in an hour," signaling a more **Telic** event, and the bigger the positive number, the more likely the sentence is telic. We discretize this continuous score into binary labels (Telic/Atelic) by applying the default threshold teasing apart telic from atelic events.

Crucially, rather than prompting the language model to generate categorical judgments (Zheng et al., 2023), our Difference in Surprisal method derives binary labels through next-token probability aggregation over constrained temporal modifiers. Using mean token surprisal (normalizing the summed negative log-probabilities by phrase length $n + C$) ensures that the metric captures per-token information density without introducing length-dependent bias.

3.3 Features for Telicity Classification

We extracted the 12 linguistic features using an automated Python pipeline. Tokenization, lemmatization, and POS tagging, which identify morphosyntactic cues, were performed with Stanza (Qi et al., 2020). Syntactic configurations, including core arguments and linear dependencies like verb_det_dist, were mapped using the Universal Dependencies parser in Stanza. Finally, we operationalized Lexical Semantics (Verb_Class) by matching lemmatized verbs against a custom, manually annotated aspectual verbs.

Syntactic Configuration: These features capture the internal structure and complexity of the VP, specifically looking for cues of boundedness. This analysis includes binary and count-based measures of the words that follow a verb. First, this involves checking whether or not determiners like 'the' or 'a' appear immediately after the verb (`n_determiner_after_verb`) (Xu and Schmitt, 2025a,b,c), the number of particles like 'up' or 'off' (`n_particle`), and checking for the presence of prepositional phrases that indicate a destination or result (`has_PP_goal`). Second, we also include linear distance metrics which measure the gap between the verb and its introductory word (`verb_det_dist`) or the verb and its actual object (`verb_obj_dist`), both of which serve as proxies for the syntactic configuration of the predicate (Gibson, 2000; Liu et al., 2017; Futrell et al., 2015; Levy, 2008; Gildea and Temperley, 2010). Third, we add the presence of core arguments (`n_subject`, `n_obj`) which are necessary to saturate the predicate's thematic requirements

Lexical Semantics: We incorporate the inherent aspectual class of the verb (`Verb_Class`), based on the classification of argument structure (Chomsky, 1957, 1995; Borer, 1993; Levin and Rappaport Hovav, 1994; Culicover and Jackendoff, 2005; Ramchand, 2008; Roberts, 2010). This feature identifies whether the verb is inherently telic or not, which provides the semantic baseline for telicity judgments (Anderson, 2017). We characterize Verb_Class into three types: `Verb_Class_0` (Unergative): Intransitive verbs where the subject is an agent (e.g., *cry*, *dance*). These are inherently atelic as they describe ongoing activities without a natural endpoint. `Verb_Class_1` (Unaccusative): Intransitive verbs where the subject undergoes a change of state (e.g., *come*, *arrive*, *fall*). These are inherently telic (eventive), as the endpoint is built into the lexical meaning. `Verb_Class_2` (Transitive): Verbs that take a direct object (e.g., *eat*, *drink*). These exhibit variable telicity (aspectual composition), where the boundedness of the event is determined by the quantization of the internal argument (e.g., "drinking milk" vs. "drinking a glass of milk").

Morphosyntactic Cues: These features track overt grammatical aspect. We operationalize verb particles via a binary presence indicator `has_particle` (replacing count metrics to avoid spurious multiverbal readings). To capture aspectual morphology without redundancy, we distinguish between `has_aspect_marker` (a binary flag denoting the presence of an auxiliary such as perfect *have/has*) and `aspect_marker_count` (an integer feature capturing stacked aspectual marking, such as perfect progressive *have been V-ing*). In English, these stacked aspectual layers fundamentally alter telic boundedness relative to simplex aspectual marking.

3.4 Diagnostic Probing Models

To compare the "acquisition logic" of children versus adults, we train **Logistic Regression** classifiers to predict the LLM-derived telicity labels (as described in Section 3.2) using the 12 extracted features. We train two separate models: Child Model

is trained on child's utterances; Parent Model is trained on adult's utterances.

By analyzing the coefficients (β) of these logistic regression models, we quantify the importance of each linguistic cue. A high positive coefficient for a feature (e.g., `n_determiner_after_verb`) implies that the presence of the feature is a strong predictor of telicity for that group. This allows us to test our hypotheses (H_1 to H_4) by observing which features impact the decision boundaries on telicity labels.

4 Results

Looking at the experimental results in Table 1, we can go back to evaluate the hypotheses and explain how the presence of post-verbal determiners as part of syntactic configuration directly informs producing telicity by children. We first analyze the performance of the diagnostic probing models (Logistic Regression) trained to predict the LLM-derived telicity labels from linguistic features in Adult vs. Child speech. We apply 70% of our data for training and 30% for testing.

Child Model: In contrast, the Child Model (Table 1) achieves perfect convergence (100%) across all metrics. As it will be discussed in Section 6, this is not a result of superior generalization, but rather indicative of a strong categorical separation driven by a single feature (`n_determiner_after_verb`). The model effectively learns a deterministic rule: the presence of a determiner implies telicity in the child subset.

Adult Model: The Adult model is trained on the adult speech. As shown in Table 1, the model achieves a robust performance with an accuracy of 0.93 and a macro F1-score of 0.84 in the testing phase.

Table 1: Performance metrics for Child and Adult regression models based on 12 linguistic features

Model	Class	Metric	Train	Test
Child	**Atelic**	Precision	1.00	1.00
		Recall	1.00	1.00
		F1-Score	1.00	1.00
	Telic	Precision	1.00	1.00
		Recall	1.00	1.00
		F1-Score	1.00	1.00
	Overall	**Accuracy**	1.00	**1.00**
Adult	**Atelic**	Precision	0.71	0.94
		Recall	0.84	0.98
		F1-Score	0.77	0.96
	Telic	Precision	0.80	0.73
		Recall	0.66	0.48
		F1-Score	0.72	0.58
	Overall	**Accuracy**	0.75	**0.93**

5 Expert Validation

To validate that our LLM-derived surprisal metric (`Diff_MeanBits`) aligns with human intuition, we evaluated the metric against human-annotated sentences. We conducted an expert judgment task. Three American English-speaking linguistic Ph.D. or Ph.D. students evaluated a stratified sample of 200 utterances ($n = 100$ Child, $n = 100$ Child-Directed). Linguists judged whether each sentence was more compatible with the telic diagnostic *in an hour* or the atelic diagnostic *for an hour*, or both if examples can have a two-way reading, which means the sentence can have both a telic and an atelic reading. Let P_1 and P_2 denote the proportions of telic and atelic judgments, respectively, computed over all three votes from the experts. We calculated a Telicity Index (TI):

$$TI = \frac{P_1}{P_1 + P_2} \tag{3}$$

An utterance is assigned a Ground Truth of **Telic** if $TI >= 0.5$, and **Atelic** otherwise. Then we compare the labels produced by surprisal difference in 3.2 with the experts' annotation.

We first use raw surprisal differences with a naive threshold of **0**. The metric captures a highly predictive signal for human judgments (Pearson $r = 0.266$), yielding an initial classification accuracy of **70.5%**. Notably, this raw correlation performs competitively alongside established evaluation benchmarks for human annotation (Pearson $r = 0.217$, $p < 0.001$; Peelle et al. 2020). In Table 2, the pre-threshold confusion matrix reveals a conservative bias: the model is accurate at identifying atelic events (True Negatives = 120) but underestimates telic events.

Table 2: Confusion Matrix (Pre-Threshold).

	Pred Atelic	Pred Telic
True Atelic	120	34
True Telic	25	21

5.1 Inter-Rater Reliability and Human Consensus

The evaluation data was independently coded by three expert linguists. Inter-rater reliability metrics

demonstrate highly robust convergence and a stable semantic signal across the dataset. The annotators achieved a majority agreement rate ($\geq 2/3$ raters) of 96.00% (192/200 sentences), corroborating the ≈95.5% reliability baseline observed in preliminary development cycles. Absolute unanimity (3/3 raters) was observed in 43.50% (87/200) of the test items.

While absolute unanimity (3/3) reached 43.50%, this variation does not signify noisy annotation quality, but reflects the inherent semantic indeterminacy of aspectual interpretation. As observed in recent computational work on telicity by Ma and Miyao (2026), there exists a persistent divergence between discrete theoretical categories and empirical human perception, claiming that telicity judgments inherently cause gradient acceptability. In our evaluation sample, disagreements predominantly stemmed from ambiguity flags (69/200 items where raters permitted dual readings or split on optional contextual licensing). These ambiguous items split evenly across child ($n = 39$) and parent ($n = 30$) speech and were symmetrically distributed across determiner-present and determiner-absent clauses, confirming that rater variance is not due to the post-verbal determiner effect. Furthermore, rater divergence also showed degrees of pragmatic accommodation: stricter syntactic readings disallow coercion in sentences like *I ate an apple for an hour*, whereas more permissive readers accommodate partial consumption readings (Krifka, 1998; Xu, 2026).

As summarized in Table 3, the model exhibits alignment with human linguistic judgments. The model achieved a lenient accuracy rate of 87.50% against the consolidated human majority consensus, with an average per-rater accuracy of 88.50%.

Table 3: LLM Agreement and Accuracy Rates Under Evaluation from three LRs (Linguist rater) ($N = 200$)

LR	Agree (%)	Sentence Count
#1	94.50%	189
#2	91.00%	182
#3	80.00%	160
Average	**88.50%**	*Cross-rater mean*
Telicity Index	**87.50%**	**175 / 200**

5.2 Evaluation Threshold Optimization

We maintain an explicit distinction between the *labeling decision threshold* ($T = 0$) and the *empirical evaluation boundary* (ϵ), which will be introduced in this section. The primary corpus labels are generated using a default threshold $T = 0$, which serves as the theoretically motivated point of indifference between the two diagnostic adverbials.

Furthermore, in order to test if the default threshold 0 is aligned with human judgment, the sweep across candidate thresholds in Figure 3 represents a post hoc, descriptive validation designed to benchmark how continuous model surprisal aligns with human majority voting across a continuous Telicity Index. Optimizing this boundary yields an empirical cutoff of $\epsilon = -0.198$, which lies adjacent to the theoretical default of 0. Importantly, across both the child and adult corpora, over 98% of sentences fall strictly outside the narrow interval $[-0.198, 0)$, meaning that less than 2% of binary classifications would shift under the empirical boundary. We therefore preserve $T = 0$ for all primary experimental probing to prevent overfitting on the 200-sentence validation sample.

Table 4: Confusion Matrix (Post-Threshold).

	Pred Atelic	**Pred Telic**
True Atelic	49	28
True Telic	46	77

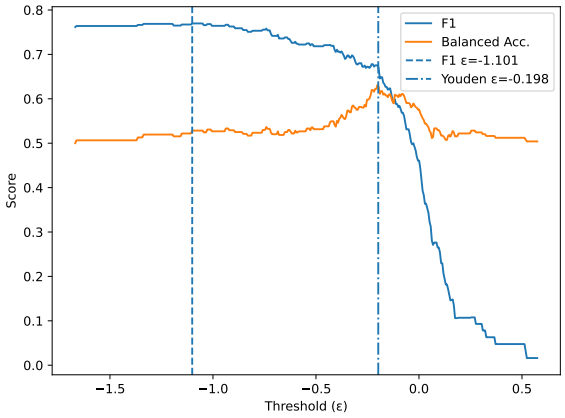


Figure 3: F1 score and balanced accuracy across thresholds for mean surprisal difference (ΔS)

Under this empirical evaluation boundary $\epsilon = -0.198$, the point-biserial correlation between the binary model output and the continuous human expert index advances to $r = 0.2999$ (□ highly significant at $p < 0.00002$). The most pronounced statistical improvement occurs within the unambiguous categorical subset ($N = 53$), where items possess definitive human majorities (*strictly telic*

Table 5: Logistic Regression (selected) Coefficients in Child Model.

Variable	Coefficient (β)	z-score	$P > \|z\|$
n_det_after_verb	**16.07**	—†	***
verb_det_dist	3.7875	218.14	***
verb_obj_dist	1.2410	115.88	***
n_subject	0.7290	50.49	***
has_aspect_marker	0.1141	2.44	*
n_obj	-2.8449	-146.21	***
verb_pos_ratio	-1.2218	-119.44	***

† *Note: Unstable z-score due to quasi-complete separation.*

Table 6: Logistic Regression (selected) Coefficients in Adult Model.

Variable	Coefficient (β)	z-score	$P > \|z\|$
Verb_Class_1	3.7520	2.99	**
verb_det_dist	2.2804	24.75	***
Verb_Class_2	2.0871	11.82	***
verb_obj_dist	1.2991	29.66	***
has_PP_goal	0.5279	5.99	***
n_subject	0.2379	4.64	***
n_det_after_verb	**-0.0210**	-0.22	0.821 (n.s.)

or *strictly atelic*). Within this group, the categorical correlation rises from an unoptimized $r = 0.2666$ to an **optimized** $r = 0.4188$, achieving high statistical significance ($p = 0.0018$).

6 Discussion

6.1 "Determiner clue" in Child Speech

A finding in our results is the Child Model's near-perfect accuracy (1.00) and the extreme magnitude of the n_determiner_after_verb coefficient (16.07) compared to the Parent Model (-0.02). At first glance, this might suggest a trivial tautology in the dataset: telic events often contain determiners. However, the fact that this feature is the *sole* dominant predictor for children—but completely neutralized in the parent model—points to a reality that in parent speech, because parent utterances are syntactically more complex and lengthy, determiners are distributed more variably across the clause. Consequently, for adults, telicity appears to be derived via implicature rather than reliance on a specific structural cue.

Our results provide computational evidence for the Syntactic Bootstrapping Hypothesis (Gleitman, 1990; Wagner, 2006). The Syntactic Bootstrapping Hypothesis proposes that children learn the meanings of verbs by observing the grammatical structures (**the syntax**) in which those verbs appear. For example, if a child hears a verb with both a subject and an object, they can deduce it describes a relationship between two entities, even if they don't know the specific action yet (Wagner, 2006). In our experiment, child model essentially learns a heuristic: "If there is a determiner, the event is telic."

This finding directly supports the logic of bootstrapping as we have explained. If a simple linear regression model trained on child data can predict telicity based on the syntactic configuration of post-verbal determiners, it proves that these structural metrics are meaningful markers that a child can use to "bootstrap" their way into understanding complex telicity. Figure 4 indicates such result.

6.2 Structural cues: Distance Matters

The most striking finding from our logistic regression analysis (Table 5) is the outsized predictive power of verb_det_dist ($\beta = 3.78$) and the "perfect separation" caused by the presence of determiners. This suggests that explicit post-verbal syntactic configurations effectively override inherent lexical defaults during early language acquisition, aligning with classic structural bootstrapping accounts (Gleitman, 1990). Because determiners constitute a highly frequent, closed-class category, they provide a dense, reliable distributional cue in the immediate post-verbal domain (Gillette et al., 1999; Yang, 2016). In contrast, verbs belong to an open lexical class characterized by significantly higher structural variety and sparser individual token frequencies, making the localized phrasal geometry a more robust anchor for the model than underlying lexical aspect.

6.3 Developmental Shift: From Syntax to Semantics

The shift in predictive features suggests a fundamental change in telicity production. As shown in Table 6, the Adult model stands in substantial contrast to the Child Model (Table 5): the feature n_determiner_after_verb coefficient drops near zero for adults, while semantic features like Verb_Class become highly significant ($\beta = 3.752$). This divergence reflects a transition in the "acquisition trajectory" from syntactic to semantic cues, a cognitive shift visually summarized in Figure 4.

Child Model/Syntax: The Child Model relies on the explicit presence of quantized objects (determiners) to identify telicity. This is a high-precision but brittle rule (e.g., it fails on "I ate popcorn" vs

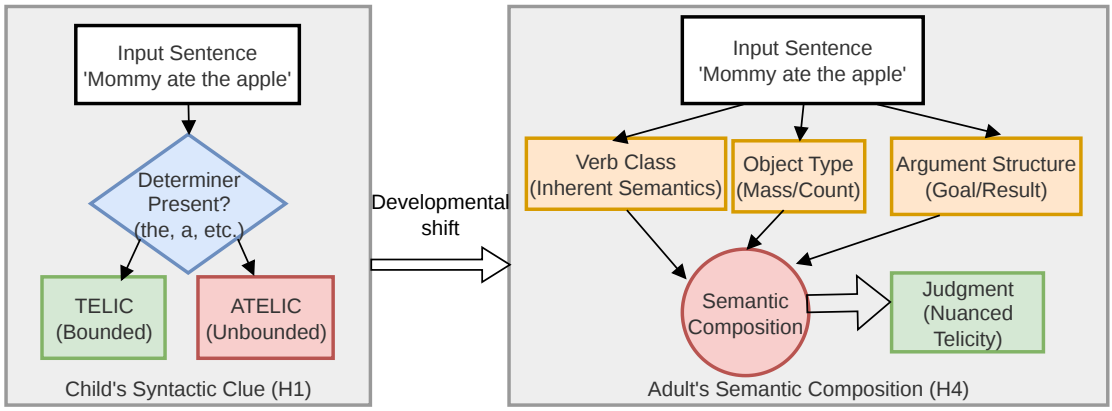


Figure 4: Conceptual model of the developmental shift in telicity acquisition

"I ate the popcorn" if the child doesn't understand mass nouns yet). It confirms that H_1 holds in child's production of telicity.

Adult Model/Semantics: The Adult Model integrates the inherent aspectual class of the verb itself (e.g., recognizing that *arrive* is inherently telic regardless of other cues). It confirms that H_4 holds in adult's production of telicity.

7 Conclusion

This study provides a novel computational bridge between formal event semantics and L1 acquisition theory. By connecting LLM-based surprisal with diagnostic probing, we have identified a developmental shift in the acquisition of telicity. Our results demonstrate that while children initially "bootstrap" event boundedness through a high-precision, surface-level "Determiner Clue" (H_1), adult speakers transition toward a more integrated model that prioritizes inherent lexical semantics (H_4). Future work will investigate whether this developmental pattern holds across morphologically richer (Spanish, German) or lesser (Mandarin, Thai) languages to determine if the "Determiner Clue" is a universal bootstrapping strategy or a language-specific adaptation to English grammar.

Limitations

Our study utilizes GPT-2 trained on adult web text, which measures alignment with adult norms rather than with the language input or language production typical in child language acquisition. Furthermore, the reliance on manual CHILDES transcripts may overlook multi-modal cues, i.e., gestures.

Ethical considerations

The human evaluation component of this study was conducted with the approval of the Michigan State University Institutional Review Board (IRB Protocol STUDY202500548). Three expert linguists in linguistics program at Michigan State University participated in the evaluation on a voluntary basis. All participants were provided with a detailed description of the task and research objectives, and written informed consent was obtained prior to data collection.

Acknowledgment

We express our sincere gratitude to Dr. Jason Smith, Drake Howard, and Millie Hacker for their generous help with data annotation during Fall 2025 and Dr. Andrew McInnerney for his suggestion on coding. We also thank the audiences at the 39th Annual Conference on Careers, Alumni and Linguistics at Michigan State (CALMS) 2025, Human Sentence Processing (HSP39) and Midwest Speech and Language Days (MSLD) 2026 for their valuable feedback and comments.

References

Curt Anderson. 2017. Contextual factors in children's calculation of telicity. In *Proceedings of the 41st Annual Boston University Conference on Language Development*, pages 20–30. Cascadilla Press.

Lois Bloom, Karin Lifter, and Jeremie Hafitz. 1980. Semantics of verbs and the development of verb inflection in child language. *Language*, 56(2):386–412.

Hagit Borer. 1993. The projection of arguments. In Elena Benedicto and Jeffrey Runner, editors, *Functional Projections*, volume 17, pages 19–47. Univer-

sity of Massachusetts Occasional Papers, Amherst, MA.

Hagit Borer. 2005. *Structuring Sense II: The Normal Course of Events*. Oxford University Press, Oxford; New York.

Lisa Lai-Shen Cheng and C.-T. James Huang. 1994. On the argument structure of resultative compounds. In Matthew Chen and Ovid Tzeng, editors, *A Festschrift for William S.-Y. Wang*, pages 187–221. Pyramid Press, Taipei.

Emmanuele Chersoni, Enrico Santus, Ludovica Pannitto, Alessandro Lenci, Philippe Blache, and C-R Huang. 2019. A structured distributional model of sentence meaning and processing. *Natural Language Engineering*, 25(4):483–502.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, 's-Gravenhage.

Noam Chomsky. 1995. *The Minimalist Program*. MIT Press, Cambridge, MA.

Yan Cong. 2024. Second language learning of degree expressions: A computational approach. *Natural Language Processing*, pages 1–23.

Anastasia Conroy and Jeffrey Lidz. 2007. Production/comprehension asymmetry in children's why questions. In *Proceedings of the 2nd Conference on Generative Approaches to Language Acquisition North America (GALANA)*, pages 73–83, Somerville, MA. Cascadilla Proceedings Project.

Peter Culicover and Ray Jackendoff. 2005. *Simpler Syntax*. Oxford University Press, Oxford.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal dependencies. *Computational Linguistics*, 47(2):255–308.

David Dowty. 1979. *Word Meaning and Montague Grammar*. D. Reidel Publishing Company, Dordrecht.

Xi Fan and Ronan Reilly. 2020. Reading development at the text level: an investigation of surprisal and embedding-based text similarity effects on eye movements in chinese early readers. *Journal of Eye Movement Research*, 13(6):10.16910/jemr.13.6.2.

Carol Farwell, Catherine Snow, and Charles Ferguson. 1979. Talking to children: Language input and acquisition. *Language*, 55:449.

Annemarie Friedrich, Alexis Palmer, and Manfred Pinkal. 2016. Situation entity types: automatic classification of clause-level aspect. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1757–1768, Berlin, Germany. Association for Computational Linguistics.

Annemarie Friedrich, Nianwen Xue, and Alexis Palmer. 2023. A kind introduction to lexical and grammatical aspect, with a survey of computational approaches. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 599–622, Dubrovnik, Croatia. Association for Computational Linguistics.

Richard Futrell, Kyle Mahowald, and Edward Gibson. 2015. Large-scale evidence of dependency length minimization in 37 languages. *Proceedings of the National Academy of Sciences*, 112(33):10336–10341.

Chiara Gambi, Jaroslaw R. Lelonkiewicz, and Davide Crepaldi. 2024. Do children (and adults) benefit from a prediction error boost in one-shot word learning? *Journal of Cognition*.

Edward Gibson. 2000. The dependency locality theory: A distance-based theory of linguistic complexity. In Alec P. Marantz, Yasushi Miyashita, and Wayne O'Neil, editors, *Image, language, brain: Papers from the first Mind Articulation Project Symposium*, pages 95–126. MIT Press, Cambridge, MA.

Daniel Gildea and David Temperley. 2010. Do grammars minimize dependency length? *Cognitive Science*, 34(2):286–310.

Jane Gillette, Henry Gleitman, Lila Gleitman, and Anne Lederer. 1999. Human simulations of vocabulary learning. *Cognition*, 73(2):135–176.

Lila Gleitman. 1990. The structural sources of verb learning. *Language acquisition*, 1(1):3–55.

Adam Goodkind and Klinton Bicknell. 2018. Predictive power of word surprisal for reading times is a linear function of language model quality. In *Proceedings of the 8th Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, pages 10–18.

Petra Hendriks. 2016. *Asymmetries between Language Production and Comprehension*. Springer, Cham.

Xuhui Hu. 2024. Bundling telicity, verbal quantification, and perfective aspect: A study on le in yixing chinese. *Journal of Linguistics*, 60:325–361.

Songhee Kim, Jeffrey R. Binder, Colin Humphries, and Lisa L. Conant. 2024. Decomposing unaccusativity: A statistical modelling approach. *Language, Cognition and Neuroscience*, 39(9):1189–1211.

Angelika Kratzer. 2005. Building resultatives. In Claudia Maienborn and Angelika Wöllstein, editors, *Event Arguments: Foundations and Applications*, pages 177–212. Max Niemeyer Verlag, Berlin and Boston.

Manfred Krifka. 1998. The origins of telicity. In Sally Rothstein, editor, *Events and Grammar*, pages 197–235. Kluwer Academic Publishers, Dordrecht.

Tom Kwiatkowski, Sharon Goldwater, Luke Zettlemoyer, and Mark Steedman. 2012. A probabilistic model of syntactic and semantic acquisition from child-directed utterances and their meanings. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 234–244, Avignon, France. Association for Computational Linguistics.

Beth Levin and Malka Rappaport Hovav. 1994. *Unaccusativity: At the Syntax–Lexical Semantics Interface*. MIT Press, Cambridge, MA.

Roger P. Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

David W. Lightfoot. 2017. *Transparency*, pages 322–337. Cambridge Handbooks in Language and Linguistics. Cambridge University Press.

Chien-Jer Charles Lin. 2003. Aspect is result: Mandarin resultative constructions and aspect incorporation. In *Proceedings of the Western Conference on Linguistics (WECOL 2003)*, volume 15, pages 437–464.

Haitao Liu, Chunshan Xu, and Junying Liang. 2017. Dependency distance: A new perspective on syntactic patterns in natural languages. *Physics of Life Reviews*, 21:171–193.

Bolei Ma and Yusuke Miyao. 2026. The imperfective paradox in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15093–15111, San Diego, California, United States. Association for Computational Linguistics.

J. E. MacDonald. 2008. Domain of aspectual interpretation. *Linguistic Inquiry*, 39(1):128–147.

Jonathan E. MacDonald. 2010. The aspectual influence of the noun: (a)telicity, (a)symmetry, incrementality and universality. *Language and Linguistics Compass*, 4(9):831–845.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*. Lawrence Erlbaum Associates, Mahwah, NJ.

Brian MacWhinney. 2010. Computational models of child language learning: an introduction. *Journal of Child Language*, 37:477–485.

Letitia Naigles. 1990. Children use syntax to learn verb meanings. *Journal of Child Language*, 17(2):357–374.

Jonathan E. Peelle, Ryland L. Miller, Chad S. Rogers, Brent Spehar, Mitchell S. Sommers, and Kristin J. Van Engen. 2020. Completion norms for 3085 english sentence contexts. *Behavior Research Methods*, 52:1795–1799.

Eva Portelance, Yuguang Duan, Michael C. Frank, and Gary Lupyan. 2023. Predicting age of acquisition for children's early vocabulary in five languages using language model surprisal. *Cognitive Science*, 47(9):e13334.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Gilliane Ramchand. 2008. *Verb Meaning and the Lexicon: A First-Phase Syntax*. Cambridge University Press, Cambridge.

Ian G. Roberts. 2010. *Agreement and Head Movement: Clitics, Incorporation, and Defective Goals*. MIT Press, Cambridge, MA.

Cristina Schmitt. 1996. *ASPECT AND THE SYNTAX OF NOUN PHRASES*. Ph.D. thesis, University of Maryland.

Yasuhiro Shirai and Roger W. Andersen. 1995. The acquisition of tense-aspect morphology: A prototype account. *Language*, 71(4):743–762.

Eric V. Siegel and Kathleen R. McKeown. 2000. Learning methods to combine linguistic indicators: Improving aspectual classification and revealing linguistic insights. *Computational Linguistics*, 26(4):595–627.

Carlotta S. Smith. 1991. *The Parameter of Aspect*. Kluwer Academic Publishers, Dordrecht.

Catherine E. Snow. 1972. Mothers' speech to children learning language. *Child Development*, 43(2):549–565.

Chenchen Song. 2018. Severing telicity from result. *Journal of East Asian Linguistics*, 27:275–307.

Rint Sybesma. 1997. Why chinese verb *le* is a resultative predicate. *Journal of East Asian Linguistics*, 6(3):215–261.

Wenkai Tay. 2024. *Resultative Expressions in Mandarin Chinese*. Doctoral thesis, University College London. Open-access via UCL Discovery.

Angeliek van Hout. 1998. On the role of direct objects and particles in learning telicity in Dutch and English. In *Proceedings of the 22nd Annual Boston University Conference on Language Development*, volume 2, pages 397–408, Somerville, MA. Cascadilla Press.

Zeno Vendler. 1967. *Linguistics in Philosophy*. Cornell University Press, Ithaca, NY.

Henk J. Verkuyl. 1972. *On the Compositional Nature of the Aspects*. D. Reidel Publishing Company, Dordrecht.

Laura Wagner. 2001. Aspectual influences on early tense comprehension. *Journal of Child Language*, 28(3):661–681.

Laura Wagner. 2006. Aspectual bootstrapping in language acquisition: Telicity and transitivity. *Language Learning and Development*, 2(1):51–76.

Ethan Gotlieb Wilcox, Jon Gauthier, Jennifer Hu, Peng Qian, and Roger Levy. 2020. On the predictive power of neural language models for human real-time comprehension behavior. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 42, pages 1707–1713.

Ellie Xia and Alan Hezao Ke. 2026. Tonal marking of telicity in hakka: An agree-based analysis. *Proceedings of the Linguistic Society of America*, 11(1):6085.

Jingying Xu. 2026. *Telicity judgments in English and Mandarin: Crosslinguistic and developmental evidence*. Ph.D. thesis, Michigan State University.

Jingying Xu and Cristina Schmitt. 2025a. Demonstratives but not verbs cause non-culmination in mandarin incremental-theme predicates: Evidence from children and adults. In *Proceedings of the 42nd West Coast Conference on Formal Linguistics (WCCFL 42)*.

Jingying Xu and Cristina Schmitt. 2025b. Grammatical versus pragmatic accounts for event non-culmination in Mandarin: An experimental study. In *Proceedings of the 60th Meeting of the Chicago Linguistic Society (CLS 60)*, pages 429–440.

Jingying Xu and Cristina Schmitt. 2025c. Pragmatic accommodation in judging event culmination. *Proceedings of the 34th Semantics and Linguistic Theory (SALT 34)*.

Charles Yang. 2016. *The price of linguistic productivity: How children learn to break the rules of language*. MIT press.

Alessandra Zarcone and Alessandro Lenci. 2008. Computational models of event type classification in context. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Yiyun Zhao, Jian Gang Ngui, Lucy Hall Hartley, and Steven Bethard. 2021. Do pretrained transformers infer telicity like humans? In *Proceedings of the 25th Conference on Computational Natural Language Learning (CoNLL)*, pages 72–81. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*, volume 36, pages 46595–46623.

A Preprocessing and Annotation Statistics

Table 7 reports the size of the CHILDES subsets before and after preprocessing, along with the resulting telicity label distribution from the Difference in Surprisal method. Table 8 reports the label distribution of the 200-sentence expert-annotated sample used for validation, both per rater and against model predictions.

Table 7: Corpus size and label distribution before and after preprocessing.

	Child	**Adult**
Raw utterances	835,151	170,844
Utterances after preprocessing	417,181	106,388
Telic (%)	31%	12%
Atelic (%)	69%	88%

Table 8: Label distribution in the 200-sentence expert-annotated sample (by rater) vs. Model Prediction.

	Telic	**Atelic**
Rater #1	45	155
Rater #2	42	158
Rater #3	46	154
Model Prediction	55	145

Measuring Form and Function in Language Models

Héctor Javier Vázquez Martínez and **Charles Yang**
{hjvm, ycharles}@upenn.edu
University of Pennsylvania
Department of Linguistics and Computer and Information Science

Abstract

We introduce quantitative metrics from child language research to evaluate language models. Our focus is on the formal syntactic and functional discourse properties of determiners in English, which young children acquire early and accurately. We propose *Contextual Alternative Choice* (CAC), a new prompting method which provides targeted tests for both syntactic and discourse knowledge of language. The method enables direct comparison of language models against children, and more importantly, against independently established benchmarks in empirical research. No current model trained on a comparable amount of data simultaneously meets both benchmarks like human children, but some models trained on much more data do. We present our results as methodological and technical contributions, with specific emphasis on the cognitive status of language models.[1]

1 Introduction: LLMs and child language

Central to American structuralist linguistics (Harris, 1951; Chomsky, 1955) is the methodological commitment to *distributionalism*: Grammar can be defined formally from a corpus of text (or phonemes) without reference to semantic content. As articulated in *Syntactic Structure* (Chomsky, 1957, p50), the most important goal of linguistic theory under the distributional approach is a *discovery procedure*: "the theory must provide a practical and mechanical method for actually constructing the grammar, given a corpus of utterances." In light of the remarkable success of LLMs, which are primarily trained on text, a set of questions naturally arises: Do LLMs constitute a discovery procedure? Do they encode linguistic knowledge in ways that are comparable to human language users? To what extent can they be regarded as a computational model of language acquisition by children? The BabyLM Challenge tackles such fundamental problems in the science of language and cognition, which may in turn provide insight on building data-efficient models to serve wider linguistic communities (Choshen et al., 2026).

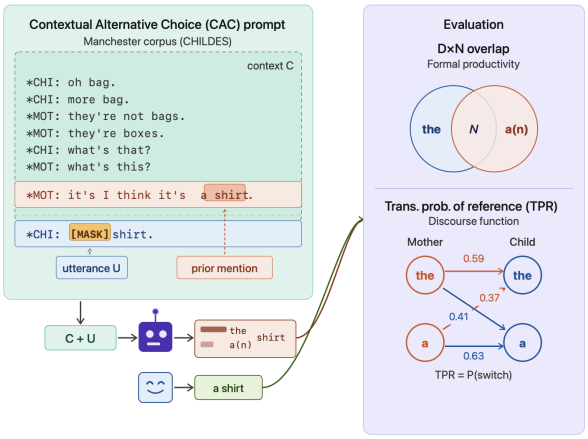


Figure 1: Contextual Alternative Choice (CAC) prompting: A target determiner-noun (D×N) site is identified in caretaker-child dialogue with *the* and *a* as grammatical choices. The model is presented with the full preceding discourse context C and predicts the determiner in target utterance U. The resulting determiner preferences are evaluated against two independently motivated benchmarks: formal productivity (overlap) and discourse-functional appropriateness (Transitional Probability of Reference, TPR) described in Section 2.

To properly approach these questions requires a framework of reference in which child and LLM language learning can be rigorously compared. To do so calls for closer connection with child language research than present. Our starting point is two striking yet underappreciated parallels between language learning by children and LLMs.

First, for very different reasons, the inner mechanisms of human and LLM learning are both poorly understood. There is, quite obviously, very little room for drastic experimentation with human learning. At the same time, LLMs have so far resisted a complete theoretical understanding despite invasive

[1] Our evaluation code and results are available at https://github.com/hjvm/llm-form-and-function.

Proceedings of the Second BabyLM Workshop, pages 438–458
October 29, 2026 ©2026 Association for Computational Linguistics

manipulation of input and model architecture. As such, their external behavior — what children say and what LLMs do — remains the primary source of data for assessing their linguistic capacities. Second, the investigation of children and LLMs both face a data poverty problem. The BabyLM Challenge is an exception but frontier LLMs are generally trained with closed-source data. Similarly, while a reasonable amount of child-directed data exists in the public domain (MacWhinney, 2000), it is amassed from many learners, with no comprehensive record for any particular child.

In this paper, we exploit these parallels between child and LLM learning by introducing a new set of investigative tools from recent developments in language acquisition. Our proposal is both methodological and empirical, which we take up in turn.

1.1 Classification vs. generation

To date, the most popular evaluation for LLMs, including the majority in the BabyLM Challenge, is based on classification of minimal pairs: grammatical vs. ungrammatical sentences, plausible vs. implausible answers in conceptual understanding and reasoning, correct vs. incorrect pairings between images and captions, etc. Classification datasets, which are easy to scale, are naturally suited for LLMs. However, there are well-known difficulties relating classification tasks to human linguistic behavior and knowledge. For example, the syntactic templates in the widely adopted BLiMP benchmark frequently introduces semantic and pragmatic anomalies (Warstadt et al., 2020). However, the influence of these factors on acceptability judgment does not appear simply additive to syntax, even for binary tasks (Sprouse et al., 2018), making it difficult to relate LLM performance to human behavior (Vázquez Martínez et al., 2023).

The classification task is even more problematic in the cognitive setting of BabyLM: Quite simply, there is no ground truth from child language to compare LLMs against. It has long been familiar to researchers that young children cannot provide reliable metalinguistic judgments (De Villiers and De Villiers, 1974; Thornton, 2021). Indeed, most methods readily applicable to adults cannot extend to children. For example, the celebrated Wug test (Berko, 1958), the most direct test for linguistic productivity (*wug-wugs*), can only reliably elicit correct responses from first graders; the plural rule, however, is already learned by age 2 (Brown, 1973). The gap is due to the heavy metalinguistic demands required for learning and using novel words in lab.

For these reasons, child language research typically uses naturalistic production data to benchmark language development. For example, the size of children's vocabulary, the simplest component of language, is estimated by only including words that children produce rather than what they (apparently) understand (Fenson et al., 1994). Similarly, the major landmarks of morphological and syntactic acquisition are all established on children's naturalistic production, ranging from the number of morphemes and words per utterance (e.g., mean length of utterance, "one/two word stage") to overgeneralization errors (e.g., *foots*, *goed*, *I died you*) that mark the productivity of rules (Ervin, 1964; Bowerman, 1982). The gold standard for assessment is Brown (1973)'s 90% criterion, which captures both the formal and functional aspects of language: Acquisition is deem successful if children produce adult-appropriate output in at least 90% of obligatory contexts. Originally established on child English, the criterion has been widely adopted in cross-linguistic research (Slobin, 1985).

1.2 Beyond well-formedness

In recent years, however, Brown's criterion has been under scrutiny. As argued by theorists such as Tomasello (2000), children's adult-like usage may simply result from memorizing, and retrieving, chunks of adult input of varying sizes rather than a systematic grammar. This should be a familiar refrain in the study of language modeling from n-grams to LLMs: It is difficult to know whether a certain linguistic capacity can be attributed to generalizations from, or regurgitation of the training data itself (McCoy et al., 2023; Merrill et al., 2024).

In this paper, we introduce two recent benchmarks for measuring child language designed to overcome the deficiencies in Brown's 90% criterion. Together, they recognize that well-formedness alone may be too weak of a test for linguistic knowledge. These benchmarks, reviewed in Section 2, compare the statistical profile of a corpus not against the statistical profile of the input but against the *expected* statistical profile of the target *grammar*. As such, they are applicable to any language sample produced by children and LLMs alike, without needing to access input/training data. Our empirical study focuses on English determiners, a richly researched domain that embodies both formal (syntax) and functional (discourse) proper-

ties of child language. In order make direct comparisons between LLMs and children, we draw inspiration from the familiar cloze task (Taylor, 1953) from psycholinguistics, especially its use at the discourse level (Chihara et al., 1977) and introduce a new LLM prompting method to make targeted grammatical choices in identical contexts as children (Section 3). We use this framework to evaluate determiner use by two-year-olds and also by 47 language models (Section 4) in the BabyLM Challenge. No model trained on a comparable amount of input data shows equivalence to the human baseline on both benchmarks; only two models, both trained on far larger data, show no evidence of divergence on both. In Section 5 and 6, we discuss related work and the implication of our work in the broader context of LLM research, with specific emphasis on their status as cognitive models of language.

2 The statistics of determiners

Determiners provide an excellent window into language acquisition. As functional words, determiners' role in language is largely formal, lacking the referential salience and semantic concreteness of content words (nouns, verbs, adjectives, etc.). Nevertheless, determiners are identified as a formal category in infancy and play an important role to establish content word categories such as nouns; see Shi (2023) for review. Determiners also appear early and frequently in child language production and have proven a fertile ground of research.

2.1 Measuring determiner productivity

Children's determiner usage is essentially error-free from the get-go (Valian, 1986): they must have learned an adult-like grammar by Brown's 90% criterion. This conclusion has been challenged in recent years (Tomasello, 2000). Adult-like usage may result from memorization and retrieval of input, which would be lexically specific rather than genuinely productive. For example, the English determiners *the* and *a*[2] are productive because they can interchangeably combine with any singular noun given suitable context. A widely adopted numerical measure, *overlap*, has been used to quantify determiner-noun (henceforth D×N) productivity (Pine and Martindale, 1996). Given a corpus, overlap is the proportion of noun types that appear with both *the* and *a* out of those appearing with either *the* or *a*. The overlap value in child language corpora is usually quite low (20-40%), and considerably lower than their caretakers, leading to claims that children do not have a productive determiner grammar but only a collection of lexically specific collocations from the input.

The overlap measure has generated a large body of quantitative tests for children's grammar but almost all rely on comparisons with, or statistical analysis of, caregiver data to determine the extent of memorization; see Section 5 for details. As such, these methods are inherently limited by the available input data, which at best is only a tiny fraction of the totality. By contrast, a formal statistical test (Yang, 2013) overcomes this obstacle. It derives the expected overlap value generated by a productive grammar, i.e., one that combines D×N independently. The test builds on two key statistical properties of language, one universal and the other specific to D×N in English. First, the frequencies of nouns are assumed to follow Zipf (1949)'s Law. As such, if a corpus of S D×N combinations involves N unique nouns, the expected frequency of the noun of rank r is $S/(rH_N)$ where $H_N = \sum_{i=1}^{N} 1/i$. Second, in D×N combinations, nouns tend to have a "favorite" (more frequent) determiner. For example, *bathroom* greatly favors *the* over *a* but for *bath*, the reverse is true. This imbalance, referred to as *bias* (b), is quantified in the aggregate as follows:

$$b = \frac{\sum_{n \in N} \max(C_{\text{the}\times n},\ C_{\text{a}\times n})}{\sum_{n \in N} (C_{\text{the}\times n} + C_{\text{a}\times n})} \quad (1)$$

where $C_{\text{the/a}\times n}$ is the frequency of *the*/*a* combined with n in the corpus. Children do not need to learn these frequencies: they presumably do not track personal hygiene (*a bath*) or bodily functions (*the bathroom*). Rather, the bias value reflects the vagaries of life in language use, and is thus remarkably stable in the aggregate: 0.82 in the billion-word COCA corpus (Davies, 2008), almost identical to the values in the child and caretaker speech analyzed in Section 4.

These two statistical properties greatly enhance the applicability of the productivity test. For a corpus, one only needs S, N, and b to calculate the expected value of overlap; see Appendix A for details. If a comparison between expected and empirical overlap values from a corpus does not

[2]Following the practice in child language, the phonological variant *an* is merged with *a* in the present study.

show a significant difference, one can conclude that the corpus is consistent with a productive grammar.

2.2 Measuring determiner reference

As emphasized in Section 1, linguistic knowledge is more than well-formedness in classification tests. To fully acquire determiners, it is not sufficient to know they freely combine with nouns: their communicative functions are equally important. The determiners *a* and *the* differ in their referential properties (definiteness, specificity, etc.) in context (Lyons, 1999); these functional roles have also been studied in language acquisition. Three-year-olds can interpret determiner reference in context with high accuracy (85%; Maratsos 1976). Analyzing children's naturalistic production, Rozendaal and Baker (2008) conclude that two-year-olds already show adult-like determiner usage with respect to specificity, definiteness and givenness. However, these authors also note that interpreting children's determiner usage requires a reconstruction of the child's mental model of discourse, which is both subjective and laborious.

Gleitman and Yang (2022) propose a statistical measure of determiner function that eliminates interpretative guesswork. They focus on contexts where one speaker uses a determiner in response to a determiner used by their interlocutor when discussing the same topic noun: In general, only one of two options is acceptable. The following two snippets of conversation are instances where the child follows the caretaker with the same determiner; changing it would be highly unnatural albeit formally grammatical.

Mother: Is it **the dog** or the little boy?
Child: **The dog** won't stand up properly.
Mother: I'll make you **a gate**.
Child: Found one. Found **a gate**.

By contrast, the following are instances of determiner change: retaining the same determiner is infelicitous.

Mother: Have you ever had **an itch**?
Child: That's **the itch** over there.
Mother: What happens in **the carwash**, John?
Child: Train's had **a carwash**.

Gleitman and Yang (2022) introduce a statistical measure of determiner discourse function: The *transitional probability of reference* (TPR) is the probability with which a speaker *changes* determiners following an interlocutor while holding the topic noun constant, which can be represented as state transitions in Fig 1. For example, in the four transitions above, TPR is 0.5. Children's TPR when talking to caretakers can be compared against adults' TPR when talking to each other, which serves a baseline benchmark. A baseline is motivated by the fact that the frequency of communicative needs (e.g., retaining or shifting reference for definiteness, specificity, etc.) is largely invariant (Givón, 1983) even when the grammar expressing these functions may change (Komen et al., 2014). Once again, the absence of statistical difference between child and adult TPRs indicates adult-like command of determiners with respect to discourse functions.

3 Methodology

Our main methodological contribution is a prompting scheme that turns child-caretaker dialogue into controlled tests of linguistic knowledge in LLMs. For each target utterance U produced by children, we construct a discourse context C consisting of the preceding dialogue in the recording session, truncated only by the model's context window. Within U, we identify the target position of a grammatical choice and elicit determiner preferences from the model — *the* or *a* — while preserving all surrounding contexts including the noun that the determiner combines with, hence *Contextual Alternative Choice* (CAC) prompting (see Fig 1).

Our empirical study is based on the Manchester Corpus (Theakston et al., 2001) which contains regular recording sessions of 12 mother-child dyads aged 2-3, when determiners begin to be systematically used in speech. Extrapolating from estimates of child-directed input (Gilkerson et al., 2017), these children will have heard approximately 10-20 million words in total. It is, however, more likely that the effective input is only 5-10 million words as the first year is primarily phonological development. These quantities provide useful checkpoints for LLMs as models of child acquisition.

The basic workflow of our study is illustrated in Fig 1. Both caretaker and child D×N pairs are extracted from the Manchester Corpus. The caretaker data is necessary for establishing the dialogue structure for the TPR analysis. Furthermore, because the caretaker's linguistic knowledge is not in doubt, their data is also evaluated against the formal and functional benchmarks to provide a robustness test. For each D×N pair in a child utterance U, a language model undergoes CAC prompting with

the context C, defined above, to elicit determiner preferences under the same condition as the child. This way, we obtain a model-generated corpus that parallels the child-produced corpus: at each D×N site, the child has produced a particular determiner (*the* or *a*) while the model has produced a probabilistic distribution over them, which is then used to analytically derive its empirical predictions (Section 3.2). Both child and model data are evaluated with our formal and functional benchmarks.

Finally, we apply the benchmarks to the traditional forced-choice task by choosing determiners according to the model's maximum likelihood determiner (MLE) at each D×N site, in line with BLiMP and other such minimal pair tasks (Section 1). To that end, we also report the extent to which a model's majority determiner choices agree with children's actual determiner choices, which previous work have shown to be adult-like. While we cannot exhaustively examine model output, strong divergence from child's usage may indicate something amiss.

3.1 Model inference

We evaluate 47 language models spanning three architecture classes: masked language models (MLM), autoregressive (AR) models, and encoder-decoder (seq2seq) models. These are the BabyLM baselines, winning submissions, and reference models publicly available as of May 2026. Full model metadata, including parameter counts, nominal training budgets, and estimated total training exposure are reported in Appendix B, including whether the Manchester Corpus is known or likely to be included in training.

For each model, we apply CAC to obtain the logits of *the* and *a* at the site of each D×N combination in children's production, which are converted to a probabilistic distribution. Due to the differences in model architecture, there is some variation in model inference. For MLMs, we replace the target determiner with a single mask token and read off the logits at that position. Because all MLMs in our evaluation tokenize the determiners as single tokens, no multi-mask setup is required. Consistent with the child language studies, the phonological variant *an* is merged into *a* as a single choice.

For AR models, we explicitly construct and score three full candidate utterances: one with *the*, one with *a*, and one with *an* at the target position. Let $U^{(\mathrm{the})}$, $U^{(\mathrm{a})}$, and $U^{(\mathrm{an})}$ denote these variants. Each candidate is scored by its conditional probability under the shared discourse context:

$$p_\theta(U^{(k)} \mid C) = \prod_{t=1}^{T} p_\theta(w_t^{(k)} \mid C, w_{<t}^{(k)}),$$

where $k \in \{\mathrm{the}, \mathrm{a}, \mathrm{an}\}$. Again we merge $U^{(\mathrm{a})}$ and $U^{(\mathrm{an})}$ as a single choice.

For seq2seq models with span-corruption objectives, we use the standard sentinel-token interface and score three teacher-forced target sequences corresponding to the same determiner alternatives. The encoder input replaces the target determiner with a sentinel token, and the decoder assigns

$$p_\theta(y^{(k)} \mid X) = \prod_{t=1}^{T'} p_\theta(y_t^{(k)} \mid \mathrm{Enc}(X), y_{<t}^{(k)}),$$

where X is the encoder input with the determiner replaced by the sentinel and $k \in \{\mathrm{the}, \mathrm{a}, \mathrm{an}\}$. Finally, *a* and *an* are merged into a single choice.

3.2 Model evaluation

Because the CAC prompting produces a probabilistic distribution of *the* and *a* for each D×N site in the model-generated corpus, it is possible to analytically derive the expected values of D×N overlap and TPRs. We discuss these in turn.

Formal D×N productivity For each child corpus which has N unique noun types and S D×N combinations, CAC prompts a model to generate a distribution over (*the*, *a*) at each D×N site. Let k_n be the number of D×N sites for a noun type n, and $p_{n,i}$ be the probability of *the* at site i where $i = 1, 2, \ldots k_n$. The expected value of overlap $\widehat{O}$ for a model-generated corpus is the probability of a noun attested with both determiners averaged over all noun types:

$$\widehat{O} = \frac{1}{N} \sum_{n \in N} \hat{o}_n, \;\; \hat{o}_n = 1 - \prod_{i=1}^{k_n} p_{n,i} - \prod_{i=1}^{k_n} (1 - p_{n,i}) \tag{2}$$

Where $\hat{o}_n$ is the probability that noun n is attested with both determiners across its k_n sites, obtained by subtracting the two complementary events—the noun occurring only with *the* (probability $\prod_i p_{n,i}$) or only with *a* (probability $\prod_i (1 - p_{n,i})$)—from one. For $k_n = 1$, $\hat{o}_n = 0$ regardless of the model's probability, since a single site can produce only one determiner.

The expected value of bias $\hat{b}$ for the corpus can also be derived analytically. It is computed at the

noun level (cf. Eq 1): for each noun type n, we sum the expected counts of *the* and *a* across all k_n sites and take the larger sum as that noun's contribution to the preferred-determiner count:

$$\hat{b} = \frac{1}{S} \sum_{n \in N} \max \left(\sum_{i=1}^{k_n} p_{n,i}, \sum_{i=1}^{k_n} (1 - p_{n,i}) \right) \quad (3)$$

Together, N, S, and $\hat{b}$ determine the predicted overlap under a fully productive grammar (Eq. 8; Appendix A). These values will be compared against the expected empirical value $\widehat{O}$; the absence of significant difference supports productivity.

Determiner transitions. TPR calculation takes place over a pair of determiner transitions $t = (C_t, M_t)$ sharing a topic noun, with C_t produced by the caretaker and M_t generated by the model. Here, C_t is either *the* or *a* but the model choice M_t is a probabilistic distribution (M_t^{the}, M_t^a). The expected value of TPR over all T transitions is:

$$\widehat{\text{TPR}} = \frac{1}{T} \sum_{t \in T} \left(M_t^{the} P(C_t = \text{a}) + M_t^a P(C_t = \text{the}) \right) \quad (4)$$

Model TPR is compared against children, caretakers, and a baseline from adult-to-adult conversations; see Section 4 for details. The comparison asks whether, in the statistical aggregate, a model uses determiners across conversational turns in a manner similar to human language users.

4 Results

We begin by testing human data against the benchmarks before turning to the models: both are summarized in Table 1.

Formal D×N productivity. In the Manchester corpus, children's mean empirical overlap was 0.259 (SD = 0.066) against the mean expected value of 0.251 (SD = 0.112), and caretakers' mean empirical overlap was 0.307 (SD = 0.040) against the mean expected value of 0.328 (SD = 0.071). A paired t-test found no evidence of divergence between empirical and expected values for either group (children: $t(11) = 0.454, p = 0.659$; caretakers: $t(11) = -1.508, p = 0.160$), replicating the findings of Yang (2013) that determiners used by children (and adults) show productivity. The JZS Bayes factor (Rouder et al., 2009) provides positive evidence for equivalence in both groups (children $\text{BF}_{01} = 3.18$; caretakers $\text{BF}_{01} = 1.41$). Full dyad-level statistics are reported in Appendix C.

Note that the average bias value is remarkably similar between children (mean = 0.835, SD = 0.043) and caretakers (mean = 0.816, SD = 0.021): a paired t-test reveals no evidence of divergence between children and caretakers ($t(11) = 2.045, p = 0.066$). One-sample t-tests also found that neither group's mean bias showed evidence of divergence from the universal baseline value of 0.82 established from COCA (Davies, 2008) (One sample t-test: children $t(11) = 1.194, p = 0.258$; caretakers $t(11) = -0.639, p = 0.536$). This confirms the similarity and stability of usage across English and in turn supports the robustness of the benchmark.

The caretakers' overlap values are considerably higher than the children's (paired t-test: $t(11) = -3.484, p = 0.005$). Previous analyses (Pine and Martindale, 1996; Pine et al., 2013) regarded the difference as children's reduced productivity. However, the difference is due to adults talking more and having more opportunities for the nouns to be used with both determiners. This is reflected in the token/type ratio (S/N), which is higher for caretakers than for children (children: mean $S/N = 4.460$; caretakers: mean $S/N = 6.440$) and is strongly correlated with empirical overlap in both groups (children: $r(10) = .904, p < .001$; caretakers: $r(10) = .729, p = .007$).

Discourse-functional benchmark (TPR). We first established the TPR baseline of 0.215 by examining 1,615 possible determiner transitions from adult-to-adult dialogue in CHILDES. For the 12 children in the Manchester Corpus, the mean TPR when interacting with caretakers is 0.226 (SD = 0.050). For the 12 caretakers, the mean TPR when interacting with children is 0.200 (SD = 0.046), see Table 1. The children's TPR values show no evidence of divergence from their caretakers' on a paired t-test ($t = 2.061, p = 0.064$) even though caretakers sometimes repeat child speech verbatim. Neither children nor caretakers show evidence of divergence from the adult-to-adult baseline (One sample t-test: children $t = 0.724$, $p = 0.484$, $\text{BF}_{01} = 2.78$; caretakers $t = -1.198$, $p = 0.256$, $\text{BF}_{01} = 1.93$), providing positive Bayesian evidence for equivalence with the adult population. The statistical results suggest that the distribution of determiners in discourse is similar for children and adults alike, supporting earlier research that children use determiners appropriately.

Table 1: Human and representative model results (compact subset of Table D1; visualized in Figure 2). Bias, empirical overlap, predicted overlap and TPR are macro-averaged across the 12 Manchester child dyads (SDs in parentheses). The D×N test column reports the paired t-test between per-dyad empirical and predicted overlap; the TPR test compares per-dyad model TPR to the adult-to-adult CHILDES baseline of 0.215. Checkmarks indicate no evidence of divergence from the human benchmark ($p > 0.05$). Determiner accuracy is the micro-averaged argmax agreement with the child's actual choice, pooled across sites.

Model	Arch.	Bias	Empirical overlap (SD)	Predicted overlap (SD)	TPR (SD)	DxN test	TPR test	Accuracy
Children	Human	0.835	0.259 (0.066)	0.251 (0.112)	0.226 (0.050)	✓ **(0.659)**	✓ **(0.484)**	–
Caretakers	Human	0.816	0.307 (0.040)	0.328 (0.071)	0.200 (0.046)	✓ **(0.160)**	✓ **(0.256)**	–
ltg-bert-bnc	MLM	0.783	0.314 (0.068)	0.287 (0.107)	0.225 (0.048)	✓ **(0.138)**	✓ **(0.505)**	0.866
roberta-base	MLM	0.776	0.323 (0.071)	0.292 (0.112)	0.239 (0.051)	✓ **(0.096)**	✓ **(0.142)**	0.853
t5-base	S2S	0.753	0.347 (0.062)	0.304 (0.102)	0.239 (0.042)	× (0.030)	✓ **(0.085)**	0.846
babylm-baseline-100m-gpt-bert-causal-focus	AR	0.782	0.314 (0.068)	0.288 (0.106)	0.247 (0.047)	✓ **(0.122)**	× (0.041)	0.898
gpt-bert-babylm-small	MLM	0.902	0.185 (0.054)	0.185 (0.091)	0.431 (0.087)	✓ **(0.997)**	× (<0.001)	0.626
opt-125m	AR	0.756	0.346 (0.067)	0.303 (0.104)	0.251 (0.044)	× (0.027)	× (0.018)	0.845
gpt2	AR	0.752	0.349 (0.069)	0.305 (0.104)	0.252 (0.042)	× (0.031)	× (0.012)	0.835
roberta-base-strict-2023	MLM	0.834	0.301 (0.067)	0.249 (0.086)	0.475 (0.062)	× (0.001)	× (<0.001)	0.585

LLM results. Table 1 summarizes representative model results compared to the human baselines across the two evaluation criteria, with the full results for all 47 models reported in Table D1. Figure 2 visualizes the representative models across all three diagnostic dimensions.

Only two (`ltg-bert-bnc` and `roberta-base`) models showed D×N overlap and discourse-appropriate usage consistent with a productive grammar. Both models, however, are trained on a much larger amount of the data (100M and 30B words) than available to 2-3 year olds. The `t5-base` model, trained on even more data (156B words), also passes the TPR test but fails the overlap test. Under the Bayes factor analysis, only `ltg-bert-bnc` provides positive evidence for equivalence on both dimensions ($BF_{01} = 1.27$ on D×N; 2.84 on TPR), and its D×N value falls just below the caretaker baseline. No BabyLM-scale model trained on developmentally plausible data provides Bayesian evidence for equivalence with the human baseline on either benchmark. Bayes factors for all evaluated models are reported in Appendix D (Table D1).

As shown in Table D1, 30 models showed no evidence of divergence on the formal overlap test, including several models such as `ltgbert-10m-2024` and `gpt-bert-small` that were trained on an amount of data (10M words) comparable to that for 2-3 year olds. Setting aside four models whose outputs were degenerate due to always predicting the same determiner, this is an impressive result: Language models can learn the formal productivity of determiners and apply them interchangeably to nouns. However, all of these models diverge from the discourse-functional benchmark, with their TPRs typically much higher: 0.433 for `ltgbert-10m-2024` and 0.431 for `gpt-bert-babylm-small`, compared to human speakers at 0.215. It is as if the models choose determiners stochastically with no regard for referential restrictions, as shown in the snippets of caretaker-child conversation.

Finally, we also evaluated the maximum likelihood determiner choices at each D×N site by the models. Doing so brings evaluation in line with classification tasks, and also allows us to measure model accuracy in comparison to children. Several models achieved high accuracy: for example, `babylm-baseline-100m-gpt-bert-causal-focus` and `babylm-baseline-100m-gpt-bert-mixed` match child choices at 0.898 and 0.889, respectively, yet neither passed the TPR test, and only narrowly satisfied the overlap test. This supports our earlier remark (Section 1) that classification is not a rigorous demonstration of adult-like knowledge.

5 Related Work

Productivity test The overlap measure has generated several quantitative assessments of child language. As noted earlier, they all require input data, which is available in very limited ways. For example, Alhama et al. (2024) mark determiner productivity as the recording session where children use both determiners with at least two unique nouns. That, however, does not rule out the possibility of them having memorized them from prior but unrecorded input data. Similarly, Meylan et al. (2017) estimate determiner probabilities for specific nouns, including the idiosyncratic *bathroom* and *bath*, to infer the degree of productivity. Due to the sparsity of the input data, the inference model fails to converge for the majority of cases under study. Valian

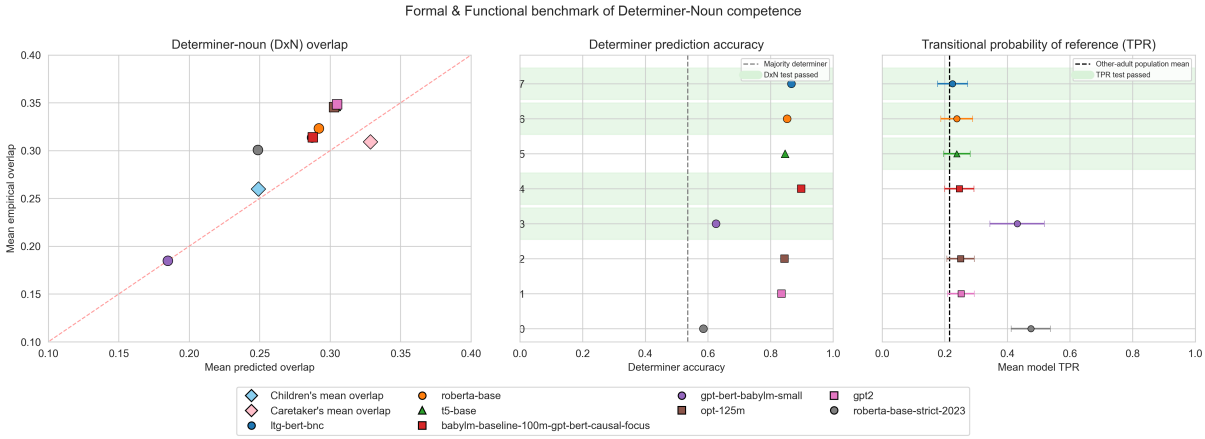


Figure 2: Formal and functional benchmarks for the representative models in Table 1. **Left:** mean empirical vs. predicted D×N overlap (macro-averaged across the 12 dyads), with the dashed diagonal indicating identity and children/caretakers shown for reference. **Middle:** determiner prediction accuracy, computed as the micro-averaged argmax agreement with children's determiner choices pooled across all valid D×N sites. Green shading marks models with no evidence of divergence from the expected D×N overlap, highlighting the dissociation between token-level accuracy and formal productivity. **Right:** mean TPR macro-averaged across the 12 dyads, with error bars showing SD across dyads; the dashed line marks the adult-to-adult CHILDES baseline (0.215), and green shading marks models with no evidence of divergence from the adult baseline.

et al. (2009) and Pine et al. (2013) compare children and caretakers directly and interpret a higher overlap value as greater productivity. But the overlap value itself can be misleading. As Yang (2013) observes, the overlap value of the Brown Corpus (Kučera and Francis, 1967), a collection of print materials, is lower than some children (Pine and Martindale, 1996) but professional writers clearly do not have a worse grammar than two-year-olds.

CAC and LLM inference CAC prompting adapts the cloze task (Taylor, 1953), which has previously served as a behavioral probe of language models across domains, from morphosyntactic analysis in context (Cotterell et al., 2018; McCarthy et al., 2019) to cross-modal brain decoding (Zou et al., 2022). CAC departs from prior cloze-style probes in two respects. First, it restricts the choice set to a closed grammatical alternation rather than an open vocabulary, motivated by recent large-scale comparisons showing that LM next-token distributions and human cloze completions are not interchangeable (Jacobs et al., 2024). Second, it is inspired by discourse cloze tasks in psycholinguistics (Chihara et al., 1977) and NLP (Paperno et al., 2016), rather than in the local sentence alone as in previous LLM work on determiners (Alhama et al., 2023; Fiandra et al., 2025). It accentuates the recognition that language must accurately fulfill communicative functions beyond the well-formedness of sentences.

The probability distributions elicited by CAC are then treated as direct behavioral evidence, similar to a framework for LM surprisal analysis. Surprisal theory (Levy, 2008) establishes that the negative log-probability of a word in context is an interpretable signal of the model's own linguistic expectations, a link that appears to hold across languages (Wilcox et al., 2023). Model probabilities may thus provide a more reliable measure of linguistic knowledge than verbalized or prompted outputs even in instruction-tuned models (Hu and Levy, 2023; Kauf et al., 2024).

6 Discussion and Conclusion

We introduce two quantitative benchmarks for language beyond classification that draw attention to the functional aspects of language that cannot be accurately assessed by well-formeness measures alone. Indeed, in the TPR analysis, both *the* and *a* are fully grammatical but the accurate choice must take discourse function and structure into account. We have shown that current language models trained on developmentally plausible data do not yet exhibit the knowledge of determiners that English-learning children demonstrate by age 2-3. Some, but not all, models trained on much more data do meet the benchmarks but it is not clear what causes success or failure. The three-way dissociation between determiner prediction accuracy, formal D×N productivity, and discourse-functional determiner use shows that these are different facets

of competence, and that optimizing for one provides no guarantee of acquiring the others.

Beyond the empirical findings, we also make a methodological contribution. The statistical benchmarks require no access to training data and are applicable to any language sample produced by children or models alike. The overlap test in particular has been adapted to study other linguistic phenomena in humans (Goldin-Meadow and Yang, 2017a) and can be extended to LLMs as well. CAC, the prompting method introduced to elicit model output suitable for these benchmarks, is not specific to determiners: it can be applied to any grammatical choice at an attested site in naturalistic dialogue.

More generally, our work calls for closer attention to child language research (Vázquez Martínez et al., 2023). Early acquisition of determiners serves as a checkpoint for LLMs. Other aspects of morphology and grammar are learned later with frequent errors (Ervin, 1964; Chomsky, 1969; Yang, 2016) and can also serve as a constraining factor: LLMs may also learn slowly with comparable errors. Finally, it is well known that the same element of grammar may be learned along different schedules across languages. For example, Italian- and Mandarin-learning children are adult-like in their usage of grammatical subjects around 2 (Valian, 1991; Wang et al., 1992) but English-learning children only manage to reach adult-level by 4 (Bloom, 1970). Similarly, children acquiring English go through years where a finite verb is substituted with an infinitive (e.g., *He eat cookies*) but children acquiring Spanish rarely do so (Legate and Yang, 2007). And so on. Such patterns of development, including variation within and across languages, have been amply documented and should shed light on the status of LLMs as cognitive models, especially with the BabyLM Challenge expanding the range of languages under investigation.

Limitations

Scope of the productivity test. The formal productivity test has additional constraints that limit its generality. First, the closed-class category must have exactly two members: the mathematical derivation of expected overlap is tractable for a binary category such as {*the*, *a*} but becomes intractable with additional members such as *this* and *that*. Second, the test assumes that the two categories combine in a fully interchangeable, statistically independent way. This characterizes determiners and nouns well, but not all grammatical combinations have this property. Not all transitive verbs passivize (*John resembles Bill* cannot become **Bill was resembled by John*), not all dative verbs participate in both the double-object and *to*-dative constructions, and so on. CAC can be deployed for such phenomena, but the formal productivity test cannot straightforwardly follow.

Corpus contamination. Many of the evaluated models were likely trained on corpora that include the Manchester data. Despite our efforts to control for this using raw probability weights, contamination cannot be eliminated entirely, and results for models with known or likely Manchester exposure should be interpreted with appropriate caution.

Compute constraints. A single inference run through all 47 models required approximately 14 days of wall-clock time, which prevented evaluation of models at the 1B parameter scale and above. Larger models, and in particular frontier instruction-tuned and RLHF-trained systems, present an additional methodological consideration beyond scale: instruction tuning and reinforcement learning from human feedback optimize for conversational behavior that may interact with the CAC prompting interface in ways that differ from models trained on language modeling objectives alone. Whether such models meet the formal and functional criteria presented here is the topic of future work.

References

Raquel G. Alhama, Ruthe Foushee, Dan Byrne, Allyson Ettinger, Afra Alishahi, and Susan Goldin-Meadow. 2024. Using computational modeling to validate the onset of productive determiner–noun combinations in English-learning children. *Proceedings of the National Academy of Sciences*, 121(50):e2316527121.

Raquel G. Alhama, Ruthe Foushee, Daniel Byrne, Allyson Ettinger, Susan Goldin-Meadow, and Afra Alishahi. 2023. Linguistic Productivity: the Case of Determiners in English. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 330–343, Nusa Dua, Bali. Association for Computational Linguistics.

Jean Berko. 1958. The child's learning of English morphology. *Word*, 14(2–3):150–177.

Lois Bloom. 1970. *Language development: Form and function in emerging grammar*. MIT Press, Cambridge, MA.

Melissa Bowerman. 1982. Reorganizational process in lexical and syntactic development. In Eric Wanner and Lila R. Gleitman, editors, *Language acquisition: The state of the art*, pages 319–346. Cambridge University Press, New York.

Roger Brown. 1973. *First Language: The Early Stages*. Harvard University Press, Cambridge.

Tetsuro Chihara, John Oller, Kelley Weaver, and Mary Anne Chavez-Oller. 1977. Are cloze items sensitive to constraints across sentences? 1. *Language learning*, 27(1):63–70.

Carol Chomsky. 1969. *The acquisition of syntax in children from 5 to 10.* MIT Press, Cambridge, MA.

Noam Chomsky. 1955. The logical structure of linguistic theory. Manuscript, Harvard University.

Noam Chomsky. 1957. *Syntactic structures*. Mouton, The Hague.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM Turns 4 and Goes Multilingual: Call for Papers for the 2026 BabyLM Workshop. *arXiv preprint*. ArXiv:2602.20092 [cs].

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Mark Davies. 2008. Corpus of Contemporary American English (COCA): 410+ million words, 1990-present. Available at http://www.americancorpus.org/.

Jill G De Villiers and Peter A De Villiers. 1974. Competence and performance in child language: Are children really competent to judge? *Journal of Child Language*, 1(1):11–22.

Susan M. Ervin. 1964. Imitation and structural change in children's language. In Eric H. Lenneberg, editor, *New directions in the study of child language*, pages 163–189. MIT Press, Cambridge.

Larry Fenson, Philip S Dale, J Steven Reznick, Elizabeth Bates, Donna J Thal, Stephen J Pethick, Michael Tomasello, Carolyn B Mervis, and Joan Stiles. 1994. Variability in early communicative development. *Monographs of the Society for Research in Child Development*, pages i–185.

Olivia La Fiandra, Nathalie Fernandez Echeverri, Patrick Shafto, and Naomi H. Feldman. 2025. Large Language Models and Children Have Different Learning Trajectories in Determiner Acquisition. In *Proceedings of the First BabyLM Workshop*, pages 100–108, Suzhou, China. Association for Computational Linguistics.

Jill Gilkerson, Jeffrey A Richards, Steven F Warren, Judith K Montgomery, Charles R Greenwood, D Kimbrough Oller, John HL Hansen, and Terrance D Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American journal of speech-language pathology*, 26(2):248–265.

Talmy Givón. 1983. *Topic continuity in discourse*. John Benjamins Publishing Company.

Lila Gleitman and Charles Yang. 2022. Two-year-olds' referential determiners in discourse. In *Boston University Conference on Language Development*.

Susan Goldin-Meadow and Charles Yang. 2017a. Statistical evidence that a child can create a combinatorial linguistic system without external linguistic input: Implications for language evolution. *Neuroscience and Biobehavioral Reviews*, 81(Part B):150 – 157.

Susan Goldin-Meadow and Charles Yang. 2017b. Statistical evidence that a child can create a combinatorial linguistic system without external linguistic input: Implications for language evolution. *Neuroscience & Biobehavioral Reviews*, 81:150–157.

Zellig S. Harris. 1951. *Methods in structural linguistics.* University of Chicago Press, Chicago.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal. Association for Computational Linguistics.

Jennifer Hu and Roger Levy. 2023. Prompting is not a substitute for probability measurements in large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5040–5060.

Cassandra L. Jacobs, Loïc Grobol, and Alvin Tsang. 2024. Large-scale cloze evaluation reveals that token prediction tasks are neither lexically nor semantically aligned. *arXiv preprint*. Version Number: 2.

Carina Kauf, Emmanuele Chersoni, Alessandro Lenci, Evelina Fedorenko, and Anna A Ivanova. 2024. Log probabilities are a reliable estimate of semantic plausibility in base and instruction-tuned language models. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 263–277, Miami, Florida, US. Association for Computational Linguistics.

Erwin Komen, Rosanne Hebing, Ans Van Kemenade, and Bettelou Los. 2014. Quantifying information structure change in english. In *Information structure and syntactic change in Germanic and Romance languages*, pages 81–110. John Benjamins Publishing Company, Amsterdam.

Henry Kučera and W. Nelson Francis. 1967. *Computational analysis of present-day American English*. Brown University Press, Providence.

Julie A. Legate and Charles Yang. 2007. Morphosyntactic learning and the development of tense. *Language Acquisition*, 14(3):315–344.

Roger Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

Christopher Lyons. 1999. *Definiteness*. Cambridge University Press, Cambridge.

Brian MacWhinney. 2000. *The CHILDES project: Tools for analyzing talk*, 3rd edition. Lawrence Erlbaum, Mahwah, NJ.

Michael Maratsos. 1976. *The use of definite and indefinite reference in young children*. Cambridge University Press, London.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sabrina J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

R. Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. 2023. How Much Do Language Models Copy From Their Training Data? Evaluating Linguistic Novelty in Text Generation Using RAVEN. *Transactions of the Association for Computational Linguistics*, 11:652–670. Place: Cambridge, MA.

William Merrill, Noah A. Smith, and Yanai Elazar. 2024. Evaluating n-Gram Novelty of Language Models Using Rusty-DAWG. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14459–14473, Miami, Florida, USA. Association for Computational Linguistics.

Stephan C. Meylan, Michael C. Frank, Brandon C. Roy, and Roger Levy. 2017. The Emergence of an Abstract Grammatical Category in Children's Early Speech. *Psychological Science*, 28(2):181–192.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 1525–1534.

Julian M Pine, Daniel Freudenthal, Grzegorz Krajewski, and Fernand Gobet. 2013. Do young children have adult-like syntactic categories? Zipf's law and the case of the determiner. *Cognition*, 127(3):345–360.

Julian M Pine and Helen Martindale. 1996. Syntactic categories in the speech of young children: The case of the determiner. *Journal of child language*, 23(2):369–395.

Jeffrey N. Rouder, Paul L. Speckman, Dongchu Sun, Richard D. Morey, and Geoffrey Iverson. 2009. Bayesian t tests for accepting and rejecting the null hypothesis. *Psychonomic Bulletin & Review*, 16(2):225–237.

Margot Isabella Rozendaal and Anne Edith Baker. 2008. A cross-linguistic investigation of the acquisition of the pragmatics of indefinite and definite reference in two-year-olds. *Journal of Child Language*, 35(4):773–807.

Rushen Shi. 2023. Functional categories, determiners, prosody and early child grammar. In D. Armoskaite, S. Armoskaite, and M Wiltschko, editors, *Oxford handbook of determiners*. Oxford University Press.

Dan Slobin. 1985. *The crosslinguistic study of language acquisition*. Lawrence Erlbaum, Mahwah, NJ.

Jon Sprouse, Beracah Yankama, Sagar Indurkhya, Sandiway Fong, and Robert C Berwick. 2018. Colorless green ideas do sleep furiously: gradient acceptability and the nature of the grammar. *The Linguistic Review*, 35(3):575–599.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Herbert S Terrace, Laura-Ann Petitto, Richard J Sanders, and Thomas G Bever. 1979. Can an ape create a sentence? *Science*, 206(4421):891–902.

Anna Theakston, Elena Lieven, Julian Pine, and Caroline Rowland. 2001. The role of performance limitations in the acquisition of verb-argument structure: An alternative account. *Journal of child language*, 28:127–52.

Rosalind Thornton. 2021. Judgments of acceptability, truth, and felicity in child language. *The Cambridge handbook of experimental syntax (Cambridge handbooks in language and linguistics)*, pages 394–420.

Michael Tomasello. 2000. Do young children have adult syntactic competence? *Cognition*, 74(3):209–253.

Virginia Valian. 1986. Syntactic categories in the speech of young children. *Developmental Psychology*, 22(4):562.

Virginia Valian. 1991. Syntactic subjects in the early speech of American and Italian children. *Cognition*, 40(1):21–81.

Virginia Valian, Stephanie Solt, and John Stewart. 2009. Abstract categories or limited-scope formulae? The case of children's determiners. *Journal of Child Language*, 36(4):743–778.

Héctor Vázquez Martínez, Annika Lea Heuser, Charles Yang, and Jordan Kodner. 2023. Evaluating Neural Language Models as Cognitive Models of Language Acquisition. In *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*, pages 48–64, Singapore. Association for Computational Linguistics.

Qi Wang, Diane Lillo-Martin, Catherine T Best, and Andrea Levitt. 1992. Null subject versus null object: Some evidence from the acquisition of Chinese and English. *Language Acquisition*, 2(3):221–254.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Ethan G. Wilcox, Tiago Pimentel, Clara Meister, Ryan Cotterell, and Roger P. Levy. 2023. Testing the predictions of surprisal theory in 11 languages. *Transactions of the Association for Computational Linguistics*, 11:1451–1470.

Charles Yang. 2013. Ontogeny and phylogeny of language. *Proceedings of the National Academy of Sciences*, 110(16):6324–6327.

Charles Yang. 2016. *The Price of Linguistic Productivity: How Children Learn to Break the Rules of Language*. The MIT Press.

George Kingsley Zipf. 1949. *Human Behavior And The Principle Of Least Effort*. Addison-Wesley Press.

Shuxian Zou, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2022. Cross-modal cloze task: A new task to brain-to-word decoding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 648–657, Dublin, Ireland. Association for Computational Linguistics.

A Derivation of the formal D×N productivity test

This appendix gives the derivation of the expected-overlap formula used in the main text for the formal D×N productivity test. Following Yang (2013), the derivation assumes two statistical properties of the corpus: (i) singular noun frequencies follow Zipf's Law, and (ii) nouns may exhibit an aggregate bias toward one determiner over the other.

Noun rank probabilities. Let S be the number of D×N combinations involving N unique noun types. Assuming nouns follow Zipf's Law, the noun of rank r has probability

$$p_r = \frac{1}{r^a H_N} \tag{5}$$

$$\text{where} \quad H_N = \sum_{i=1}^{N} \frac{1}{i}. \tag{6}$$

Determiner bias. See Eq. 1 in the main text.

Expected overlap for a noun of rank r. For a noun of rank r, the probability that it is never observed at all in S draws is

$$(1 - p_r)^S.$$

Now consider the event that the noun is observed, but only with one determiner. Under the aggregate bias b, the probability that a token is either not this noun or is this noun with its favored determiner is

$$1 - (1 - b)p_r = bp_r + (1 - p_r),$$

so the probability that the noun never appears with its disfavored determiner is

$$(bp_r + 1 - p_r)^S.$$

Subtracting the probability that the noun is never observed at all gives the probability that it appears only with its favored determiner:

$$(bp_r + 1 - p_r)^S - (1 - p_r)^S.$$

By the same reasoning, the probability that the noun appears only with its disfavored determiner is

$$((1 - b)p_r + 1 - p_r)^S - (1 - p_r)^S.$$

A noun contributes to overlap if it appears with both determiners at least once. Therefore, the expected overlap contribution of the noun of rank r is

one minus the probabilities of the three complementary events: never observed, observed only with the favored determiner, and observed only with the disfavored determiner. This yields

$$\begin{aligned} E_r = 1 &- (1-p_r)^S \\ &- \left[(bp_r + 1 - p_r)^S - (1-p_r)^S\right] \\ &- \left[((1-b)p_r + 1 - p_r)^S - (1-p_r)^S\right]. \end{aligned} \tag{7}$$

Expected overlap for the sample. The expected overlap for the full sample is the mean of these noun-level expectations over all noun ranks:

$$E[S] = \frac{1}{N}\sum_{r=1}^{N} E_r. \tag{8}$$

Empirical comparison. The empirical overlap statistic used in the main text is the proportion of noun types that occur with both determiners among all noun types that occur with either:

$$\text{empirical} = \frac{1}{|N|}\sum_{n\in N} \mathbb{1}\left[\forall d \in \{\textit{the}, \textit{a}\}, C_{d\times n} > 0\right] \tag{9}$$

The formal productivity test compares this empirical value with the expected value in Equation 8. If the two do not differ significantly, the sample is treated as consistent with a fully productive D×N system.

Scope of the test. Although we apply the test here to English determiner–noun combinations, the logic is more general. As discussed by Yang (2013) and Goldin-Meadow and Yang (2017b), the same framework can be extended to other two-way combinatorial systems in which one category is small and closed, the other is larger and approximately Zipfian, and the question is whether the observed overlap is compatible with productive combination. Yang (2013) also used it to reject productivty in Nim Chimpsky's ASL signs, providing independent support for distributional analysis of videos by his trainers (Terrace et al., 1979).

B Evaluated Models

Tables B1–B3 list the language models evaluated in this study together with metadata relevant to the interpretation of our determiner experiments. Model names are given as HuggingFace repository paths where applicable. For each model, we report its parameter count, nominal training-data budget, the estimated total number of tokens seen during training, and whether the Manchester corpus is known or likely to be included in pretraining. We distinguish between nominal dataset size and total training exposure because several models were trained for multiple passes over relatively small corpora, so the total number of tokens seen during training exceeds the training budget allowance.

Column descriptions. **Dataset size** denotes the nominal size of the training corpus or benchmark track. **Total tokens seen** reports our compiled estimate of total training exposure. **Manchester in pretraining?** indicates whether the Manchester corpus is known to be included in pretraining (*yes*), confirmed absent (*no*), likely but not independently confirmed (*possibly*), or not determinable from available documentation (*unknown*). Models trained on the BabyLM Strict (100M) or Strict-Small (10M) datasets are marked *yes* because both tracks contain Manchester corpus material.

C Human validation on Manchester

As a validation of our preprocessing pipeline, we first applied the same extraction and overlap-computation procedure used throughout the paper to the human Manchester data. Singular noun phrases were identified automatically using the spaCy's `en-core-web-sm` noun chunk extractor (Honnibal and Johnson, 2015), after which we computed determiner–noun counts, empirical overlap, predicted overlap, and TPR for each child and caretaker sample. (Our results and conclusion hold when using the annotation provided by CHILDES.) We apply this same preprocessing pipeline to the language models evaluated in the main experiments.

The resulting raw overlap and TPR statistics are shown in Table C1. These values closely reproduce the formal productivity pattern reported by Yang (2013): in the Manchester corpus, children's empirical D×N overlap is statistically indistinguishable from their predicted overlap values ($t(11) = 0.454, p = 0.659$). This result is consistent with the productivity expected under a fully productive system, and caretaker samples show the same general profile ($t(11) = -1.508, p = 0.160$) despite differences in sample size and vocabulary. The same table also shows that both children and caretakers are statistically indistinguishable from the adult population TPR average (children: $p = 0.484$; caretakers: $p = 0.256$). In this sense,

Table B1: Masked language models evaluated in this study. Full Hugging Face repository paths are provided here; later tables use shortened model names for readability.

Repository / family	Variant	Params.	Dataset size	Total tokens seen	Manchester?
phueb/BabyBERTa	-1	5M	5M words	50M words	No
	-2	5M	5M words	50M words	No
	-3	5M	5M words	50M words	No
lgcharpe/ELC_BERT	_small_baby_10M	24M	10M words	~127B	Yes
	_baby_100M	98M	100M words	~270B	Yes
ltg/ltg-bert	-bnc	98M	100M words	124B	No
	-babylm	98M	100M words	124B	Yes
	-10m-2024	30M	10M words	Unknown	Possibly
	-100m-2024	119M	100M words	Unknown	Yes
BabyLM-community/ babylm-baseline-10m-gpt-bert	-masked-focus	31M	10M words	100M words	Yes
	-mixed	31M	10M words	100M words	Yes
BabyLM-community/ babylm-baseline-100m-gpt-bert	-masked-focus	120M	100M words	1B words	Yes
	-mixed	120M	100M words	1B words	Yes
roberta	-base	125M	~30B	~2T	No
	-large	355M	~30–50B tokens	~2T	No
nyu-mll/roberta-med-small	-1M-1	45M	1M words	~26B tokens	No
	-1M-2	45M	1M words	~2.6B tokens	No
	-1M-3	45M	1M words	~8.1B tokens	No
nyu-mll/roberta-base	-10M-1	125M	10M words	~10B tokens	No
	-10M-2	125M	10M words	~2.6B tokens	No
	-10M-3	125M	10M words	~8.1B tokens	No
	-100M-1	125M	100M words	~26B tokens	No
	-100M-2	125M	100M words	~33B tokens	No
	-100M-3	125M	100M words	~8.1B tokens	No
	-1B-1	125M	1B words	~26B tokens	No
	-1B-2	125M	1B words	~33B tokens	No
	-1B-3	125M	1B words	~520B tokens	No
babylm/roberta-base-strict	-2023	125M	100M words	2B words	Yes
	-small-2023	125M	10M words	200M words	Yes

Table B2: Autoregressive models evaluated in this study. Full Hugging Face repository paths are provided here; later tables use shortened model names for readability.

Repository / family	Variant	Params.	Dataset size	Total tokens seen	Manchester?
ltg/gpt-bert-babylm	-small	30M	10M words	15.6B	Possibly
	-base	119M	100M words	31.25B	Possibly
BabyLM-community/ babylm-baseline-10m-gpt-bert	-causal-focus	31M	10M words	100M words	Yes
	-mixed	31M	10M words	100M words	Yes
BabyLM-community/ babylm-baseline-100m-gpt-bert	-causal-focus	120M	100M words	1B words	Yes
	-mixed	120M	100M words	1B words	Yes
timinar/baby-llama	-58m	58M	10M words	Unknown	Yes
babylm/babyllama	-10m-2024	58M	10M words	~100M words	Yes
	-100m-2024	58M	100M words	~1B words	Yes
facebook/opt	-125m	125M	180B	~180B tokens	Unknown
babylm/opt-125m-strict	-2023	125M	100M words	2B words	Yes
	-small-2023	125M	10M words	200M words	Yes
openai-community/gpt	2	124M	~8B tokens	~10B tokens	Unknown
BabyLM-community/ babylm-baseline	-10m-gpt2	124M	10M words	100M words	Yes
	-100m-gpt2	124M	100M words	1B words	Yes

our results align with the broader use of the overlap framework in Goldin-Meadow and Yang (2017b) and validate the TPR benchmark as well.

We include these statistics here for two reasons. First, they provide a sanity check on our implementation of both overlap analyses. Second, they corroborate our Manchester preprocessing pipeline as the basis for model-side evaluation, since the pipeline recovers the established human baseline on both the formal and functional benchmarks before being applied to BabyLM systems.

D Full model results

Table D1 reports the full model-by-model results for all evaluated systems. It combines the formal D×N productivity results and the functional discourse Transitional Probability of Reference (TPR) results into a single table.

For each model, we report mean empirical overlap and mean predicted overlap across the 12 Manchester child dyads, with standard deviations in parentheses, together with determiner bias, determiner accuracy, the outcome of the formal D×N test, mean TPR, the outcome of the TPR test, and JZS Bayes factors (BF_{01}) quantifying the evidence for equivalence on each test. In the two test columns, the symbol indicates whether the model passed the corresponding criterion, and the exact p-value is given in parentheses in the same cell. The BF_{01} columns give the corresponding Bayes factor, with values above 1 indicating positive evidence for equivalence with the human benchmark.

Models with non-significant differences between empirical and predicted overlap are treated as consistent with formal D×N productivity. Models with non-significant differences under the TPR comparison are considered to have achieved discourse-appropriate functional usage of determiners. The main-text summary table is a compact subset of this full table.

Although four models with bias≥ 0.98 appear to pass the formal D×N test, this is an artifact of degenerate determiner predictions. Concretely, all three BabyBERTa variants and babylm-10m-gpt-bert-mixed (MLM) overwhelmingly predict the same determiner ('a(n)'), as evidenced by the bias value being close to 1, yielding trivially small overlap values rather than substantive evidence of productivity.

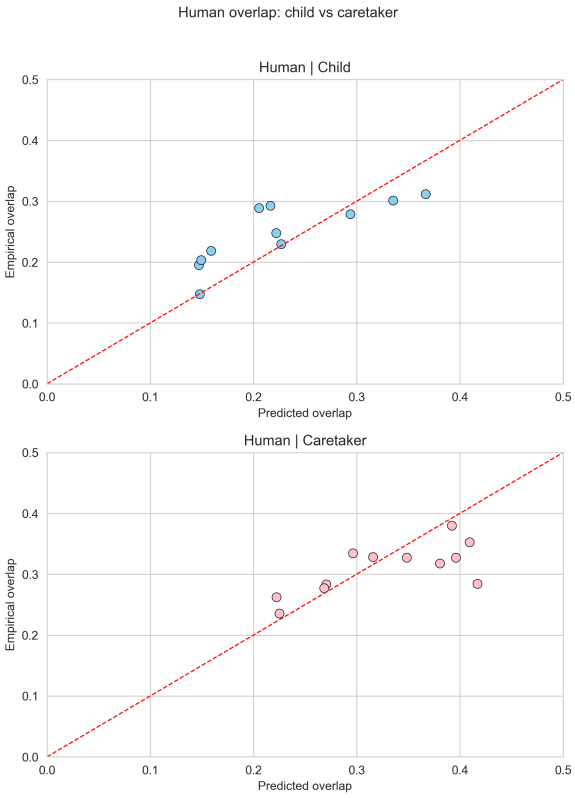


Figure C1: Full formal D×N overlap results for 12 children (top) and their corresponding caretakers (bottom) in the Manchester corpus. No statistically significant difference is found between the empirical and predicted overlap scores (paired t-test $t(11) = 0.454, p = 0.659$ children and $t(11) = -1.508, p = 0.160$ for caretakers), meeting the formal D×N overlap criterion.

Table B3: Encoder–decoder models evaluated in this study. Full Hugging Face repository paths are provided here; later tables use shortened model names for readability.

Repository / family	Variant	Params.	Dataset size	Total tokens seen	Manchester?
`google-t5/t5`	`-base`	220M	156B words	34B tokens	Possibly
`babylm/t5-base-strict`	`-2023`	220M	100M words	2B words	Yes
	`-small-2023`	220M	10M words	200M words	Possibly

Table C1: Types (N), tokens (S), determiner bias score, token/type ratio (r), and observed (empirical) versus predicted overlap values for 12 children and their corresponding caretakers in the Manchester corpus. Singular noun phrases were extracted automatically using spaCy's noun chunker. At the group level, children's empirical overlap does not differ significantly from predicted overlap ($t(11) = 0.454, p = 0.659$), and the same is true for caretakers ($t(11) = -1.508, p = 0.160$), reproducing the formal productivity pattern reported in prior work and validating the preprocessing pipeline used throughout the paper.

Dyad	Speaker	$\lvert N\rvert$	$\lvert S\rvert$	Bias	Empirical	Predicted	n_{TPR}	TPR
Gail	Child	354	1013	0.873	0.192	0.148	224	0.241
	Caretaker	865	3795	0.837	0.258	0.223	286	0.192
Dominic	Child	153	411	0.883	0.157	0.148	172	0.221
	Caretaker	540	4238	0.794	0.287	0.416	188	0.261
Becky	Child	390	1564	0.852	0.282	0.214	438	0.194
	Caretaker	605	3754	0.827	0.321	0.317	548	0.197
Liz	Child	331	1459	0.868	0.254	0.222	373	0.166
	Caretaker	611	3177	0.838	0.278	0.266	410	0.193
Carl	Child	412	4041	0.771	0.400	0.519	910	0.260
	Caretaker	512	3838	0.791	0.350	0.407	1077	0.208
Joel	Child	390	1221	0.889	0.200	0.146	331	0.145
	Caretaker	800	3583	0.840	0.237	0.227	445	0.133
Ruth	Child	223	806	0.813	0.220	0.241	279	0.272
	Caretaker	692	4656	0.803	0.325	0.355	372	0.183
Aran	Child	411	2004	0.795	0.277	0.299	802	0.261
	Caretaker	1038	8432	0.805	0.317	0.390	1079	0.152
Anne	Child	356	1275	0.838	0.292	0.208	500	0.168
	Caretaker	698	6140	0.836	0.378	0.392	657	0.158
John	Child	354	1912	0.792	0.297	0.332	473	0.307
	Caretaker	732	3849	0.791	0.336	0.300	526	0.276
Nicole	Child	261	770	0.865	0.222	0.164	234	0.214
	Caretaker	804	4353	0.835	0.275	0.269	280	0.179
Warren	Child	433	2683	0.779	0.314	0.371	826	0.262
	Caretaker	838	6159	0.797	0.316	0.377	944	0.265

Table D1: Full model-by-model results for all evaluated systems. The table reports determiner bias, mean empirical and predicted D×N overlap across the 12 Manchester child dyads (macro-averaged, SDs in parentheses), mean TPR (macro-averaged), the outcomes of the formal D×N and TPR tests, and determiner accuracy (micro-averaged agreement with children's argmax determiner choices). Checkmarks indicate no evidence of divergence from the benchmark under a paired t-test ($p > 0.05$). The BF_{01} columns report JZS Bayes factors (Rouder et al., 2009) at Cauchy prior scale $r = \sqrt{2}/2$; values above 1 indicate positive evidence for equivalence with the benchmark, values below 1 indicate evidence for divergence. With $n = 12$ dyads, BF_{01} is bounded near ceiling, so human values reflect the strongest evidence achievable in this design.

Model	Arch.	bias	empirical (SD)	predicted (SD)	TPR (SD)	D×N test	BF_{01}	TPR test	BF_{01}	Accuracy
Children	Human	0.835	0.259 (0.066)	0.251 (0.112)	0.226 (0.050)	✓ **(0.659)**	3.18	✓ **(0.484)**	2.78	–
Caretakers	Human	0.816	0.307 (0.040)	0.328 (0.071)	0.200 (0.046)	✓ **(0.160)**	1.41	✓ **(0.256)**	1.93	–
BabyBERTa-1	MLM	0.996	0.014 (0.008)	0.014 (0.007)	0.497 (0.100)	✓ **(0.057)**	0.67	× (<0.001)	5.8e-05	0.580
BabyBERTa-2	MLM	0.999	0.005 (0.003)	0.005 (0.003)	0.497 (0.100)	✓ **(0.445)**	2.66	× (<0.001)	6.0e-05	0.580
BabyBERTa-3	MLM	0.999	0.006 (0.004)	0.006 (0.004)	0.497 (0.100)	✓ **(0.863)**	3.43	× (<0.001)	6.0e-05	0.580
BabyLM-2026-Baseline-GPT2-Strict	AR	0.701	0.381 (0.062)	0.327 (0.096)	0.393 (0.024)	× (0.007)	0.12	× (<0.001)	6.3e-09	0.653
BabyLM-2026-Baseline-GPT2-Strict-Small	AR	0.718	0.375 (0.064)	0.321 (0.102)	0.379 (0.036)	× (0.010)	0.17	× (<0.001)	6.6e-07	0.700
ELC_BERT_baby_100M	MLM	0.799	0.296 (0.081)	0.277 (0.118)	0.299 (0.058)	✓ **(0.308)**	2.16	× (<0.001)	0.01	0.781
ELC_BERT_small_baby_10M	MLM	0.949	0.143 (0.042)	0.116 (0.040)	0.498 (0.090)	× (<0.001)	9.0e-04	× (<0.001)	2.3e-05	0.580
baby-llama-58m	AR	0.741	0.352 (0.061)	0.309 (0.100)	0.271 (0.046)	× (0.025)	0.35	× (0.002)	0.04	0.817
babyllama-100m-2024	AR	0.775	0.319 (0.070)	0.292 (0.108)	0.256 (0.050)	✓ **(0.126)**	1.20	× (0.017)	0.26	0.885
babyllama-10m-2024	AR	0.768	0.327 (0.067)	0.296 (0.107)	0.257 (0.048)	✓ **(0.094)**	0.97	× (0.012)	0.20	0.870
babylm-baseline-100m-gpt-bert-causal-focus	AR	0.782	0.314 (0.068)	0.288 (0.106)	0.247 (0.047)	✓ **(0.122)**	1.17	× (0.041)	0.52	0.898
babylm-baseline-100m-gpt-bert-masked-focus	MLM	0.980	0.062 (0.052)	0.054 (0.041)	0.489 (0.094)	× (0.043)	0.54	× (<0.001)	4.5e-05	0.582
babylm-baseline-100m-gpt-bert-mixed	AR	0.777	0.319 (0.069)	0.291 (0.107)	0.251 (0.046)	✓ **(0.113)**	1.10	× (0.023)	0.33	0.889
babylm-baseline-100m-gpt-bert-mixed	MLM	0.976	0.074 (0.038)	0.067 (0.035)	0.485 (0.095)	× (0.013)	0.21	× (<0.001)	5.4e-05	0.581
babylm-baseline-100m-gpt2	AR	0.729	0.366 (0.075)	0.317 (0.107)	0.405 (0.041)	× (0.013)	0.21	× (<0.001)	5.1e-07	0.726
babylm-baseline-10m-gpt-bert-causal-focus	AR	0.762	0.334 (0.067)	0.300 (0.109)	0.267 (0.048)	✓ **(0.072)**	0.80	× (0.003)	0.07	0.861
babylm-baseline-10m-gpt-bert-masked-focus	MLM	0.957	0.118 (0.054)	0.101 (0.046)	0.488 (0.092)	× (0.003)	0.07	× (<0.001)	3.5e-05	0.581
babylm-baseline-10m-gpt-bert-mixed	AR	0.756	0.340 (0.069)	0.303 (0.107)	0.266 (0.043)	✓ **(0.060)**	0.69	× (0.002)	0.04	0.856
babylm-baseline-10m-gpt-bert-mixed	MLM	0.984	0.052 (0.035)	0.049 (0.031)	0.487 (0.096)	✓ **(0.128)**	1.21	× (<0.001)	5.2e-05	0.582
babylm-baseline-10m-gpt2	AR	0.740	0.348 (0.065)	0.311 (0.108)	0.389 (0.047)	✓ **(0.066)**	0.74	× (<0.001)	5.1e-06	0.725
gpt-bert-babylm-base	MLM	0.916	0.175 (0.059)	0.168 (0.081)	0.441 (0.081)	✓ **(0.560)**	2.98	× (<0.001)	6.4e-05	0.606
gpt-bert-babylm-small	MLM	0.902	0.185 (0.054)	0.185 (0.091)	0.431 (0.087)	✓ **(0.997)**	3.48	× (<0.001)	1.7e-04	0.626
gpt2	AR	0.752	0.349 (0.069)	0.305 (0.104)	0.252 (0.042)	× (0.031)	0.41	× (0.012)	0.20	0.835
ltg-bert-babylm	MLM	0.797	0.297 (0.082)	0.278 (0.119)	0.314 (0.059)	✓ **(0.289)**	2.08	× (<0.001)	4.0e-03	0.774
ltg-bert-bnc	MLM	0.783	0.314 (0.068)	0.287 (0.107)	0.225 (0.048)	✓ **(0.138)**	1.27	✓ **(0.505)**	2.84	0.866
ltgbert-100m-2024	MLM	0.739	0.352 (0.071)	0.311 (0.110)	0.308 (0.057)	✓ **(0.055)**	0.65	× (<0.001)	5.3e-03	0.776
ltgbert-10m-2024	MLM	0.792	0.287 (0.055)	0.280 (0.106)	0.433 (0.059)	✓ **(0.714)**	3.27	× (<0.001)	4.9e-06	0.640
opt-125m	AR	0.756	0.346 (0.067)	0.303 (0.104)	0.251 (0.044)	× (0.027)	0.37	× (0.018)	0.27	0.845
opt-125m-strict-2023	AR	0.812	0.299 (0.060)	0.263 (0.071)	0.430 (0.068)	× (0.001)	0.03	× (<0.001)	2.0e-05	0.500
opt-125m-strict-small-2023	AR	0.745	0.354 (0.067)	0.308 (0.098)	0.368 (0.030)	× (0.014)	0.22	× (<0.001)	2.6e-07	0.697
roberta-base	MLM	0.776	0.323 (0.071)	0.292 (0.112)	0.239 (0.051)	✓ **(0.096)**	0.98	✓ **(0.142)**	1.30	0.853
roberta-base-100M-1	MLM	0.756	0.343 (0.072)	0.303 (0.115)	0.267 (0.050)	✓ **(0.052)**	0.63	× (0.004)	0.08	0.787
roberta-base-100M-2	MLM	0.752	0.343 (0.076)	0.305 (0.117)	0.285 (0.053)	✓ **(0.070)**	0.77	× (<0.001)	0.02	0.796
roberta-base-100M-3	MLM	0.748	0.353 (0.076)	0.308 (0.115)	0.282 (0.051)	× (0.030)	0.41	× (<0.001)	0.02	0.780
roberta-base-10M-1	MLM	0.862	0.225 (0.076)	0.227 (0.113)	0.367 (0.068)	✓ **(0.907)**	3.46	× (<0.001)	4.4e-04	0.702
roberta-base-10M-2	MLM	0.823	0.278 (0.082)	0.260 (0.120)	0.338 (0.055)	✓ **(0.328)**	2.25	× (<0.001)	4.3e-04	0.746
roberta-base-10M-3	MLM	0.836	0.267 (0.081)	0.250 (0.111)	0.347 (0.065)	✓ **(0.296)**	2.11	× (<0.001)	9.8e-04	0.729
roberta-base-1B-1	MLM	0.770	0.331 (0.077)	0.295 (0.119)	0.278 (0.055)	✓ **(0.072)**	0.79	× (0.002)	0.05	0.786
roberta-base-1B-2	MLM	0.747	0.351 (0.076)	0.308 (0.115)	0.283 (0.048)	× (0.031)	0.42	× (<0.001)	0.01	0.797
roberta-base-1B-3	MLM	0.775	0.324 (0.076)	0.292 (0.117)	0.285 (0.059)	✓ **(0.101)**	1.02	× (0.002)	0.04	0.805
roberta-base-strict-2023	MLM	0.834	0.301 (0.067)	0.249 (0.086)	0.475 (0.062)	× (0.001)	0.03	× (<0.001)	1.6e-06	0.585
roberta-base-strict-small-2023	MLM	0.777	0.355 (0.074)	0.290 (0.102)	0.487 (0.056)	× (0.002)	0.04	× (<0.001)	3.6e-07	0.584
roberta-large	MLM	0.766	0.331 (0.071)	0.297 (0.116)	0.270 (0.054)	✓ **(0.111)**	1.09	× (0.005)	0.09	0.832
roberta-med-small-1M-1	MLM	0.754	0.331 (0.059)	0.303 (0.099)	0.475 (0.042)	✓ **(0.149)**	1.34	× (<0.001)	3.4e-08	0.601
roberta-med-small-1M-2	MLM	0.693	0.390 (0.070)	0.332 (0.107)	0.459 (0.019)	× (0.010)	0.17	× (<0.001)	3.4e-11	0.587
roberta-med-small-1M-3	MLM	0.787	0.300 (0.061)	0.284 (0.100)	0.476 (0.051)	✓ **(0.363)**	2.38	× (<0.001)	2.3e-07	0.594
t5-base	S2S	0.753	0.347 (0.062)	0.304 (0.102)	0.239 (0.042)	× (0.030)	0.41	✓ **(0.085)**	0.90	0.846
t5-base-strict-2023	S2S	0.786	0.315 (0.083)	0.286 (0.114)	0.360 (0.067)	✓ **(0.106)**	1.06	× (<0.001)	5.3e-04	0.748
t5-base-strict-small-2023	S2S	0.657	0.420 (0.067)	0.341 (0.090)	0.483 (0.037)	× (<0.001)	0.02	× (<0.001)	7.1e-09	0.436

E Full results figures

Figure E1 visualizes the formal D×N overlap results for the full set of evaluated models, with children and caretakers shown as human reference points. Figure E2 shows determiner prediction accuracy, with models that pass the formal overlap test highlighted. Figure E3 shows mean TPR by model with standard-deviation whiskers, alongside the human child benchmark and models that pass the TPR criterion.

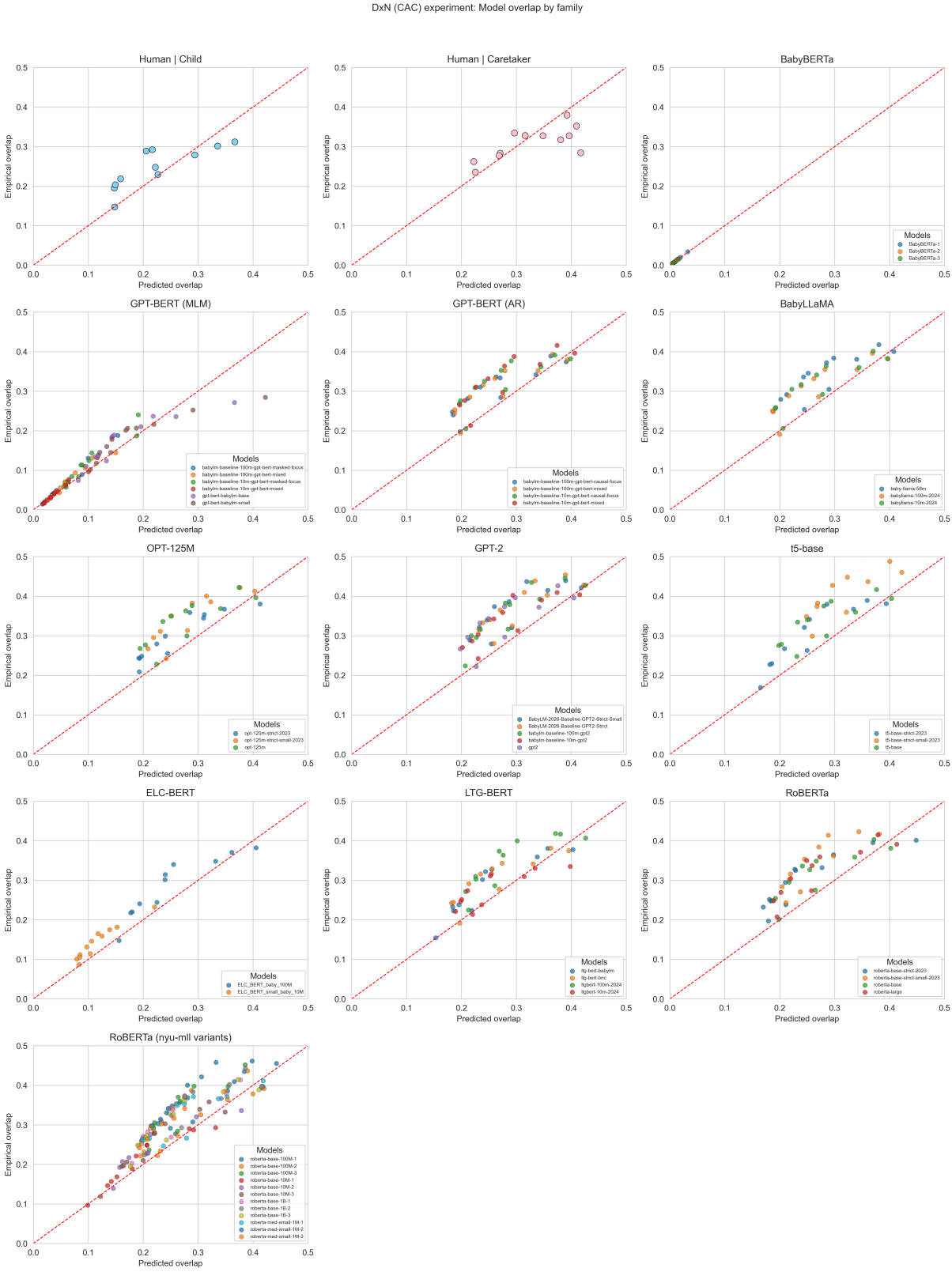


Figure E1: Full formal D×N overlap results for all evaluated models. Each panel shows empirical overlap against predicted overlap for a model family, with the diagonal line indicating equality between observed and predicted overlap. The child and caretaker Manchester dyads are included as human reference panels.



Figure E2: Full determiner prediction accuracy results for all evaluated models. The dashed vertical line indicates the accuracy of a majority-determiner baseline. Models that pass the formal D×N overlap test are highlighted in green.

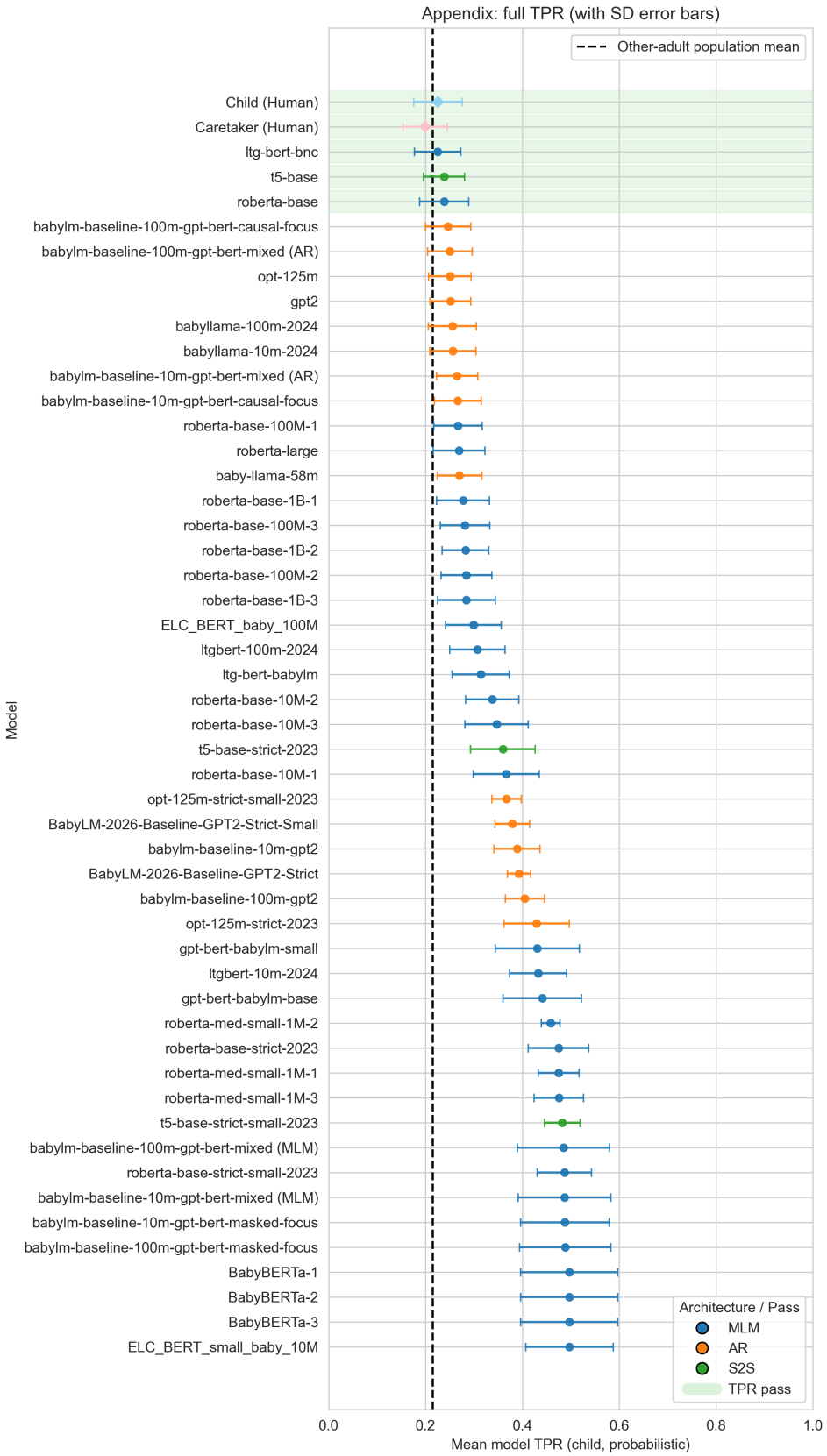


Figure E3: Full Transitional Probability of Reference (TPR) results for all evaluated models, with the 12 Manchester corpus children and corresponding caretakers at the top for reference. Points show mean TPR and whiskers show standard deviations across 12 dyads. The dashed vertical line indicates the adult population mean TPR, and models that pass the TPR test are highlighted in green.

Stories for the Win: Creating and Evaluating a Synthetic Dutch Children's Storytelling Dataset

Sabijn Perdijk, Gijs Wijnholds, Suzan Verberne and M.J. van Duijn
University Leiden

Abstract

Stories have been shown to be potentially valuable for language model pretraining because their semantics are largely self-grounded. Existing human storytelling datasets are relatively small, and existing synthetic datasets lack realism. Here we describe the guided synthesis of a story corpus that is both large and realistic. We evaluate a set of metrics for measuring the quality of individual stories, including coherence, surprise, complexity, as well for quality of the dataset as a whole. Using the metrics we extensively evaluate multiple prompting approaches and three small models (3-8B). Based on the results, we steer dataset generation to yield a high-quality 'synthetic twin' of a Dutch children's storytelling collection, similar in terms of lexical and semantic diversity, but much larger, which we call *Stories for the Win* (StoriesFTW). We validate our metrics and evaluate samples from our corpus with crowd, expert, and LLM judgements.[1]

1 Introduction

Narratives shape society since the earliest beginnings: they explain, educate, and provide purpose to mankind (Koning, 2015). They play a constitutive role in child development, boosting acquisition of communicative skills (Southwood and Russell, 2004), world knowledge (McKeough and Genereux, 2003), and Theory of Mind (Nicolopoulou, 2018). Training of Large Language Models (LLMs) also builds upon narratives: novels and short stories are classic elements of pretraining datasets (e.g. Gao et al., 2020). So far it is largely unclear what exactly the contribution of narrative language is in the LLM pipeline, yet there are empirical indications that training on relatively small amounts of children's stories can be an effective and data-efficient way to extract meaningful word vectors (Van Dijk et al., 2023) or create small language models that are fluent in various domains (Eldan and Li, 2023; Paraskeva et al., 2025). This is in line with theoretical and empirical links between narrative, language learning, and cognitive development more broadly (see §2.1).

To enable further investigation of the role children's narratives play in pretraining LLMs, **we construct a large and realistic synthetic narrative dataset**. In light of Wilcox et al. (2025)'s plea to use more data that resembles children's input, and considering the theoretical and empirical work cited above and in §2.1, we focus on realistic everyday narratives for and by children. Existing child-narrative datasets do not match our criteria. TinyStories, a synthetic dataset by Eldan and Li (2023), is large enough, but its design provides a limited view of child narratives. Children Stories Text Corpus (CSTC) (Bens, 2021), a subset of Project Gutenberg, offers a collection of world literature for children, but the idiosyncratic stories it contains (e.g. fairy tales) mostly do not align with children's daily experiences. Examples of datasets that do align with children's worlds are the English Nicolopoulou Corpus on CHILDES (Nicolopoulou, 2019) and the Dutch ChiSCor (Van Dijk et al., 2023). The Nicolopoulou Corpus contains stories told by and for children in their daily environments, yet from a limited range of age groups and social backgrounds. ChiSCor contains stories from diverse age groups and backgrounds, but with 619 stories and 74k tokens is too small to serve as a pretraining corpus. To combine the qualities of ChiSCor with pretraining scale, we generate a 1000 times larger synthetic twin of this Dutch children's narrative corpus. We do so with a systematic approach to prompt design and model selection, guided by metrics of story quality at different levels. We discuss and test the alignment of the metrics with human judgements. These metrics, partly newly constructed and partly grounded in previ-

[1]Dataset available at https://huggingface.co/datasets/Sabijn/StoriesfortheWin, code available at: https://github.com/sabijn/stories4thewin

Proceedings of the Second BabyLM Workshop, pages 459–478
October 29, 2026 ©2026 Association for Computational Linguistics

ous research, measure coherence, creativity, and complexity at the story-level and overall quality and diversity at the dataset-level. Afterwards, the stories are evaluated with human judgements.

Our contributions are: (1) a set of metrics for measuring the quality of synthetic story data benchmarked on human judgements, which we use for (2) systematic evaluation of models and prompting approaches to steer the creation of (3) a synthetic Dutch children's story dataset delivered for language model training and research.

2 Background

2.1 Storytelling and Child Development

Engaging with stories, either by producing or consuming them, urges the human brain to engage in a form of 'cognitive pretend play' and project an imagined, alternative reality, referred to as the *story-world* (Smith and Roopnarine, 2018; van Duijn, 2025). Objects, characters, and events gain meaning against the background of this imagined story-world, enabling a form of surrogate grounding (van Dijk, 2025). In other words, whereas most forms of discourse are grounded in a context outside the linguistic realm (e.g., a physical setting with its objects, participants, actions, etc.), stories are to a large degree grounded in themselves. Creating and understanding self-grounded discourse helps children develop their language skills (Nicolopoulou, 2019). This self-grounding likely also allows LLMs to learn new words more efficiently (Ma et al., 2023; van Dijk, 2025).

2.2 Evaluating at Narrative and Dataset Level

Training data quality is recognized as a crucial factor in the success of LLMs (Gunasekar et al., 2023; Wilcox et al., 2025). We focus on quality at the level of the individual dataset entries (the stories) and at a global level.

Story level Chhun et al. (2022) define six human criteria for 'good' narratives: *Relevance*, *Coherence*, *Surprise*, *Complexity*, *Empathy*, and *Engagement*. These criteria were used to rank several automatic metrics, and it was concluded that current metrics are insufficient to completely resemble human ratings. We, therefore, propose additional metrics based on their criteria, aiming to better reflect human experience.

Ismayilzada et al. (2024) evaluate creative short story generation in humans and LLMs based on diversity, novelty, surprise, and complexity. Tian et al. (2024) evaluate narrative development and plot progression, concluding that LLMs fall short in discourse understanding.

Another field working with narrative quality is automatic storytelling. Prior research in this field mainly focussed on storytelling based on plot planning (Yao et al., 2019). Current work, related to the *Coherence* criterion, focusses on maintaining narrative order (Yao et al., 2019) or alterations on the narrative order in the form of flashbacks (Han et al., 2022; Lu et al., 2023).

Various evaluation criteria associated with high-quality narrative, such as linguistic complexity, are also applicable to generated text in general. Researchers have released open-source Python packages for measuring and extracting repetition in text (Shaib et al., 2025), benchmarking linguistic diversity in LLMs (Guo et al., 2025), or exploring precision and recall for the diversity of LLMs (Le Bronnec et al., 2024). Additional criteria include novelty (McCoy et al., 2023) and linguistic complexity (Muñoz-Ortiz et al., 2024).

Dataset level Various research lines have focused on measuring the global quality of artificially generated datasets (Li et al., 2023; Chen et al., 2023; Long et al., 2024). Dang and Verma (2024) have sought to create a taxonomy and standardize metrics, proposing five categories for measuring overall dataset quality: linguistic, semantic, anomalies, diversity, and classifier performance.

2.3 Dataset Generation

Dataset generation is an evolved form of data augmentation and can be applied at various text levels, from word-level to document-level (Zhou et al., 2024). At the **word level**, the most intuitive data augmentation method is to substitute words with their synonyms. A thesaurus like WordNet (Kolomiyets et al., 2011) or trained neural networks (Zhang et al., 2015) are the primary techniques for this, but off-the-shelf LLMs can also be used (Smilga and Alabiad, 2024). In addition, words can be substituted with words close in the embedding space (Marivate and Sefara, 2020) or sampled from a learned distribution (Kobayashi, 2018). In back-translation, generally operating at the **phrase level**, a piece of text is translated into a pivot language and back (Sennrich et al., 2016). **Document-level** methods can roughly be divided in work that trains or fine-tunes task-specific generative models (Gangal et al., 2022; Anaby-Tavor et al., 2020; Kumar

et al., 2020) and work that uses off-the-shelf models (Yoo et al., 2021; Sahu et al., 2022; Piedboeuf and Langlais, 2023). Methods in the second line of work are often prompting-based. Within the narrative domain, examples include the mentioned TinyStories, a dataset created via prompting containing stories for 3- and 4-year-olds (Eldan and Li, 2023), and SimpleStories, a similar synthetic dataset to TinyStories (Finke et al., 2025).

3 Methods: Metrics

The metrics for story quality are described in §3.1 and benchmarked on human judgments in §5.1. The metrics for dataset quality are described in §3.2.

3.1 Quantifying Story Quality

We use the six human criteria defined by Chhun et al. (2022) as a guideline for categorizing our metrics. Before calculating the metrics, text preprocessing is applied (Appendix A).

Basic Requirements encompass the criteria *Relevance*, *Coherence*, and *Grammaticality*. We argue that *Relevance* is a criterion that is automatically met, as we assume that instruction-tuned LLMs would be well enough able to follow the instructions in a generation prompt to stay within the relevant text domain. A manual check was performed confirming this argumentation. Following Purdy et al. (2018), we measure *Coherence* with local contextuality, defined as the average of the cosine similarities of every adjacent pair of sentences in the story. As an embedding model for the individual sentences, we use the SentenceTransformer variant of `bert-base-dutch-cased`.

We validate *Grammaticality*, following Purdy et al. (2018), as the log probability of POS-tag sequences in a reference corpus, the Corpus Gesproken Nederlands (van Eerten, 2007).

Surprise is defined as how surprising the end of the story was (Chhun et al., 2022) or the broader degree of deviation from an expected pattern (Karampiperis et al., 2014; Ismayilzada et al., 2024). To measure the degree of deviation from expected patterns, we propose a slightly adjusted form of perplexity: **Creative Perplexity**. This metric employs perplexity in combination with linguistically constrained bigrams to capture higher level creative constructs. An additional explanation of this metric is added in Appendix B.1.

Another way to measure surprise, is **diversity**. We measure diversity at the lexical and semantic levels. On a lexical level, we use Moving-Textual-Lexical-Diversity (MTLD) (McCarthy, 2005), the average segment length for segments with a given type-token ratio. On the semantic level, we employ Self-BLEU (Zhu et al., 2018).

Complexity describes how elaborate the story is. This can be measured at word level, i.e., lexical complexity, and sentence level, i.e., syntactic complexity. We approximate lexical complexity with two simple metrics well-established in readability research: the number of unique words in a story (UW) (Holmes, 1985) and the average word length (WL) (Flesch, 1948). The syntactic complexity is measured using the average occurrence of clausal complements (AC) (Kyle and Crossley, 2018), the mean dependency distance (DD) (Liu, 2008; Liu et al., 2017), and the average syntactic tree depth (SD) (Yngve, 1960).

Human evaluation is used to benchmark the automated metrics and measure more subjective story qualities such as *Engagement*. Sixty participants (recruited via Prolific) each rated ten stories, and three experts each rated thirty stories, sampled from a mix of StoriesFTW and ChiSCor, on a five-point Likert scale with the following items: *grammaticality*, *coherence*, *originality*, *creativity*, *complexity*, *likeability*, and *human likeness*. In total, two-hundred stories received 3 independent crowd ratings and thirty stories 3 expert ratings. For a detailed description of the human evaluation, see Appendix E.1.

LLM-as-a-judge is performed to investigate whether LLMs judge real and generated stories similarly to humans. Bound by computational resources and the ethical guidelines of ChiSCor, stating that the stories cannot be fed to a proprietary model, we chose the following four models: `qwen2.5:14b` (Yang et al., 2024), `mistral-small3.2` (MistralAI, 2025), `aya-expanse:32b` (Dang et al., 2024), and `gemma3:12b` (Gemma Team et al., 2025). The setup is kept as similar to the human evaluation as possible. For a detailed description of the LLM-as-a-judge evaluation, see Appendix F.

3.2 Quantifying Dataset Quality

To measure the quality of the overall dataset, we focus on dataset diversity from the Dang and Verma

(2024) taxonomy (given that linguistic and semantic quality are already measured within the stories themselves). The four diversity metrics used are explained below:

MAUVE is a measure designed to compare generated and human text distributions, employing the Kullback-Leibler divergence in a quantized, low-dimensional space (Pillutla et al., 2021). A high MAUVE score indicates similar diversity and quality in the generated and human texts; a low score could indicate a lack of variety or poor quality in the generated text.

Vendi Score is a content-based diversity metric capturing semantic variations using sentence embeddings (Friedman and Dieng, 2023). It is defined as the exponential of the Shannon entropy of the eigenvalues of a similarity matrix and can be interpreted as the effective number of dissimilar elements in a story.

Compression Ratio is the ratio between the size of a compressed text and that of the original text (Shaib et al., 2025). A higher compression ratio implies a high number of redundant parts in the text. This metric, however, heavily relying on the size of the story, is sensitive to text length, complicating the interpretation of the final score.

Model-as-a-metric for Lexical Semantics To evaluate the lexical semantic properties of the generated data, we train a Word2Vec model and evaluate how the semantic space correlates with human judgments, by computing Spearman correlation against the Dutch Simlex999 dataset (Brans and Bloem, 2024). A high correlation indicates that the semantic space built by the candidate dataset is similar to the human semantic space, suggesting that this dataset contains sufficient information to train a model aligned with human preferences.

4 Methods: Dataset Generation

The dataset generation is divided into two phases: first, designing (§4.2) and evaluating (Appendix C.4) potential generation prompts and, after that, the generation of the dataset (§4.3). The model used for generation is described in §4.1.

4.1 Model Choice

We prompted `gemma-3-4b` (Gemma Team et al., 2025), `phi-3.5-mini` (Abdin et al., 2024), and `llama-3.1-8b` (Grattafiori et al., 2024) to each generate 50 stories, which we analysed qualitatively and quantitatively. The set generated by `llama-3.1-8b` most closely resembled the language of children, so henceforward `llama-3.1-8b` is used. The three story sets and the analysis are added in Appendix C.2.

4.2 Designing Story Generation Prompts

The potential prompts are structured using the framework of Long et al. (2024), who divide a prompt into a *task*, a set of *generation conditions*, and *in-content demonstrations*.

Task Specification The goal of the research, generating stories, forms the basis of the task (hereafter, *baseline*). Additionally, previous research suggests that adding an in-domain persona to the task improves the overall quality of the generated answer (Salewski et al., 2023). So, in the light of our objective to match the narration style of ChiSCor, we investigate the impact of adding the persona of a child between 4-12 years old to the task (hereafter, *persona*).

Demonstrations We investigate whether few-shot prompting with three randomly sampled ChiSCor stories, improves the quality and diversity of the generated stories (referred to as *few-shot*).

Conditions Four different conditions to steer the generation space are considered. Previous research experimented with adding conditions for a theme, topic, style, narrative feature, grammar feature, and an author persona to steer the generation space (Finke et al., 2025). To avoid overly stylised stories, uncommon in the genre of children's stories, we only investigate *themes* and *narrative features* by adjusting the list used by Finke et al. (2025) to contain only elements common to child language. The complete lists are shown in Appendix C.3.

Additionally, following Eldan and Li (2023), we evaluate the effect of adding three randomly sampled words: a noun, a verb, and an adjective. We build on their idea of sampling from a pre-defined dictionary by randomly selecting three words from ChiSCor stories (hereafter, *ChiSCor words*).

Finally, just as Finke et al. (2025), we observe that the majority of generated stories start with the genre-specific phrase 'Once upon a time...' (Dutch: 'Er was eens...'). To minimise that amount, we investigate the effect of adding a compulsory POS tag at the beginning of the story. These POS tags are not randomly sampled but drawn from the POS-

	ChiSCor			StoriesFTW		
	Public	**Expert**	**LLM**	**Public**	**Expert**	**LLM**
Grammar	.22 [.07, .36]	.32 [-.05, .55]	.32 [.21, .41]	.12 [-.01, .24]	.56 [.25, .76]	.13 [.05, .22]
Coherence	.30 [.16, .42]	.49 [.05, .74]	.39 [.29, .48]	.19 [.05, .31]	.41 [-.01, .66]	.28 [.17, .38]
Originality	.32 [.20, .44]	.13 [-.18, .38]	.28 [.17, .37]	.05 [-.08, .18]	-.15 [-.37, .11]	-.10 [-.16, -.03]
Creativity	.41 [.29, .51]	.28 [-.04, .53]	.63 [.54, .70]	.05 [-.07, .16]	-.14 [-.39, -.04]	.24 [.13, .33]
Complex	.35 [.23, .46]	.41 [.11, .57]	.37 [.26, .46]	.09 [-.03, .20]	-.28 [-.39, .14]	-.01 [-.11, .07]
Likeability	.17 [.04, .31]	.48 [.13, .66]	.53 [.43, .61]	.10 [-.03, .22]	.12 [-.19, .36]	.28 [.14, .39]
Human	.05 [-.06, .18]	-.21 [-.36, -.04]	.04 [-.04, .11]	.02 [-.10, .13]	-.36 [-.42, -.31]	.09 [.09, .30]

Table 1: Krippendorff's α per criterion and evaluation method for stories sampled from ChiSCor (left) and StoriesFTW (right). Brackets show 95% bootstrapped confidence intervals.

	Creativity	**Gramm.**	**Coherence**		**Complexity**				
	C. PPL.	*Gramm.*	*Self-BLEU*	*LC*	*UW*	*WL*	*AC*	*DD*	*SD*
ρ_{public}	.181	.181	-.301*	-.154	.721*	.229	.122	.356*	.217
ρ_{expert}	.463	.339	-.523	.015	.846*	.403	.133	.033	-.009
ρ_{LLM}	.257*	.331*	-.154	-.089	.826*	.098	.304*	.327*	.230*

Table 2: Spearman's ρ between the automated metrics and the corresponding human category. $*$ means $p < 0.05$. *C. PPL*: Creative Perplexity; *LC*: Local Contextuality; *UW*: Unique Words; *WL*, average Word Length; *AC*: Average clausal Components; *DD*: Dependency Distance; *SD*: Syntactic tree Depth.

tag distribution of ChiSCor to mimic 'real' story openings (referred to as *POS-tags*).

4.3 Generating *Stories for the Win*

Stories for the Win (henceforward called *StoriesFTW*) is generated with the set of prompts that generated stories closest to ChiSCor, according to our analysis. The generation of these candidate stories is described in Appendix C.4. To avoid a collapse in the diversity and quality of the data, the dataset quality metrics, MAUVE and Vendi score, are calculated every 5000 stories, using a subset of 50 randomly sampled stories. If either metric drops by more than 20%, the prompt is changed to the next-best-performing prompt. The generation is performed on an RTX 3090, with a temperature of 0.7 and sampling disabled (GPU $<$ 500 hours). The generation settings are chosen based on preliminary experiments.

To evaluate the resulting quality of StoriesFTW and keep the comparison with ChiSCor fair, we sample five times 600 stories and report the mean across runs.

5 Results

We first establish the usability of the metrics by correlating them with human and LLM judgements (§5.1) and showing that they are able to differentiate human and generated stories (§5.2). Next, we discuss which prompting methods generate a dataset most similar to ChiSCor (§5.3). Finally, we compare StoriesFTW to ChiSCor (§5.4).

5.1 Validating the Metrics

The metrics are validated on human, both crowd and expert, and LLM judgements on stories sampled from ChiSCor. Firstly, we report inter-rater reliability scores (Krippendorff's α (Krippendorff, 2018)) in Table 1, showing poor agreement across the board; *Human likeness* even shows a structural disagreement for the expert ratings ($\alpha = -.21$). The low scores can be attributed to the subjectivity of the task, a non-standardized annotation scheme, or a misalignment between adult expectations and children's actual language use.

The scores per story are averaged and correlated with the automated metrics using Spearman's ρ (Table 2). Overall, weak non-significant correlations are found, except for the correlation between *unique words* and *Complexity* ($\rho = .721$), which is not found in prior work (Cachola et al., 2025). A handful of metrics has a weak significant correlation with one of the evaluation methods, such as for *Creativity* ($\rho_{LLM} = .257$). Overall, the low α makes it difficult to pinpoint whether the low correlations are caused by malfunctioning automated metrics or unreliable human judgements. Low correlations with metrics well-grounded in the literature, such as syntactic depth, seem to point to unreliable human judgements. While remaining cautious, we decided to use all metrics in the generation process.

	Suprise	Diversity		Complexity				
	C. PPL.	*Self-BLEU*	*MTLD*	*UW*	*WL*	*AC*	*DD*	*SD*
ChiSCor	273	.160	30.3	58.4	3.70	.277	2.66	2.37
Baseline	179	.201	**36.1**	75.3	4.01	.172	2.42	**2.28**
Persona	228	.158	51.0	86.2	4.11	**.281**	**2.80**	2.72
Few-shot	198	.115	56.5	**63.7**	**3.75**	.416	2.96	2.83
Themes	242	.188	54.0	92.4	4.08	.303	2.83	2.77
Narrative feat.	348	.165	52.0	85.6	4.05	.377	3.01	2.60
ChiSCor Words	315	**.160**	53.1	85.9	4.14	.421	2.94	2.79
POS-tags	**290**	.145	53.8	85.7	4.08	.337	2.84	2.72

Table 3: Story quality results per prompting element. Bold-faced values indicates the value closest to ChiSCor on that metric. *C. PPL*: Creative Perplexity; *MTLD*: Measure of Textual Lexical Diversity; *UW*: Unique Words; *WL*, average Word Length; *AC*: Average clausal Components; *DD*: Dependency Distance; *SD*: Syntactic tree Depth.

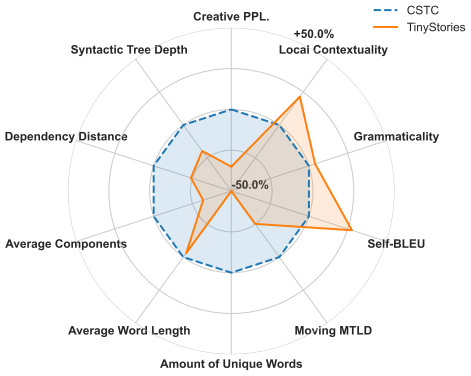


Figure 1: Relative scores of TinyStories compared to a natural story dataset (CSTC) on our defined metrics. Scores are capped on a 0.5 relative difference for visualisation purposes.

	MAUVE	VS	CS
ChiSCor	1.00	24.4	3.15
Baseline	.749	7.35	3.30
Persona	.666	6.78	3.14
Few-shot	**.914**	**12.7**	2.78
Themes	.679	9.06	2.98
Narrative feat.	.674	10.4	2.93
POS-tags	.641	8.28	2.96
ChiSCor words	.641	11.4	**3.15**

Table 4: Dataset quality results per prompting element. Bold-faced values indicates the value closest to ChiSCor on that metric. MAUVE score is relative to ChiSCor. *VS*: Vendi Score; *CS*: Compression Score.

5.2 Differentiating Generated Stories

We use two existing datasets containing children stories, one human-written (a manual selection from CSTC; see Appendix A) and one synthetic (TinyStories), to show the metrics abilities. The comparison is visualised in Figure 1. Each axis represents one of the tested metrics. The dashed blue line represents scores for the CSTC corpus; the solid orange line represents the scores for TinyStories relative to the CSTC corpus. The figure shows that the synthetic corpus is similar to the human-written one in terms of *grammaticality* and *avg. word length*, but different in all other dimensions, showing that the metrics are able to measure a difference between human and generated stories.

5.3 Prompt Evaluation Results

To test which prompt conditions contribute to the diversity and quality of the generated dataset, we systematically evaluate all tasks, conditions, and demonstrations. Table 3 shows the influence of each addition to the baseline prompt on quality metrics at the story-level and Table 4 at the dataset-level.

5.3.1 Story Quality

Basic requirements The results for the coherence metric show little to no difference and a Kruskal-Wallis test, combined with Dunn's test, only showed a statistically significant difference between ChiSCor and few-shot prompting ($H = 25.5$, $p < 0.05$). Despite the statistically significant difference, the effect sizes were both small, suggesting limited practical relevance. The same observation can be made for the results for the grammaticality metric; here the Kruskal-Wallis test showed no statistically significant difference between the datasets. The detailed results are in Appendix B.2. These results indicate that the coherence and grammaticality of the candidate datasets are roughly similar to the ones of human stories, supporting the decision to continue testing with all prompt additions.

Surprise, Diversity, Complexity Table 3 shows the results of the *Surprise*, *Diversity*, and *Complexity* metrics. First of all, it is visible that the POS-

	MAUVE	VS	CS	MaM
ChiSCor	1.00	24.4	3.15	.360
StoriesFTW	.612	17.3	3.01	.324

Table 5: Dataset quality results for ChiSCor and the new dataset. *MaM*: model-as-a-metric, measured as Spearman correlation with $p < 0.05$.

tags prompting element scores closest to ChiSCor for *creative perplexity*. Narrative features and ChiSCor words receive an even higher creativity score, indicating that these prompt elements give access to a larger creative generation space. More research is needed to confirm whether it is caused by genuine creativity or that is impacted by the lexical and syntactic complexity of the stories. Second, the baseline prompt achieves the most similar scores to ChiSCor on *lexical diversity* and *syntactic tree depth*. For the other complexity metrics persona prompting scores closest to ChiSCor. We hypothesize that the two task-only prompts, baseline and persona, score closest to ChiSCor because it does no additional request, allowing the model to focus on generating child language. Few-shot prompting scores close to ChiScor on *lexical complexity*.

5.3.2 Dataset Quality

MAUVE, Vendi, and Compression ratio In Table 4, it is evident that few-shot prompting generates a dataset closest in diversity to ChiSCor. The high MAUVE score shows that it most closely resembles the ChiSCor distribution among the other prompting additions; a relatively unsurprising result given the ChiSCor demonstrations in the prompt. More surprisingly, baseline prompting also has a high MAUVE score, indicating that it is relatively similar to the ChiSCor distribution. The low compression score of few-shot prompting supports the claim that it creates a diverse dataset. Surprisingly, ChiSCor achieves one of the highest compression scores, suggesting it is not as diverse as previously assumed. This result can be explained by an inherent drawback of compression scores, which are lower when the texts are longer (Shaib et al., 2025). The length distribution of the generated texts are relatively equal, as they are generated with the same length condition of 200 words. ChiSCor, however, contains stories of varying lengths, making its compression score less robust.

Lexical Semantics Analysis We consider correlation scores with SimLex999 for the highest scoring prompting elements on dataset quality. Surprisingly, the baseline prompt yields the same Spearman correlation as ChiSCor ($\rho = .360$), suggesting that the baseline stories are suitable for building a lexical-semantic space. The high score, however, is more likely due to the high frequency of common lemmas, which provide more context to build a meaningful word vector. Further, the absence of correlation for the ChiSCor words prompt ($\rho = .099$) is most likely due to the randomness introduced by the compulsory words, leading to words occurring in less likely contexts. Few-shot prompting shows a weak, non-significant correlation ($\rho = .200$). Table 7 in Appendix B.3 contains all correlation scores with their associated p-values.

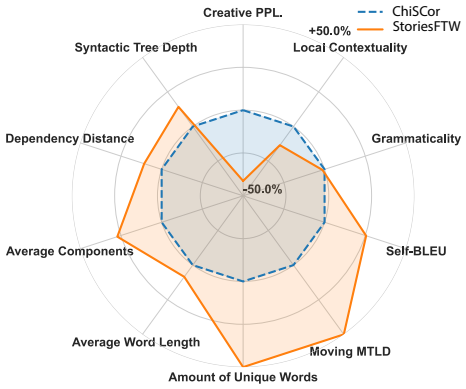


Figure 2: Relative scores of StoriesFTW compared to a natural story dataset (ChiSCor) on our defined metrics. Scores are capped on a 0.5 relative difference for visualisation purposes.

In conclusion, the baseline prompting is closest to the syntactic complexity of ChiScor and has a semantic space that better correlates with human judgements. Few-shot prompting generates the most diverse dataset and achieves a lexical complexity closest to ChiSCor. Further, ChiSCor words has a Self-BLEU score closest to ChiSCor and a relatively high *creative perplexity*. The prompting element closest to ChiSCor on *creative perplexity* is POS-tags. Therefore, we decide to start generating the with few-shot prompt, prioritising dataset diversity, and switch to ChiSCor words and POS-tags if the diversity during generation is dropping.

5.4 Analysing StoriesFTW

The final prompts and exemplary stories are added in Appendix D.

Automatic Metrics Figure 2 shows the relative differences between ChiSCor and StoriesFTW on the story quality metrics. The solid orange line

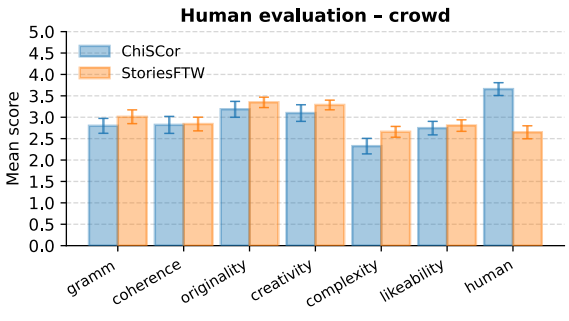


Figure 3: Mean scores calculated over stories on all criteria for crowd evaluation. Error bars indicate 95%-CI over stories. Gramm. is *Grammaticality*.

represents the relative scores of the new dataset compared to ChiSCor. The figure shows that the datasets correspond exactly on *grammaticality* and are close on *syntactic tree depth* and *dependency distance*. StoriesFTW has lower values on *creative perplexity* and *local contextuality*. The lower value for creative perplexity probably indicates fewer creative stories within StoriesFTW, but this conclusion needs to be drawn with caution as the creativity metric only weakly correlated with human judgements (§5.1). StoriesFTW scores higher on all lexical and syntactic diversity and complexity metrics, indicating that the language in the generated dataset is more complex and probably less child-like.

In Table 5, the results of the dataset metrics are shown for ChiSCor and StoriesFTW. The MAUVE score is relatively low compared to the score achieved during prompt testing. A low MAUVE score can be caused by low quality or low diversity. The relatively high Vendi Score and the lower compression score indicate that StoriesFTW should have a similar diversity to ChiSCor, suggesting that the lower MAUVE score is due to the lower quality of the generated dataset. Despite this, the SimLex score is comparable to ChiSCor's, indicating that a meaningful semantic space can be constructed using StoriesFTW.

Human Evaluation The inter-rater reliability scores for StoriesFTW, reported in Table 1, point to poor agreement and in some cases even structural disagreement (Krippendorff, 2018), indicating that the task is highly subjective. Noticeable is that α is even lower than for the metric benchmarking (§5.1). Due to these low agreement scores, the results need to be interpreted with care, especially as averages of these ratings are not representing the variety of human judgements.

In Figure 3, the average crowd scores per criterion are plotted. Note that error bars indicate 95% confidence intervals across stories. The expert and LLM scores are added in Appendix E.2 and F.2. The Mann-Whitney U test show no statistical difference between StoriesFTW and ChiSCor on all criteria except for *Complexity* ($U = 3715$, $n_1 = n_2 = 100$, $p < 0.05$) and *Human likeness* ($U = 8205.5$, $n_1 = n_2 = 100$, $p < 0.05$). The statistical difference on complexity aligns with the difference found with the automated metrics.

6 General Discussion

While testing the metrics, the low observed correlation between the automated metrics and human evaluation was concerning. This can be caused by the subjective nature of the story rating task, already observed in prior work (Peeperkorn et al., 2025). This hypothesis is strengthened by the low inter-rater reliability for both the crowd and the expert ratings. This could be improved by adding more annotators (Long and Wang, 2022).

The low correlations can, however, not completely be attributed to the human evaluations. It is possible that the chosen metrics are not suited to measure children's language as they are designed and benchmarked for adults. This could be the case for the creativity metric which is bound to grammatical models trained on adult text. Further, this explanation could shed light on the low correlation with the SimLex999 score for two of the four generated datasets as SimLex contains adult synonym judgements, and perhaps the lemma pairs should, in the opinion of children, receive a different score. Last, the low coherence scores for ChiSCor is also hypothesised to be inherent to the type of stories it contains: short spontaneous child narratives where events progress quickly. This results in relatively low similarity between adjacent sentences.

It is difficult to say whether our generated dataset StoriesFTW matches the creativity present in ChiSCor. This uncertainty aligns with prior work: some work concludes that generated stories are less creative than human stories (Ismayilzada et al., 2024), while others find no significant difference between generated and human children stories (Orwig et al., 2024). Besides that, we find, in line with prior work (Ismayilzada et al., 2024), that the generated stories are linguistically more complex than human stories. This difference, however, is not visible in TinyStories and CTSC. This is probably due to the

different ages of the target groups of TinyStories and CTSC.

The relatively low MAUVE score for our generated dataset deemed further exploration. A qualitative post-hoc inspection of the data indeed shows that there are several grammatical mistakes in the stories, such as the use of incorrect definite articles. This is most likely due to `llama3.1-8b` being multilingual, but not overly trained on Dutch. If these errors, however, are similar to those children make while learning the language, they would not be problematic in light of our aims. Errors young children make are usually associated with overregularization, omissions, and word order errors (Brown, 1973). In the case of Dutch definite articles, monolingual Dutch children overgeneralize the common gender article 'de' to neuter nouns (Blom et al., 2008). This hints to the fact that at least some of the errors are driven by few-shot prompting with colloquial speech examples.

7 Conclusion

In this paper we have proposed and validated a set of metrics for story and dataset quality and we used the metrics for generating a large collection of stories (StoriesFTW) resembling a Dutch children's storytelling collection (ChiSCor). We found that few-shot prompting with three demonstrations sampled from ChiSCor resulted in narrative language that most closely resembled ChiSCor. On the story level, we combined three prompting techniques to generate a diverse narrative dataset 1000 times ChiSCor's size, amounting to 619,000 stories.

Our evaluation shows low correlation between the automated metrics and the human evaluation, which can be partly explained by the subjective nature of the task. Despite that drawback, we are carefully positive about our result, as the generated stories cannot be distinguished from ChiSCor stories on all defined categories except for *Complexity* and *Human likeness*.

For the near future, we hope that StoriesFTW enables further research into the role that narrative language can play in the LLM training pipeline. Second, additional approaches to enhancing creativity and diversity of generated narratives can be explored, such as setting up a co-creative writing process with children and LMs.

Ethical Considerations

For the human evaluation, we followed the checklist of the Ethics committee of our university, which indicated that evaluation by the ethics review committee was not needed, because we do not collect personal data, and do not involve participants from vulnerable user groups.

The use of LLMs has a considerable environmental footprint. However, as this research aims to present a dataset to improve pretraining efficiency, we deem this justified.

Limitations

The story quality analysis is restricted to metrics capturing lexical and syntactic complexity, while discourse-level complexity is not assessed. As a result, we cannot draw conclusions on the quality of higher-order structural properties of the generated stories, such as discourse organization. A second limitation concerning the metrics is that the notion of diversity is limited to *linguistic* diversity. Broader aspects of diversity, including societal representation and potential biases in generated stories, are not explicitly addressed. Thirdly, multiple metrics rely on SpaCy for syntactic and dependency parsing. During this research, we established that SpaCy's parsing performance for Dutch is not impeccable, which may have led to parsing errors and, consequently, distorted values for metrics that depend on its syntactic analyses. Finally, the use of a relatively small language model may have introduced grammatical errors in the generated Dutch texts. Employing a larger language model could potentially reduce such errors and yield more linguistically robust outputs. It could, however, also lead to less childlike language.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, and 110 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *Pro-*

ceedings of the AAAI conference on artificial intelligence, volume 34(05), pages 7383–7390.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, and 1 others. 2025. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 238–255.

Eden Bens. 2021. Children stories text corpus.

Elma Blom, Daniela Polišenská, and Fred Weerman. 2008. Articles, adjectives and age of onset: The acquisition of dutch grammatical gender. *Second language research*, 24(3):297–331.

Lizzy Brans and Jelke Bloem. 2024. SimLex-999 for Dutch. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14832–14845, Torino, Italia. ELRA and ICCL.

Roger Brown. 1973. *A first language: The early stages*. Harvard University Press.

Isabel Cachola, Daniel Khashabi, and Mark Dredze. 2025. Evaluating the evaluators: Are readability metrics good measures of readability? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 24022–24038.

Maximillian Chen, Alexandros Papangelis, Chenyang Tao, Seokhwan Kim, Andy Rosenbaum, Yang Liu, Zhou Yu, and Dilek Hakkani-Tur. 2023. PLACES: Prompting language models for social conversation synthesis. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 844–868, Dubrovnik, Croatia. Association for Computational Linguistics.

Cyril Chhun, Pierre Colombo, Fabian M. Suchanek, and Chloé Clavel. 2022. Of human criteria and automatic metrics: A benchmark of the evaluation of story generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5794–5836, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, Sandra Kublik, Meor Amer, Viraat Aryabumi, Jon Ander Campos, Yi-Chern Tan, Tom Kocmi, Florian Strub, Nathan Grinsztajn, Yannis Flet-Berliac, and 26 others. 2024. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *Preprint*, arXiv:2412.04261.

Vu Minh Hoang Dang and Rakesh M Verma. 2024. Data quality in nlp: Metrics and a comprehensive taxonomy. In *International Symposium on Intelligent Data Analysis*, pages 217–229. Springer.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Lennart Finke, Chandan Sreedhara, Thomas Dooms, Mat Allen, Juan Rodriguez, Noa Nabeshima, Thomas Marshall, and Dan Braun. 2025. Parameterized synthetic text generation with simplestories. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Dan Friedman and Adji Bousso Dieng. 2023. The vendi score: A diversity evaluation metric for machine learning. *Transactions on Machine Learning Research*.

Varun Gangal, Steven Y Feng, Malihe Alikhani, Teruko Mitamura, and Eduard Hovy. 2022. Nareor: The narrative reordering problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36(10), pages 10645–10653.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *Preprint*, arXiv:2101.00027.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, and 1 others. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Yanzhu Guo, Guokan Shang, and Chloé Clavel. 2025. Benchmarking linguistic diversity of large language models. *Transactions of the Association for Computational Linguistics*, 13:1507–1526.

Rujun Han, Hong Chen, Yufei Tian, and Nanyun Peng. 2022. Go back in time: Generating flashbacks in stories with event temporal prompts. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1450–1470.

David I Holmes. 1985. The analysis of literary style—a review. *Journal of the Royal Statistical Society: Series A (General)*, 148(4):328–341.

Mete Ismayilzada, Claire Stevenson, and Lonneke van der Plas. 2024. Evaluating creative short story generation in humans and large language models. *arXiv preprint arXiv:2411.02316*.

Pythagoras Karampiperis, Antonis Koukourikos, and Evangelia Koliopoulou. 2014. Towards machines for measuring creativity: The use of computational tools in storytelling activities. In *2014 IEEE 14th International Conference on Advanced Learning Technologies*, pages 508–512. IEEE.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457.

Oleksandr Kolomiyets, Steven Bethard, and Marie Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 271–276.

Hugo H. Koning. 2015. *Mythologie*. Elementaire deeltjes ; 32. AUP, Amsterdam.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd workshop on life-long learning for spoken language systems*, pages 18–26.

Kristopher Kyle and Scott A Crossley. 2018. Measuring syntactic complexity in l2 writing using fine-grained clausal and phrasal indices. *The Modern Language Journal*, 102(2):333–349.

Florian Le Bronnec, Alexandre Verine, Benjamin Negrevergne, Yann Chevaleyre, and Alexandre Allauzen. 2024. Exploring precision and recall to assess the quality and diversity of LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11418–11441, Bangkok, Thailand. Association for Computational Linguistics.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10443–10461, Singapore. Association for Computational Linguistics.

Haitao Liu. 2008. Dependency distance as a metric of language comprehension difficulty. *Journal of Cognitive Science*, 9(2):159–191.

Haitao Liu, Chunshan Xu, and Junying Liang. 2017. Dependency distance: A new perspective on syntactic patterns in natural languages. *Physics of life reviews*, 21:171–193.

Haiying Long and Jue Wang. 2022. Dissecting reliability and validity evidence of subjective creativity assessment: A literature review. *Educational psychology review*, 34(3):1399–1443.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082.

Zhicong Lu, Li Jin, Guangluan Xu, Linmei Hu, Nayu Liu, Xiaoyu Li, Xian Sun, Zequn Zhang, and Kaiwen Wei. 2023. Narrative order aware story generation via bidirectional pretraining model with optimal transport reward. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6274–6287.

Ziqiao Ma, Jiayi Pan, and Joyce Chai. 2023. World-to-words: Grounded open vocabulary acquisition through fast mapping in vision-language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 524–544, Toronto, Canada. Association for Computational Linguistics.

Vukosi Marivate and Tshephisho Sefara. 2020. Improving short text classification through global augmentation methods. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 385–399. Springer.

Philip M McCarthy. 2005. *An assessment of the range and usefulness of lexical diversity measures and the potential of the measure of textual, lexical diversity (MTLD)*. Ph.D. thesis, The University of Memphis.

R. Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. 2023. How much do language models copy from their training data? evaluating linguistic novelty in text generation using RAVEN. *Transactions of the Association for Computational Linguistics*, 11:652–670.

Anne McKeough and Randy Genereux. 2003. Transformation in narrative thought during adolescence: The structure and content of story compositions. *Journal of Educational Psychology*, 95(3):537.

MistralAI. 2025. Mistral-small-3.2-24b-instruct-2506. `https://huggingface.co/mistralai/Mistral-Small-3.2-24B-Instruct-2506`.

Alberto Muñoz-Ortiz, Carlos Gómez-Rodríguez, and David Vilares. 2024. Contrasting linguistic patterns in human and llm-generated news text. *Artificial Intelligence Review*, 57(10):265.

Ageliki Nicolopoulou. 2018. Pretend and social pretend play: Complexities, continuities, and controversies of a research field. In *The Cambridge Handbook of Play*, pages 183–199. Cambridge University Press.

Ageliki Nicolopoulou. 2019. Chapter 12. using a storytelling/story-acting practice to promote narrative and other decontextualized language skills in disadvantaged children. In *Narrative, Literacy and Other Skills*, volume 25, pages 263–284. John Benjamins Publishing Company.

William Orwig, Emma R Edenbaum, Joshua D Greene, and Daniel L Schacter. 2024. The language of creativity: Evidence from humans and large language models. *The Journal of creative behavior*, 58(1):128–136.

Andreas Paraskeva, Max Johannes van Duijn, Maarten de Rijke, Suzan Verberne, and Jan N van Rijn. 2025. Data efficient pre-training for language models: An empirical study of compute efficiency and linguistic competence. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Max Peeperkorn, Tom Kouwenhoven, Dan Brown, and Anna Jordanous. 2025. Mind the gap: Conformative decoding to improve output diversity of instruction-tuned large language models. *arXiv preprint arXiv:2507.20956*.

Frédéric Piedboeuf and Philippe Langlais. 2023. Is chatgpt the ultimate data augmentation algorithm? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15606–15615.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

Christopher Purdy, Xinyu Wang, Larry He, and Mark Riedl. 2018. Predicting generated story quality with quantitative measures. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 14(1), pages 95–101.

Gaurav Sahu, Pau Rodriguez, Issam Laradji, Parmida Atighehchian, David Vazquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 47–57.

Leonard Salewski, Stephan Alaniz, Isabel Rio-Torto, Eric Schulz, and Zeynep Akata. 2023. In-context impersonation reveals large language models' strengths and biases. *Advances in neural information processing systems*, 36:72044–72057.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 86–96.

Chantal Shaib, Venkata S Govindarajan, Joe Barrow, Jiuding Sun, Alexa Siu, Byron C Wallace, and Ani Nenkova. 2025. Standardizing the measurement of text diversity: A tool and comparative analysis. In *Proceedings of The 14th International Joint Conference on Natural Language Processing and The 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 36–46.

Veronika Smilga and Hazem Alabiad. 2024. Tüduo at semeval-2024 task 2: Flan-t5 and data augmentation for biomedical nli. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 737–744.

Peter K Smith and Jaipaul L Roopnarine. 2018. *The Cambridge handbook of play: Developmental and disciplinary perspectives*. Cambridge University Press.

Frenette Southwood and Ann F. Russell. 2004. Comparison of Conversation, Freeplay, and Story Generation as Methods of Language Sample Elicitation. *Journal of Speech, Language and Hearing Research*, 47(2):366–376.

Agnes Tellings, Nelleke Oostdijk, Iris Monster, Franc Grootjen, and Antal Van den Bosch. 2018. Basiscript: A corpus of contemporary dutch texts written by primary school children. *International journal of corpus linguistics*, 23(4):494–508.

Yufei Tian, Tenghao Huang, Miri Liu, Derek Jiang, Alexander Spangher, Muhao Chen, Jonathan May, and Nanyun Peng. 2024. Are large language models capable of generating human-level narratives? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17659–17681, Miami, Florida, USA. Association for Computational Linguistics.

Leonoor van der Beek, Gosse Bouma, Rob Malouf, and Gertjan Van Noord. 2002. The alpino dependency treebank. In *Computational linguistics in the Netherlands 2001*, pages 8–22. Brill.

B.M.A. van Dijk. 2025. Theory of mind in language, minds, and machines: a multidisciplinary approach.

Bram Van Dijk, Max van Duijn, Suzan Verberne, and Marco Spruit. 2023. Chiscor: A corpus of freely-told fantasy stories by dutch children for computational linguistics and cognitive science. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 352–363.

Max J van Duijn. 2025. Introduction to part 3: Narrative meaning-making. *Social Psychology and the Ancient World: Methods and Applications*, 8:175.

Laura van Eerten. 2007. Over het corpus gesproken nederlands. *Nederlandse Taalkunde*, 12(3):194–215.

Ethan Gotlieb Wilcox, Michael Y. Hu, Aaron Mueller, Alex Warstadt, Leshem Choshen, Chengxu Zhuang, Adina Williams, Ryan Cotterell, and Tal Linzen. 2025. Bigger is not always better: The importance of human-scale language modeling for psycholinguistics. *Journal of memory and language*, 144:104650–.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33(01), pages 7378–7385.

Victor H Yngve. 1960. A model and an hypothesis for language structure. *Proceedings of the American philosophical society*, 104(5):444–466.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yue Zhou, Chenlu Guo, Xu Wang, Yi Chang, and Yuan Wu. 2024. A survey on data augmentation in large model era. *arXiv preprint arXiv:2401.15422*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 1097–1100.

A Preprocessing

Each story is preprocessed using SpaCy's pipeline (Dutch model: `nl_core_news_lg`; English model: `en_core_web_sm`): all words are lowercased, punctuation is removed, and the story is split into sentences. As the Dutch SpaCy does not handle multiple newlines or newlines combined with white spaces well, all newlines are replaced with a single newline.

Further, the CSTC contained a few works not conventionally seen as child literature. We manually removed these stories. The list we retained can be found in the code.

B Metrics

B.1 Additional Explanation Creative PPL.

Perplexity is the de facto evaluation metric of LMs; the lower the perplexity score, the better the model learned the data. Inverting this reasoning, a high perplexity would indicate that the model did not accurately learn the data distribution, implying that this type of story was not overly present in the training set and can therefore be considered creative or surprising.

Instead of using regular bigrams to represent the stories, we use linguistically constrained bigrams (i.e., bigrams selected based on topic-heavy dependency relations; for example, subject-verb relations), rather than regular bigrams, to create a representation at the discourse level rather than the word level. For example, in the sentence 'The banana flew to Canada', the selected bigram is '(subject, verb), (banana, fly)'. To capture the essence of the expected storyline, we create a linguistically constrained bigram lexicon of BasiScript (Tellings et al., 2018), a dataset containing text written by children in an open domain.

B.2 Results per Metric for Basic Requirements

In Table 6, the basic requirements metrics for story quality per prompting elements are shown.

B.3 Spearman correlation SimLex999

The spearman correlation with SimLex999 judgements for candidate datasets is shown in Figure 7.

	Coherence	Gramm.
ChiSCor	0.357	-2.15
Baseline	0.358	-2.19
Persona	0.333	-2.18
Few-shot	0.309	-2.11
Themes	0.338	-2.16
Narrative feat.	0.348	-2.15
POS-tags	0.336	-2.17
ChiSCor words	0.352	-2.15

Table 6: Basic requirement metrics for story quality per prompting element

	Spearman ρ	*p*-value
ChiSCor	.360	.02
Baseline	.360	.04
Few-shot	.200	.19
ChiSCor words	.099	.52

Table 7: Spearman correlation between fruitful prompting elements and SimLex999-score (Brans and Bloem, 2024). As the set of lemmas present in all datasets is too small to analyse (23), we sampled 45 lemmas for each dataset.

C Generation

C.1 Used prompts

The baseline prompt in English is shown in Figure 4 (translation in Figure 6). The adjusted prompt in English is shown in Figure 5 (translation in Figure 7).

System prompt You are the narrator of a short story (around 200 words).
Use only very simple words that a 3-year-old can understand.
Do not give the story a title or introduction.
User prompt Tell a story.

Figure 4: Baseline prompt in English

C.2 Model Choice

In Table 8, the scores on lexical and syntactical complexity for the three different models are shown. We chose to focus on the complexity metrics for model choice as it is most important that the model outputs language resembling ChiSCors language form.

It is immediately noticeable that all values of the

System prompt You are the narrator of a short story (around 200 words).
You are a child between 4 and 6 years old, and you are telling a story to classmates.
Your audience consists of children your age.
Do not give the story a title or introduction.
User prompt Tell a story.
> INSERT CONDITION OR DEMONSTRATION SPECIFIC USER PROMPT.

Figure 5: Adjusted prompt in English.

System prompt Je bent een verteller van een kort verhaal (rond de 200 woorden).
Gebruik alleen hele simpele woorden die een 3-jarig kind kan begrijpen.
Geef het verhaal geen titel of introductie.
User prompt Vertel een verhaal.

Figure 6: Baseline prompt in Dutch.

Phi3.5 stories are too high to resemble the stories in ChiSCor and child language in general. Both the stories generated by gemma3 and llama3.1-8b receive scores close to ChiSCor, making a closer, qualitative look required. Below are the beginnings of two stories generated respectively by gemma3 and llama3.1-8b.

> "Hey, listen up everyone! It was really great! I was in the woods, right? With my stuffed animal, Pip. Pip is a bunny, and he's really smart. We went looking for a fairytale creature. I saw a squirrel first!..." - Gemma3

	Complexity				
	UW	*WL*	*AC*	*DD*	*SD*
ChiSCor	58.4	3.70	.277	2.66	2.37
Llama3.1	**87.0**	3.96	**.238**	2.83	**2.61**
Phi3.5	146	4.79	.360	3.52	3.86
Gemma3	111	**3.89**	.104	**2.73**	2.01

Table 8: Lexical and syntactic complexity results per tested model. Bold-faced values are the prompting elements that scored closest to ChiSCor on that metric. Further, the column names are as follows: *UW*, Unique Words; *WL*, average Word Length; *AC*, Average clausal Components; *DD*, Dependency Distance; *SD* Syntactic tree Depth.

System prompt Je bent een verteller van een kort verhaal (rond de 200 woorden).
Je bent een kind tussen de 4 en 6 en je vertelt een verhaal aan klasgenoten.
Je publiek bestaat uit kinderen van jouw leeftijd.
Geef het verhaal geen titel of introductie.
User prompt Vertel een verhaal.
> INSERT CONDITION OR DEMONSTRATION SPECIFIC USER PROMPT.

Figure 7: Adjusted prompt in Dutch.

> "Hé, luister allemaal! Het was echt super! Ik was in het bos, hè? Met mijn knuffel, Pip. Pip is een konijntje en hij is heel slim. We gingen op zoek naar een sprookjesbeest. Ik zag eerst een eekhoorn!..."

> "One time, I was at the playground with my friend Lotta. We were climbing on the big jungle gym when suddenly a rabbit popped out of its hole. It was as big as a dog!..." - Llama3.1-8b

> "Een keer was ik met mijn vriendin Lotta in de speeltuin. We waren aan het klimmen in de grote klimrek en opeens kwam een konijn uit zijn holletje. Hij was zo groot als een hond! ..."

Both stories are written in language that resembles the simplicity of child language. Noticeable is that the three lines of the gemma story already contain three exclamation marks and a further investigation of the other stories generated by gemma show that all stories contain an abundance of exclamation marks. This way of storytelling feels, after reading multiple stories, quickly forced and generic, bringing us to the conclusion that llama3.1-8b generates the best stories for our purpose.

C.3 List of narrative elements and themes

Narrative elements: in medias res, een morele les, een onverwachte wending, een onbetrouwbare verteller, vooruitwijzing, innerlijke monoloog, symboliek, een niet-lineaire tijdlijn, een omgekeerde tijdlijn, circulaire verhaalsstructuur, een flashback, een geneste structuur, meerdere perspectieven, een cliffhanger, contrast (juxtapositie), climax-structuur

Story theme: sprekende dieren, fantasiewerelden, tijdreizen, een deadline of tijdslimiet, ruimteverkenning, mystieke wezens, onderwateravonturen, dinosaurussen, piraten, superhelden, sprookjes, het heelal, verborgen schatten, magische landen, betoverde bossen, geheime genootschappen, robots en technologie, sport, schoolleven, vakanties, culturele tradities, magische voorwerpen, verloren beschavingen, ondergrondse werelden, vervlogen tijdperken, onzichtbaarheid, reusachtige wezens, miniatuurwerelden, ontmoetingen met buitenaardse wezens, behekste plekken, vormverandering, eilandavonturen, ongewone voertuigen, geheime missies, droomwerelden, virtuele werelden, raadsels, rivaliteit tussen broers en zussen, schattenjachten, sneeuwavonturen, seizoenswisselingen, mysterieuze kaarten, koninkrijken, levende objecten, tuinen, verloren steden, de kunsten, de hemel

C.4 Evaluating Potential Prompts

For each prompt adjustment, 50 stories are generated with a temperature of 0.7 and sampling disabled. These stories are then evaluated with the metrics discussed in §3. The generation settings are chosen based on preliminary experiments.

All adjustments, except for the *baseline*, use a system prompt *including persona prompting* to keep the generation conditions as close as possible to the final prompt. The baseline and adjusted prompts in English are added together with the original Dutch ones in Appendix C.1.

The 200-word limit in both system prompts is based on the average length of stories in ChiSCor. In the baseline prompt, the target age corresponds to the one used in TinyStories to enable comparison. The target age in the adjusted system prompt is experimentally chosen (see Appendix C.5). The last line in both prompts is added post-hoc as models tended to generate a title or an introduction.

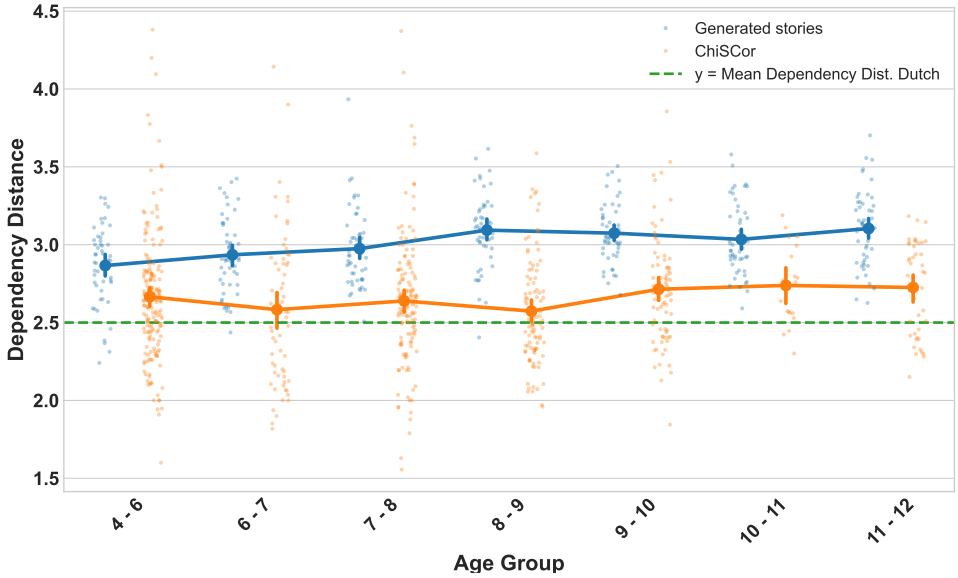


Figure 8: Dependency distance per age group. The blue and orange dots represent respectively the raw data of the generated stories and ChiSCor. The coloured bold lines represent the means of both datasets. Further, the dashed green line represents the mean dependency distance of the Dutch language, calculated by (Liu, 2008). Note that, this mean is calculated based on the Alpino dependency treebanks, containing mostly newspaper articles (van der Beek et al., 2002).

C.5 The Impact of Age in the Prompt

In the final prompt used to create the dataset with, the age group was kept constant between 4 and 6 even though the age group of ChiSCor ranges from 4 to 12. In Figure 8, we show the reasoning behind this choice. It shows that the syntactic complexity is almost stable across age groups as depicted with the solid blue line. Similarly, varying the age groups in the prompt shows a neglect-able increase in dependency distance throughout the age groups. In our effort to model the generated dataset like ChiSCor, we, therefore, chose to keep the age group constant while generation the new dataset.

System prompt You are the narrator of a short story (around 200 words).
You are a child between 4 and 6 years old, and you are telling a story to classmates.
Your audience consists of children your age.
Do not give the story a title or introduction.
User prompt Tell a story.
Here are 3 stories as inspiration:
Story 1: STORY_1
Story 2: STORY_2
Story 3: STORY_3

Figure 9: Final few-shot prompt.

D *StoriesFTW*

D.1 Used Prompts *StoriesFTW*

In Figure 9, the final few-shot prompt is shown (Dutch translation in Figure 10). In Figure 11, the final ChiSCor-words prompt is shown (Dutch translation in Figure 12). In Figure 13, the final POS-tag prompt is shown (Dutch translation in Figure 14).

D.2 Example Stories *StoriesFTW*

Story 1 "Once upon a time there was a girl and she had a friend. And she said: 'Did you know what was really bad once?' My father was at a festival once. And he had to go to the toilet. And then my father wanted to go to the toilet very quickly. But there was no toilet nearby. So then he really needed to pee. But then he ended up peeing in his pants. So because of that, that girl told the friend... and then the other friend went and told her parents. To her teacher and to all the children in the whole city! And then the friend got very angry!"

Story 1 "Er was eens een meisje en die had een vriendin. En die zei: "Wist je wat heel erg was een keer?" Mijn vader was een keer op een festival. En hij moest toen naar de wc. En toen wou m'n vader heel snel naar de wc gaan. Maar er was geen wc in de buurt. Dus toen moest hij heel nodig plassen. Maar toen plaste die uiteindelijk in

System prompt Je bent een verteller van een kort verhaal (rond de 200 woorden).
Gebruik alleen hele simpele woorden die een 3-jarig kind kan begrijpen.
Geef het verhaal geen titel of introductie.
User prompt Vertel een verhaal. Hier zijn 3 verhalen om inspiratie uit op te doen:
Verhaal 1: STORY_1
Verhaal 2: STORY_2
Verhaal 3: STORY_3

Figure 10: Final few-shot prompt in Dutch.

User prompt Tell a story.
The story needs to contain the following verb: VERB, the following noun: NOUN and the following adjective: ADJECTIVE."

Figure 11: Final ChiSCor-words user prompt.

zijn broek. Dus daardoor zei dat meisje dat tegen de vriendin... en toen ging die andere vriendin het aan haar ouders vertellen. Aan haar juffrouw en aan alle kinderen in de hele stad! En toen werd de vriendin heel boos!"

Story 2 "First, that stupid chicken! It saw a snail and was so scared that it ran away... And then it laid an egg! How strange, isn't it? And then the snail ate it. That is very strange, too. I think I'd rather not talk about those animals. My girlfriend has a pet and it eats happily out of her hand. Now I have a story too. Once upon a time, there lived a boy in a world with knights. He wasn't very rich, but he did have many friends. In the forest, he saw a lot of animals: deer, squirrels, rabbits... And then he saw a broken house. But no one came out. Still, he went inside and there sat a grandma. She asked him if he wanted a candy. No, said the boy. But his own house caught fire later! What is this crazy story? He went back to the forest and asked the grandma: "What happened to my house?" And she answered: "You didn't eat my candy, now your house is in ashes."

Story 2 "Eerst die domme kip! Die zag een slak en schrok zo erg dat hij wegrende... En toen legde hij een ei! Wat vreemd, hè? En toen at de slak hem op. Dat is ook heel vreemd. Ik denk dat ik liever niet over die dieren praat. Mijn

User prompt Vertel een verhaal.
Het verhaal moet het volgende werkwoord bevatten: VERB, het volgende zelfstandig naamwoord: NOUN en het volgende bijvoegelijk naamwoord: ADJECTIVE."

Figure 12: Final ChiSCor-words user prompt in Dutch.

User prompt Tell a story.
Start the story with a word with the following pos-tag POS-TAG

Figure 13: Final POS-tag user prompt.

vriendin heeft een huisdier en die eet lekker uit haar hand. Nu heb ik ook een verhaal. Er woonde eens een jongen in een wereld met ridders. Hij was niet erg rijk, maar hij had wel veel vrienden. In het bos zag hij heel veel dieren: herten, eekhoorns, konijnen... En toen zag hij een kapot huis. Maar er kwam niemand uit. Toch ging hij naar binnen en daar zat een oma. Zij vroeg hem of hij een snoepje wilde. Nee, zei de jongen. Maar zijn eigen huis stond later in brand! Wat is dit gekke verhaal? Hij ging terug naar het bos en vroeg aan de oma: "Wat is er met mijn huis gebeurd?" En die antwoordde: "Je hebt mijn snoepje niet opgegeten, nu ligt je huis in as."

E Detailed Description of Human Evaluation

E.1 Set-up Human Evaluation

Questionnaire Set-up The frame of the questionnaire consisted of a consent question (Figure 15), information about the task and criteria and a repeating block of the story, a comprehension question, and a five-point Likert scale. The information about the task and criteria is shown in Figure 16. In the repeated block, first only the story was shown to force participants to read carefully. After that, a comprehension question was asked. The amount of wrong answers over all ten stories was counted as a proxy for individual judge reliability. When the comprehension question was correctly answered, the Likert items were shown.

Criteria The criteria *Grammaticality*, *Coherence*, and *Complexity* map directly to the criteria defined in §3. Then, the criterion *Likeability*

User prompt Vertel een verhaal.
Begin het verhaal met een woord met het volgende pos-tag POS-TAG

Figure 14: Final POS-tag user prompt in Dutch.

By clicking "I consent" below, you confirm that you have read and agree to the following information.

- I confirm that I have been informed about the nature and method of the research.
- I agree to participate in this research. I understand that I can withdraw from my participation at any time.
- If my research results are used in scientific publications or made public in any other way, this will be done anonymously. If you experience accessibility issues during the experiment, please feel free to contact the researcher.

Figure 15: Consent question.

is added to measure *Engagement* (as defined by (Chhun et al., 2022)). We stray here from the original definition to make it more suitable for children's stories. The criteria *Creativity* and *Originality* are respectively used to measure creativity and as construct validity. Last, the criterion *human likeness* is straight up used to investigate how close the generated stories align to the real stories.

In the questionnaire, the criteria are sparsely explained to participants to avoid a bias towards our own definition of the more subjective criteria, e.g. creativity. The explanation for the participants is shown in Figure 17.

Participants We recruited 60 participants with a Dutch nationality to rate in total 200 stories of which 100 from ChiScor and 100 from *StoriesFTW*. Each participant rated 10 stories such that all stories have 3 ratings. Participants were paid £12 per hour, which is the upper limit of the advised pay rate on Prolific. Besides that, we ask 3 experts (people with a PhD in literary studies) to rate 30 stories.

For this study, we ask you to evaluate 10 short stories based on 7 criteria: *Grammaticality*, *Coherence*, *Originality*, *Creativity*, *Complexity*, *Likeability*, and *Human likeness*.

The stories average around 200 words. First, only the story is shown for 15 seconds. This gives you time to read it carefully. Afterwards, a simple comprehension question is asked. After correctly answering the comprehension question, the criteria are displayed. You evaluate the criteria by giving a score on a scale of 1 to 5.

Some stories were told by children, others were generated by AI.

Figure 16: Information about the task and criteria.

E.2 Results of the Human Evaluation

The inter-rater reliability is divided based on source (either ChiSCor or *StoriesFTW*) and are shown in Table 1.. The spearman correlations with the automatic metrics is shown in Table 2. The average score per criteria for the experts and LLMs are shown in Figure 18.

F Detailed Description of LLM-as-a-judge Evaluation

F.1 Set-up LLM Evaluation

The LLM-as-a-judge evaluation is modelled after the human evaluation. We analyse both the generation and the log probabilities. The prompts are given in Figure 19 and 20. We used the following four models: `qwen2.5:14b` (Yang et al., 2024), `mistral-small3.2` (MistralAI, 2025), `aya-expanse:32b` (Dang et al., 2024), and `gemma3:12b` (Gemma Team et al., 2025) as judges.

F.2 Results LLM-as-a-judge Evaluation

The inter-rater reliability scores (Krippendroff's α), reported in Table 1 point to poor agreement and in some cases even structural disagreement (Krippendorff, 2018), indicating that the task is highly subjective for LLMs.

In Figure 18, the average llm scores per criterion are plotted on the left side of the figure. The plotted values are the ratings based on the generations but it is confirmed that they align with the

To begin, we provide an explanation of the seven criteria:

1. Grammaticality. To what extent is the story grammatically correct? Do not be too strict: some sentence structures work in spoken language but not in written language.
2. Coherence. To what extent is the story a coherent whole? Think, for example, of a consistent storyline or the sudden introduction of characters.
3. Originality. How original do you find this story? There is no right or wrong answer here; we are looking for your opinion!
4. Creativity. Is this story creative in your eyes? There is no right or wrong answer here; we are looking for your opinion!
5. Complexity. How complex is this story? Think, for example, of the complexity of the sentence structure, but also that of the storyline.
6. Likeability. Did you enjoy reading this story? There is no right or wrong answer here; we are looking for your opinion!
7. Human likeness. How likely do you think that this story was generated by AI rather than told by a child?

Figure 17: Explanation for participants.

log-probabilities. Mann-Whitney U tests showed a statistical difference on all criteria ($532.5 < U < 6803$, $n_1 = n_2 = 100$, $p < 0.05$) except for *Originality* ($U = 5048.5$, $n_1 = n_2 = 100$, $p = 0.905$). The error bars are considerably smaller than for the expert ratings as the sample size is larger and the LLM evaluation has 4 raters per story. Interestingly, LLMs rated *StoriesFTW* higher on all criteria, except for *Human likeness*, confirming prior research (Bavaresco et al., 2025) that LLM might be biased towards generated text.

G License of Used Datasets

Within this work, we used the following datasets: TinyStories (Eldan and Li, 2023), CSTC (Bens, 2021), CGN (van Eerten, 2007), and BasiScript (Tellings et al., 2018). Tinystories is under a Community Data License Agreement – Sharing, Version 1.0 and CSTC under a CC0: Public Domain license. For the use of both CGN and BasiScript, a non-commercial license is signed. This license allows us to use the datasets for our own research and use it to develop new datasets as long as the datasets are not solely copied. With the creation of StoriesFTW, we adhere to this license.

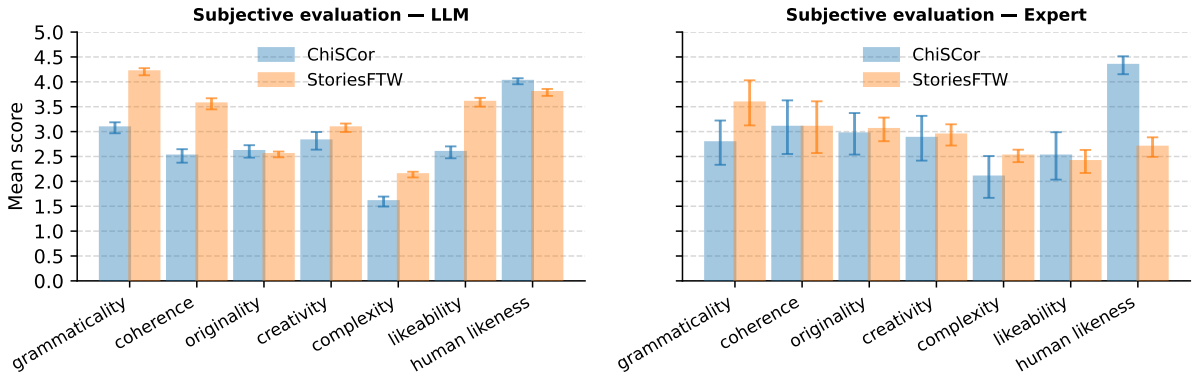


Figure 18: Mean scores on all criteria for public and expert human evaluation. Error bars indicate 95%-CI.

(Generation) System prompt You are an expert evaluator of Dutch children's stories. Your task is to rate a short Dutch story on seven quality criteria, each on a scale from 1 (very poor) to 5 (excellent).

The stories are a mix of real stories told by children and AI-generated stories designed to mimic child-told stories. Rate each story objectively based on the criteria below.

> INSERT CRITERIA AND EXPLANATIONS.

Return ONLY a JSON object with exactly these keys: grammaticality, coherence, originality, creativity, complexity, likeability, human likeness. Each value must be an integer between 1 and 5. Do not include any other text.
Example:
{"grammaticality": 4, "coherence": 3, "originality": 2, "creativity": 3, "complexity": 2, "likeability": 4, "human likeness": 3}

Figure 19: Prompt LLM evaluation.

(Log probability) System prompt You are an expert evaluator of Dutch children's stories.
You will rate a story on a single criterion on a scale from 1 (very poor) to 5 (excellent).
Respond with ONLY a single digit: 1, 2, 3, 4, or 5. No explanation, no other text.

Figure 20: Prompt LLM evaluation based on log probabilities.

Findings of the Fourth BabyLM Challenge & Workshop: Going Beyond English

BabyLM Team

Leshem Choshen
IBM Research, MIT

Ryan Cotterell
ETH Zürich

Mustafa Omer Gul
Cornell University

Jaap Jumelet
University of Groningen

Tal Linzen
NYU

Aaron Mueller
Boston University

Suchir Salhan
University of Cambridge

Raj Sanjay Shah
Georgia Tech

Alex Warstadt
UC San Diego

Ethan Gotlieb Wilcox
Georgetown University

Abstract

This report summarizes the findings from the 4th BabyLM Challenge. The BabyLM Challenge is a shared task aimed at closing the data-efficiency gap between human and machine language learners. This year, the challenge was held as part of the 2nd BabyLM Workshop. We retained the text-only evaluation tracks, *Strict* and *Strict-Small*, while allowing multimodal data and interactive training within these tracks. We also introduced a new *Multilingual* track, where language models were required to perform well on BabyLM evaluations in English, Dutch, and Mandarin Chinese. We observe that GPT-BERT remains a strong architectural foundation for sample-efficient pre-training, and that more training epochs, recursion/looping, and data preprocessing generally improve performance. This year's winning submissions to the new multilingual track also show that LLM-generated parallel data can be effective at improving multilingual capabilities in low-data settings.

1 Introduction

In recent years, the trend has been toward ever-larger-scale language model (LM) training. This leads to LM methods research often being considered the domain of industry research. Further, LM methods research is also often considered irrelevant to the study of language learning and processing in humans, as the number of words required to train a state-of-the-art model is orders of magnitude greater than the number of words a human would hear in their lifetime.

To advance the science of language modeling at the academic scale and create more cognitively plausible LMs, the BabyLM Challenge encourages researchers to train language models with the amount of language typical of human language acquisition. This paper presents and analyzes the main findings from the fourth iteration of the BabyLM Challenge.[1] We also describe participating systems and some key takeaways from the BabyLM Workshop, to which participants could submit papers without submitting to the challenge.

The objective of the BabyLM Challenge is to train a model with 100M words or fewer. For entrants wishing to work at an even smaller scale, we also organized a 10M-word track. The challenge explicitly refrains from restrictions on anything other than the word count. In doing so, we hope to encourage new approaches that improve LMs' sample efficiency and reveal why standard LLMs are so data hungry. The previous year's interaction track has been folded into these tracks. This year, we additionally introduce a multilingual track inspired by BabyBabelLM (Jumelet et al., 2026a). The multilingual track encourages authors to train efficient models that perform well in three languages (English, Dutch, and Mandarin Chinese) within the same fixed word budget as the monolingual tracks.

Summary of takeaways. BabyLMs continue to improve at cognitive-alignment and NLP tasks in the *Strict* and *Strict-Small* tracks. Statistical analyses of challenge contributions show that models with longer sequences, more training epochs, and more layers tend to perform better. Data preprocessing and specific model architectures (e.g.,

[1] For findings from previous years, see Warstadt et al. (2023); Hu et al. (2024); Charpentier et al. (2025).

Proceedings of the Second BabyLM Workshop, pages 479–492
October 29, 2026 ©2026 Association for Computational Linguistics

GPT-BERT) also boost overall performance. In the multilingual track, we observe overall higher scores in English and Dutch, with models' performance on Chinese slightly underpowered based on their overall capabilities. The BabyLM workshop contributions explored a range of topics, with particular emphasis on multilingual analysis and data-efficient learning via architectural and training innovations. Overall, we were pleased to see an increase in the number of BabyLM participants, indicating growing interest in the intersection of cognitive modeling and data-efficient pretraining.

2 Competition Details

2.1 Data

Text-only Corpora. We provided updated versions of the BabyLM corpora used in the previous challenges, with 10M-word and 100M-word datasets for the *Strict-Small* and *Strict* tracks (Choshen et al., 2026). These corpora included child-directed speech from CHILDES; dialogue from the spoken portion of the British National Corpus and Switchboard; children's stories from Project Gutenberg; movie and television subtitles from OpenSubtitles; and articles from Simple English Wikipedia.

Dataset Revision. The main change from previous years was detoxification of the training data. This revision followed prior work identifying potentially toxic content in the BabyLM corpus and toxic generations during interactive training (Salhan et al., 2025; Trhlík et al., 2026).

Multimodal and Multilingual Data. Participants could use multimodal training data within the relevant track's word budget. The new Multilingual track focused on English, Dutch, and Chinese, using data from BabyBabelLM (Jumelet et al., 2026a). Participants could choose their own mixture of these languages, subject to the total data budget and language-specific Byte Premium adjustments.

2.2 Strict & Strict-Small Tracks

Track Overview. The *Strict-Small* and *Strict* tracks required submissions to be trained on datasets of at most 10M and 100M words, respectively. Participants could use either the provided BabyLM corpus or construct their own training datasets (provided they adhered to the word limit). Participants could use any model architecture that could assign log-likelihoods or pseudo-log-likelihoods to strings of text. We evaluated both tracks on language-only tasks. This year, we also merged the *Multimodal* and *Interaction* tracks from previous BabyLM offerings into the *Strict* and *Strict-Small* tracks.

Interactive Training. Participants could use an external model in their submission model's training pipeline. Like last year's *Interaction* track, the external model could provide scalar rewards or natural language feedback, or generate training examples conditioned on the submission model's outputs. External models were restricted to a predefined list, but their training and fine-tuning were otherwise unrestricted. External model training text counted toward the word budgets for the respective tracks. The constraints also required that the submission model could not access the external model's tokenizer, weights, hidden states, or output distribution.

Training Duration Limitations. We restricted competition submissions to at most 100M words of cumulative training exposure for *Strict-Small* and 1B words for *Strict*, counting repeated passes over the same input. For models that generated text during training, the exposure budget also includes their generated output. These limits corresponded to ten epochs over a dataset at the full word budget.

Intermediate Checkpoints. Following last year's setup, we asked participants to submit intermediate model checkpoints at increasing word-exposure intervals. Specifically, we required checkpoints every 1M words through 10M words and every 10M words through 100M words; for *Strict*, we also required checkpoints every 100M words through 1B words.

2.3 Multilingual Track

The Multilingual track was introduced this year, in which participants trained the BabyLM model on a mixture of languages. We limited the number of languages to keep the competition format computationally feasible on smaller hardware. With a similar compute budget as the Strict track of 100M tokens, participants were free to define a mixture of English, Chinese, and Dutch data. Models were scored based on the average performance across these three languages.

2.4 Evaluation

Evaluation for the *Strict* and *Strict-Small* tracks remained similar to prior editions. We divided tasks into two groups: **NLP tasks** that could be measured against ground truth, such as GLUE and BLiMP, and **Human-like tasks**, which were correlated with human judgments. From previous edition's results, it became clear that these human-like correlations are quite unstable and caused some models to score highly on the leaderboard by pure chance. Leaderboard winners are now decided only based on NLP task performance. Evaluation of the **multilingual** track was leveraged from BabyBabelLM, which contains both zero-shot and fine-tuning evaluation tasks for the three languages of the track. Tasks include language-specific linguistic evaluation (Warstadt et al., 2020; Suijkerbuijk et al., 2025; Liu et al., 2026; Jumelet et al., 2026b), classification tasks (Adelani et al., 2024), and question-answering (Romanou et al., 2024).

Hidden tasks. We supplemented the released evaluation suite with hidden tasks to assess generalization beyond the public evaluations. Across all three tracks, we introduced Global-PIQA (Chang et al., 2026), which tested physical commonsense reasoning using parallel and non-parallel evaluation sets. We used the English tasks for *Strict* and *Strict-Small*, and the English, Dutch, and Chinese tasks for the Multilingual track. For the Multilingual track, we also introduced **part-of-speech probing**, released two weeks before the submission deadline. Linear probes trained on Universal Dependencies annotations assessed the part-of-speech information available in model representations across languages (Nivre et al., 2020; Ravishankar et al., 2019). For Chinese, we added the Hanzi-structure and Hanzi-pinyin tasks, which use minimal pairs to assess models' knowledge of character structure and phonology, respectively (Song et al., 2026).

Multilingual reading-time evaluation. We introduced new evaluations on eye-tracking data using the Multilingual Eye-Tracking Corpus (MECO) (Siegelman et al., 2022). Models were evaluated on how their surprisals improve prediction of human first-pass reading times, controlling for word length, frequency, and sentence position. Models were evaluated on first language (L1) MECO data for Dutch, Chinese, and English, and second language (L2) English.

2.5 Baselines

Similar to the evaluation, baselines for the Strict and Strict-Small tracks are mostly the same as in the last iteration, except for removing GPT-BERT as a baseline. We have retrained the naive, autoregressive GPT-2 Small baseline (Radford et al., 2019) using the same recipe, but with the revised dataset. As a result of the merging of the Interaction track with the Strict and Strict-Small tracks this year, we have also retrained the Interaction baselines from the Third BabyLM Challenge (Charpentier et al., 2025) on the updated dataset.

For the Multilingual track, we re-use the autoregressive GPT-2 Small training recipe but vary the dataset construction process. Our primary baseline for the track mixes the BabyLM corpora for English, Dutch, and Chinese (Jumelet et al., 2026a) such that each language occupies a similar proportion after Byte-level Premium adjustments. For each language dataset, we subsample the necessary number of words while preserving the proportion of tokens contributed by each constituent data source for the language. We then train a BPE tokenizer on this data mixture and train the model without any further interventions in terms of language ordering. This is a naive baseline intended to provide participants with the resources and utilities to process and mix the datasets.

3 Submissions

We received 44 Challenge submissions, 8 ARR submissions, and 58 workshop submissions. Out of these, we accepted 36 Challenge submissions (81.8%), 7 ARR submissions (87.5%), and 35 workshop papers (60.3%). This number of submissions marks a stark increase compared to prior editions, which we take as a sign of the community growing and there being a strong interest in the Challenge as a way to test data-efficient language modeling in a controlled environment, and the Workshop as a place to explore broader scientific questions related to this topic.

4 Analysis of Challenge Results

The BabyLM Challenge submissions explore a wide range of strategies for improving language modeling under strict data and compute constraints. We survey the challenge submissions, followed by a quantitative analysis of leaderboard scores.

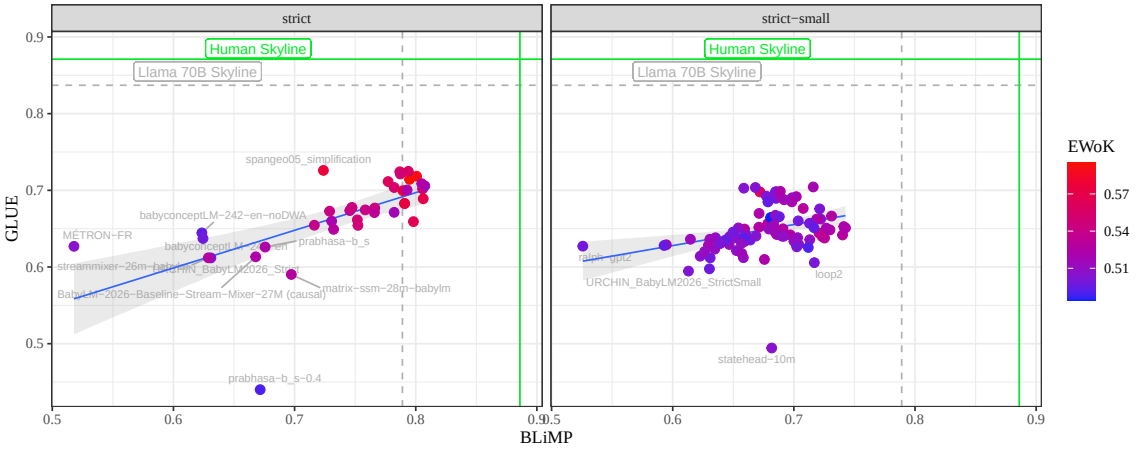


Figure 1: Comparison between models' scores on BLiMP and GLUE vs. Human scores reported for these tasks. Scores are shown for the *Strict* and *Strict-Small* tracks. We visualize these two tasks, as they have been used consistently through all iterations of the BabyLM challenge. For overall scores, see fig. 3. All leaderboard submissions are considered here.

4.1 Survey

Architecture. Submissions investigated mixture-of-experts models (Mustafa et al., 2026; Jain et al., 2026; Bohn et al., 2026), recurrent and weight-tied architectures (Furr, 2026; Miles, 2026; Gülbahar et al., 2026; Kamzela et al., 2026; Fan et al., 2026), higher-order attention (Chan, 2026), and parameter-efficient embedding and projection layers (Mosquera-Gómez et al., 2026). Other approaches included a horizontal spiking architecture that operates without attention (Chiang, 2026) and input-dependent routing over earlier-layer representations (Wang, 2026). Furr (2026) demonstrated grammatical-discrimination capabilities with about 10M parameters and a fixed-size recurrent state, although its gains did not consistently extend to semantic or entity-tracking tasks. Gülbahar et al. (2026) explored recursive Transformers that increase computational depth without proportionally increasing the number of parameters.

Several papers examined whether additional recurrent computation could compensate for limited parameters or data (Hotta et al., 2026; Haller et al., 2026; Fan et al., 2026). The results depended on the task and comparison budget. For example, Miles (2026) reported lower validation bits per byte with looped depth, but the downstream gain was small relative to its uncertainty, and training took longer. Hotta et al. (2026) found a text-only BLiMP advantage for recurrence under matched inference compute, but recurrence did not clearly mitigate the performance decline associated with their multimodal training condition. These findings motivate distinguishing parameter efficiency, data efficiency, and computational cost.

Training Objective and Optimization. Submissions explored auxiliary objectives (Palomino, 2026; Benedikt et al., 2026), preference optimization (Štěpánková et al., 2026; Mishra and Roy, 2026), adaptive teacher feedback based on corrections to student-generated text (Islam et al., 2026), and optimizer and learning-rate choices (Satheesh et al., 2026). Other work combined masked and causal language modeling in a multilingual OLMo2 model (Wei, 2026), introduced objectives for syntagmatic and paradigmatic relations (Yang et al., 2026b), or varied the balance between replaced-token detection and masked language modeling (Mikat and Fahim, 2026). Štěpánková et al. (2026) investigated how preference-data selection affected downstream performance, while Mishra and Roy (2026) applied preference post-training after downstream performance had saturated. The evidence did not uniformly favor additional objectives or feedback; the data2vec auxiliary did not yield significant task-level gains in the controlled comparison of Benedikt et al. (2026), and the official-corpus baseline retained the highest overall score in Islam et al. (2026).

Data composition and curricula. Submissions examined multilingual language allocation and targeted data design (Mori et al., 2026), tokenization, packing, and curriculum learning (Yang et al., 2026a), code-switching curricula (Rooryck et al., 2026; Huang et al., 2026), meaning-preserving

rewrites (Wu et al., 2026), and concise parallel text generated by an LLM (Edman and Fraser, 2026). Hwang et al. (2026) compared mixed multilingual training with a curriculum that began with monolingual exposure. Mixed training performed better in all 15 matched configurations, and several language-dominant mixtures exceeded the balanced baseline; these comparisons used one run per configuration. Rodriguez et al. (2026) combined curriculum pretraining with layer repetition at inference.

Tokenization and representation. Submissions studied subword segmentation learned during language-model training (Meyer, 2026), semantically informed morphological tokenization for English (Rodriguez, 2026), and character-level tokenization in Qwenfant (Edman and Fraser, 2026). Other papers investigate how models can acquire particular capabilities or representations, including dynamic concept computation (yizhe xu et al., 2026), linguistic parameters (Chesi et al., 2026), reducible-loss spans and self-generation (Kalfus and Goyal, 2026), and practical life skills (Štěpánková et al., 2026). Finally, several submissions examine broader questions about data-efficient learning, including whether looped computation can substitute for parameters (Fan et al., 2026), whether recurrent depth is beneficial under data limitations (Haller et al., 2026), whether layer duplication can improve small models (Rodriguez et al., 2026), how relational inductive biases affect learning (Yang et al., 2026b), and how dense supervision and curriculum-guided rebalancing can improve small-data pretraining (Mikat and Fahim, 2026).

Together, these submissions span architectural, data, objective, tokenization, and training-procedure interventions for studying how much language learning can be achieved from limited exposure. The highest-scoring Challenge submissions span several approaches to improving learning under limited data. These papers illustrate the breadth of the Challenge: improving small language models can involve changing the model's computational structure, the representation of its input, the composition and ordering of its experience, or the optimization procedure used to learn from that experience.

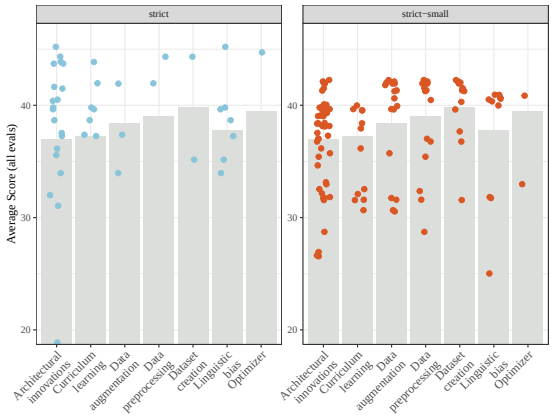


Figure 2: Approaches to the challenge broken down by score. Submissions that use more than one approach are counted multiple times.

4.2 Quantitative Analysis

We conduct various 'meta-analyses' on the models submitted to the BabyLM Leaderboard to gain an understanding of the types of submissions people have made this year, and which model properties drive strong performance. Participants had to enter various model parameters and experimental design choices, which we can leverage to plot and fit correlations between these properties and leaderboard scores. We hope this will inspire next year's participants to set up their experimental design.

Figure 1 shows the GLUE performance set out against BLiMP performance for the Strict and Strict-Small models. Models are set out against a Human and LLM 'skyline'. While strong GLUE performance remains challenging for models of this size, this result does show that their *linguistic* abilities are on par with Llama3-70B, a model several orders of magnitude larger.

In Figure 2 we show a breakdown of model scores by Contribution Type, using contribution type categories from previous BabyLM findings papers. The majority of model submissions introduced architectural innovations, resulting in a wide range of model scores. Curriculum learning remains popular, while data augmentation and data preprocessing both see a strong set of model contributions. Contribution Type alone, however, is not sufficient to explain model results in detail, so we also conduct a broader linear modeling analysis.

In Table 1 we show the β coefficients of a linear model fitted on the normalized average NLP task scores of the 118 BabyLM submissions in the Strict and Strict-Small tracks. Significant variables

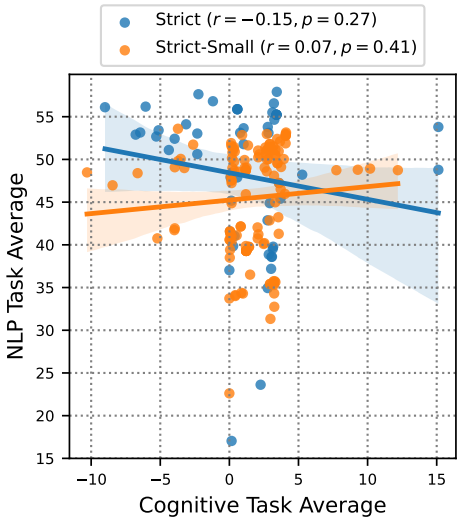


Figure 3: Connection between NLP task performance and "Cognitive tasks". Cognitive tasks are correlated with human scores and include reading time prediction and age-of-acquisition.

Variable	Value	n	β	p
Batch Size			-0.12	0.16
Learning Rate			0.19	0.17
Max Seq Length			0.22	**0.02**
PFLOPS			-0.03	0.72
Token Set Size			-0.07	0.44
# Training Epochs			0.27	**<0.01**
# of Attention Heads			0.16	0.20
# of Layers			0.20	**0.03**
# of Parameters (M)			0.05	0.66
Main Contribution	Architectural	62	-0.24	0.20
	Controlled experiments	37	-0.08	0.70
	Curriculum learning	19	0.34	0.21
	Data augmentation	13	-0.18	0.50
	Data preprocessing	11	0.72	**0.03**
	Dataset creation	7	0.01	0.97
	Hyperparameter tuning	6	0.16	0.67
	Linguistic bias	17	-0.21	0.37
	Teacher models	4	-0.08	0.89
	Training objective	27	-0.02	0.95
Model Type	Decoder	68	-0.07	0.70
	Encoder	36	0.57	**<0.01**
	Encoder-Decoder	8	-1.48	**<0.01**
	GPT-BERT	6	0.97	**<0.01**
Optimizer	AdamW	71	0.24	0.30
	LAMB	26	-0.11	0.64
	Muon	10	-0.12	0.65
	MuonAdamW	8	-0.40	0.19
	Other	3	0.39	0.39
Track	strict	35	0.27	**<0.01**
	strict-small	83	-0.27	**<0.01**

Table 1: Linear model predicting z-scored NLP Average for 118 BabyLM Strict and Strict-Small submissions ($R^2 = 0.62$). Continuous predictors are z-scored; categorical coefficients are deviations from the mean across levels.

are highlighted, indicating various model properties driving strong performance. *Max Sequence Length* is positively correlated with performance, as is the number of training epochs and the number of layers (more so than the number of parameters). Out of the Contribution Types, *Data Preprocessing* is the only significant factor. Model Type again shows, as in similar years, that the hybrid GPT-BERT model (Charpentier and Samuel, 2024), which combines the strengths of decoder and encoder models, remains a very strong model architecture. Encoder models outperform decoder models; encoder-decoder models perform poorly. Most contributions used the AdamW optimizer, but none of the optimizer types had a significant effect.

Finally, we look at the connection between NLP task performance and the Human-like (Cognitive) tasks in Figure 3. Measuring 'human likeness' remains a challenge for evaluation; many of the tasks we considered for this did not work well at the BabyLM scale, and the remaining few tasks we do have turned out to be brittle and prone to noisy artifacts. Neither Strict nor Strict-Small models show a significant effect of NLP Task performance on cognitive task performance. Improving Cognitive evaluation remains an open question that we look forward to addressing in next year's edition.

4.3 Multilingual Track

For the Multilingual Track, we compare model performance cross-lingually. Figure 4 shows language averages against overall multilingual averages. It can be seen in this figure that across all models, performance on Dutch and English is slightly above average, whereas Chinese is below average. Whether this is driven by easier language transfer between English and Dutch requires a broader experimental setup, though. In next year's edition, we will consider ways to incorporate a broader set of languages while keeping the track computationally light enough to make it easy to participate.

4.4 Challenge: Winning Submissions

Strict-Small Track Winner. Satheesh et al. (2026) achieved the highest score on the Strict-Small track with the BabySteps model. These models are based on GPT-BERT. Via various ablation studies, they find that a well-tuned baseline model can perform exceptionally well without any out-of-the-ordinary interventions to the training data or method. Muon-family optimizers (AdaMuon for Strict-Small, Muon for Strict) and careful learning rate tuning yielded the largest and most reliable gains, whereas curriculum learning, synthetic data, distillation, and data mixing did not reliably improve the overall score.

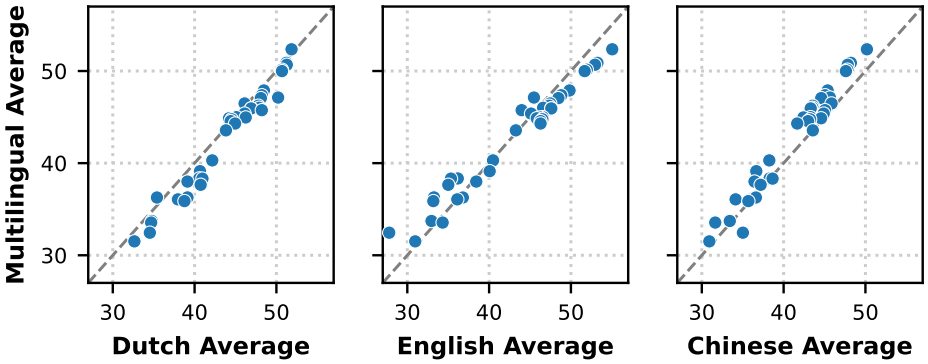


Figure 4: Model performance of the multilingual track submissions, with average language performance set out against overall multilingual average performance. Scores below the diagonal have skewed performance towards that language.

Strict Track Winner. Satheesh et al. (2026) also achieved the highest score on the Strict track. The architecture and methods for their submission to this track were largely identical to those of the strict-small track. We note that RecGPT (Gülbahar et al., 2026) came a close second. It is a decoder-only model that reuses one Transformer block recurrently to spend more computation per token without adding parameters. It also factorizes the vocabulary embeddings and uses a loss term that predicts the next hidden state.

Multilingual Track Winner. Edman and Fraser (2026) win the Multilingual track with Qwenfant-0.4B. Their performance appears to come primarily from data preprocessing: they pack training examples, use an LLM to summarise them into more concise text, and translate the summaries to produce parallel English, Dutch, and Chinese data. In addition, they use a relatively large Qwen-3 architecture and a character-level vocabulary. They also use LoRA-style fine-tuning, rather than the more typical full-rank fine-tuning.

5 Analysis of Workshop Papers

The BabyLM Workshop extended the Challenge's focus on data-efficient language learning to cognitive modeling and related methodological questions, including studies outside the competition's training constraints. The papers examined how multilingual experience, model structure, and training signals affect learning from limited data, and how standard benchmark results relate to more targeted measures of linguistic and developmental behavior.

5.1 Highlighted Workshop Papers

Multimodality. TinyLibrary used illustrated children's books to investigate age-based curricula for small vision-language models (Varghese, 2026). Ordering training material by its intended reader age did not show a reliable advantage over random ordering, suggesting that reader age alone is an insufficient guide to model learning difficulty.

Multilinguality. The release of BabyBabelLM (Jumelet et al., 2026a) and the introduction of the multilingual challenge (Choshen et al., 2026) provided a setting for studying cross-linguistic learning. Workshop papers also addressed language-specific questions of corpus construction. Haider and Can (2026) examined how mixing Modern Standard Arabic and Dialectal Arabic affected learning under different tokenization schemes. Pulipaka and Barak (2026) used IndicTrans2 to translate the English BabyLM corpus into Hindi and Telugu, then trained language models on the resulting corpora, testing translation as a means of extending BabyLM-style training to additional languages.

Tokenization. Badawy et al. (2026) compared BPE with morphology-aware tokenization for Arabic. Their experiments did not show consistent downstream improvements over BPE, even when a segmentation constraint substantially reduced perplexity. Similarly, Li et al. (2026) found that reweighting BPE merges using association measures changed segmentation without reliably improving held-out bits per byte or grammatical evaluation.

Architectures and adaptive training. Relational BabyLM separated object-level and relational information using a Dual Attention Transformer and investigated an auxiliary next-latent prediction objective (Brasoveanu et al., 2026). The architectural change contributed more strongly to grammatical generalization than the auxiliary objective. Another work, ARMOR-DP, adjusted language sampling using per-language validation loss as the weakest language changed during training. Its dynamic sampling achieved a better trade-off between overall performance and balance across languages than a fixed sampling strategy that favored one language (Liao et al., 2026). The Arabic studies of Haider and Can (2026) and Badawy et al. (2026) also show how BabyLM's multilingual extension supports research beyond the languages included in the main Challenge.

5.2 Creative Approaches

Cross-linguistic learning. Theodoropoulos et al. (2026) examined cross-linguistic effects in bilingual models trained on phonemic input. Ren (2026) tested transfer from a previously learned language, finding no uniform advantage for typologically closer language pairs. Mita et al. (2026) introduced typological flexibility into symbolic pretraining, retaining average grammatical gains while avoiding some construction-specific degradations associated with fixed structural priors.

Tokenization and representation. Li et al. (2026) found that reweighting BPE merges using association measures changed segmentation without reliably improving held-out bits per byte or grammatical evaluation. **?** found that prosodically annotated phonemic input produced the strongest taxonomic organization among the input representations they tested. Finegan-Dollak (2026) found that representation standardization improved lexical similarity for models trained on large corpora but reduced performance for models trained at cognitively plausible data scales.

Data composition. Perdijk et al. (2026) introduced StoriesFTW, a synthetic Dutch children's storytelling corpus evaluated using human and model judgments. Vaira et al. (2026) compared BabyLM and encyclopedia corpora at matched word budgets, finding that BabyLM data favored linguistic acceptability while encyclopedia text often favored entity tracking.

Training mechanisms. Related Challenge work by Hwang et al. (2026) examined how language exposure distribution and order shaped multilingual learning. Mixed multilingual training outperformed a curriculum beginning with monolingual exposure in all 15 matched configurations, although each configuration was evaluated with a single run. Veitsman et al. (2026) investigated pretraining on symbolic sequences, including music, probabilistic grammars, and cellular automata. Several sources improved subsequent language-modeling loss, but these improvements did not consistently translate into better downstream linguistic performance. Maheshwari (2026) found that reducing the number of causal-language-modeling updates mitigated late-training declines in grammatical evaluation for RWKV-7 models. Because update count and the learning-rate trajectory remained coupled, the experiments did not isolate the mechanism responsible for this benefit. Related Challenge work also examined practical-life-skills post-training through preference-data selection (Štěpánková et al., 2026).

Architectures and resource efficiency. Workshop papers compared causal, hybrid, and simulated ternary Transformers (Srivastav, 2026), and studied how token mixers and generative objectives interact under restricted data budgets (Laitenberger et al., 2026). Recursive approaches included shared-weight computation for non-autoregressive generation (Oytaç, 2026) and additional tied depth in hybrid recurrent–attention models (Sasidhar, 2026). Related Challenge work investigated whether looped computation could substitute for additional parameters (Fan et al., 2026). HydraLLM varied active model width through nested subnetworks that shared parameters (Kemme et al., 2026). Parameter Golf analyzed techniques from a separate community competition constrained by model-artifact size and training time (Paudel and Sheshappanavar, 2026). Taylor (2026) examined copy mechanisms for data-efficient language modeling. These studies addressed distinct constraints on data, computation, and model size, requiring comparisons that account for their different training and evaluation regimes.

Learning signals and context. Ploner and Akbik (2026) randomized token prediction order to expose models to different conditioning contexts during training. Öztürk et al. (2026) added sentence-

level learning signals, but found that a learned next-sentence representation had limited influence on generation in their controlled tests. Kim et al. (2026) likewise distinguished access to context from its effective use: their small translation models could become strongly dependent on prompt examples without gaining a consistent additional benefit from relevant retrieval. Wasserman and Beauchemin (2026) showed that single-token scoring for a French-only BabyLM was sensitive to tokenizer and answer-template choices, motivating native-language evaluation and controls for scoring artifacts.

Cognitively inspired learning and evaluation. Huda (2026) identified differences in noun frequencies between the training corpora and recorded infant input, examining the developmental plausibility of the training distribution. Ding et al. (2026) probed acquisition-order information in model representations, finding that strong Challenge performance did not necessarily correspond to better recovery of children's word acquisition order.

Other studies tested specific learning behaviors that broad benchmark averages can obscure. Cacioli (2026) observed repetition priming without the emergence of mutual exclusivity in the tested text-only models. Bylinina (2026) used visually grounded embedding initialization to test ostensive word learning, finding improvements in property knowledge that were not reflected in most grammatical benchmarks. Zhang (2026) showed that high aggregate accuracy on Japanese past-tense inflection concealed substantial errors in a rare irregular verb class.

6 Conclusion

We were pleased to see an increase in the overall number of BabyLM participants for this cycle, indicating a growing interest in cognitive modeling and data-efficient pretraining. We decided to include a multilingual track after consulting the BabyLM community, which made it clear that multilingualism was a top priority. We anticipate including the multilingual track in future versions of the challenge, and hope to grow the number of languages offered. In a similar vein, discussions with the community have highlighted model evaluation—particularly generation— as an additional area of interest. We anticipate addressing this in a future iteration of the challenge.

References

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O. Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Mohamed Badawy, Marion Di Marco, and Alexander Fraser. 2026. Bpe vs. morphology-aware tokenization: A case study on arabic language modeling at babylm scale. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Martin Benedikt, Ivan Dostál, and Robin Klubarski. 2026. Does a data2vec auxiliary help small masked language models? a babylm-2026 strict-small study. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Jeremias Bohn, Sebastian Feiertag, Nicolas Raube, Emil Schwenger, and Georg Groh. 2026. Domain-based routing in mixture-of-experts models for babylms. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Adrian Brasoveanu, Ece Takmaz, and Jakub Dotlacil. 2026. Relational attention for data-efficient language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Lisa Bylinina. 2026. Augustinian babylm: What ostensive definition can and cannot teach a small language model. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Jon-Paul Cacioli. 2026. Repetition without exclusivity: Scale sensitivity of referential mechanisms in child-scale language models. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Joseph Yew Choe Chan. 2026. Can higher order attention help small language models? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Tyler A. Chang, Catherine Arnett, Abdelrahman Sadallah, Abdelrahman Eldesokey, Abeer Kashar, Abolade Daud, Abosede Grace Olanihun, Adamu Labaran Mohammed, Adeyemi Praise, Adhikarimayum Meerajita Sharma, Aditi Gupta, Adril Putra Merin, Adwoa Bremang, Afitab Iyigun, Afonso Simplício, Ahmed Essouaied, Aicha Chorana, Akhil Eppa, Akintunde Oladipo, Akriti Kuri, Akshay Ramesh, Aleksei Dorkin, Alfred Malengo Kondoro, Alham Fikri Aji, Ali Eren Çetintaş, Allan Hanbury, Alou Dembele, Alp Niksarli, Álvaro Arroyo, Amin Bajand, Amol Khanna, Ana Chkhaidze, Ana Carolina Condez, Anamaria-Roberta Hartl, Andiswa

Mkhonto, Andrew Hoblitzell, Andrew Tran, Angelos Poulis, Anirban Majumder, Anjali Chaudhary, Anna Vacalopoulou, Annette Kuuipolani Kanahele Wong, Annika Simonsen, Anton Kovalev, Anupam Nayak, Ashvanth S, Ayodeji Lana, Ayu Purwarianti, Bashar Alhafni, Benedict Busole, Bernard Ghanem, Bharti Nathani, Biljana Stojanovska Đurić, Blessing Ogundipe, Bolaotan Agbonile, Bragi Bergsson, Bruce Torres Fischer, Burak Tutar, Burcu Çınar, Cade Kane, Can Udomcharoenchaikit, Chadi Helwe, Chaithra Reddy Nerella, Chen Cecilia Liu, Chiamaka Nwokolo, Christopher Homan, Clément Sampebgo, Cristina España-Bonet, Cynthia Amol, Daeyoep Lee, Dan Saattrup Smart, Dana Arad, Daniil Dzenhaliou, Dasol Choi, David Liu, David Semedo, David Anugraha, Deborah Popoola, Deividas Mataciunas, Delphine Nyaboke, Dennis Owusu, Dhyuthy Krishna Kumar, Diogo Tavares, Diogo Glória-Silva, Divyanshu Goyal, DongGeon Lee, E. Kelly Buchanan, Ebele Nwamaka Anajemba, Egonu Ngozi Grace, Elena Mickel, Elias Herranen, Eliza Acharya, Eman Nisar, Emile Anand, Emmanuel Habumuremyi, Emuobonuvie Maria Ajiboye, Eryawan Presma Yulianrifat, Esther Adenuga, Ewa Rudnicka, Faith Itiola, Faran Taimoor Butt, Fareeha Fayyaz Sheikh, Fathima Thekkekara, Fatima Haouari, Faustin Nsengiyumva, Fenal Ashokbhai Ilasariya, Filbert Aurelian Tjiaranata, Firas Laakom, Francesca Grasso, Francesco Periti, Francesco Orabona, Gbenga Kayode Solomon, Genta Indra Winata, Gia Nghia Ngo, Gloria Udhedhe-oze, Gonçalo Vinagre, Gopi Naga Sai Ram Challagolla, Gorka Urbizu-Garmendia, Gouthami Vadithya, Guijin Son, Gulnaz Abdykadyrova, Gyan Swaroop Mohapatra, Hafeez Ullah, Hafsteinn Einarsson, Hai Hu, Hamidreza Saffari, Hamza Zaidi, Haopeng Zhang, Harethah Abu Shairah, Harry Vuong, Hele-Andra Kuulmets, Hitesh Laxmichand Patel, Houda Bouamor, Hwanjo Yu, Iben Nyholm Debess, İbrahim Ethem Deveci, Ikhlasul Akmal Hanif, Ikhyun Cho, Inês Vieira, Inês Calvo, Isaac Manzi, Ismael Illa Salifou, Ismail Daud, Ismail Yusuf, Itay Itzhak, Ivan Zhelyazkov, Ivan Belashkin, Ivan Spada, Jacob Brinton, Jafar Isbarov, Jaka Čibej, Jan Kocoń, Jan Cuhel, Jauza Krito, Jebish Purbey, Jennifer Za, Jennifer Mickel, Jenny Kunz, Jessica Ratovondranto, Jeyarajalingam Varsha, Jihae Jeong, Jimena Tena Dávalos, Jinu Lee, João Magalhães, John Seon Keun Yi, Jongin Kim, Joseph Chataignon, Joseph Marvin Imperial, Jubeerathan Thevakumar, Judith Land, Julia Alekseenko, Junchen Jiang, Jungwhan Kim, Kairit Sirts, Kamesh R, Kamesh V, Kanda Tshinu, Kätriin Kukk, Kaustubh Ponkshe, Kavsar Huseynova, Ke He, Kenneth Enevoldsen, Kent Joshua Alvarez, Kerem Zaman, Khalil Mrini, Kian Kyars, Komal Gour, Krishnakumar Lainitha, Krister Kruusmaa, Kunal Mukherjee, Kusum Chouhan, Laura Castro, Laura M. Porrino-Moscoso, Lenny Sivi Za Nzambi, Leshem Choshen, Levent Sencan, Lilja Øvrelid, Lisa Alazraki, Loretta Oma Jones, Lovina Ehimen-Ugbede, Luheerathan Thevakumar, Luxshan Thavarasa, Mahnoor Malik, Mamadou K. Keita, Mansi Jangid, Marco De Santis, Marcos Garcia, Marek Šuppa, Mariam D'Ciofalo, Marii Ojastu, Marium Attaullah, Maryam Sikander, Mausami Narayan, Maximos Skandalis, Mehak Mehak, Mehmet İlteriş Bozkurt, Melaku Bayu, Menan Velayuthan, Mhasilenuo Vizo, Michael Leventhal, Michał Marcińczuk, Mina Almasi, Mirna Potočnjak, Mithil Bangera, Mohammadamin Shafiei, Mohiba Ansari, Mridul Sharma, Mrityunjaya Indoria, Mughees Ur Rehman, Muhammad Ravi Shulthan Habibi, Murat Kolić, Murat Barkın Kınay, Nada Galant, Naina Singh Rathore, Naphat Permpredanun, Narada Maugin, Nathalie Norman, Nicholas Kluge Corrêa, Nikola Ljubešić, Nirmal Thomas, Nisansa de Silva, Nisheeth Joshi, Nitish Ponkshe, Nizar Habash, Nneoma Udeze, Noel Thomas, Noémi Ligeti-Nagy, Nouhoum Coulibaly, Odunayo Ogundepo, Odunayo Kareemat Buliaminu, Oghojafor Godswill Fejiro, Okechukwu God'spraise, Olanrewaju Samuel, Olaoye Deborah Oluwaseun, Olasoji Akindejoye, Olga Snissarenko, Onyinye Anulika Chiemezie, Orkun Kınay, Osman Tursun, Oyelade Oluwafemi Joshua, Oyesanmi Fiyinfoluwa, Pablo Rodríguez, Pablo Gamallo, Palak Arora, Pedro Valente, Peter Rupnik, Philip Oghenesuowho Ekiugbo, Prakhar Agarwal, Pramit Sahoo, Prokopis Prokopidis, Pua Niau-Puhipau, Quadri Yahya, Rachele Mignone, Raghav Singhal, Rahul Raja, Ram Mohan Rao Kadiyala, Raphael Merx, Rasmus Larsen, Ratnavel Rajalakshmi, Rishav Ghosh, Romina Oji, Ron Kekeha Solis, Rui Guerra, Rushikesh Zawar, Sa'ad Nasir Bashir, Saeed Alzaabi, Sahil Sandeep, Sai Pavan Batchu, Sai Sandeep Kantareddy, Saleha Muzammil, Salsabila Zahirah Pranida, Sam Buchanan, Samuel Rutunda, Sander Land, Sarah Sulollari, Sardar Ali, Saroj Sapkota, Sarveswaran Kengatharaiyer, Saulius Tautvaisas, Sayambhu Sen, Sayantani Banerjee, Sebastien Diarra, Segun Afolayan, Senthilnathan M, Sewoong Lee, Shaan Shah, Shankar Venkitachalam, Sharifa Djurabaeva, Sharon Ibejih, Shivanya Shomir Dutta, Siddhant Gupta, Silvia Paniagua Suárez, Sina Ahmadi, Sivasuthan Sukumar, Siyuan Song, Snegha A, Sokratis Sofianopoulos, Sona Elza Simon, Sonja Benčina, Sophie Gvasalia, Sphurti More, Spyros Dragazis, Stefan Milosavljević, Stephan P. Kaufhold, Suba S, Sultan Alrashed, Surangika Ranathunga, Taiga Someya, Taja Kuzman Pungeršek, Tal Haklay, Tasi'u Jibril, Tatsuya Aoyama, Tea Abashidze, Terenz Jomar Dela Cruz, Terra Blevins, Themistoklis Nikas, Theresa Idoko, Thu Mai Do, Tilek Chubakov, Tina Munda, Tobiloba Owoeye, Tommaso Gargiani, Uma Rathore, Uni Johannesen, Uwuma Ugwu, Vallerie Alexandra Putra, Vanya Bannihatti Kumar, Varvara Arzt, Vasily Konovalov, Vasudevan Nedumpozhimana, Viktoria Ondrejova, Viktoryia Horbik, Vishnu Vardhan Reddy Kummitha, Vuk Dinić, Walelign Sewunetie, Winston Wu, Xiaojing Zhao, Yacouba Diarra, Yaniv Nikankin, Yash Mathur, Yash Bagla, Yeshil Bangera, Yixi Chen, Yiyuan Li, Yolanda Xavier, Yonatan Belinkov, Zaid Alyafeai, Zhargal Batozargalova, Zhengyang Shan, Zhi Rui Tam, Zilu Tang, Zuzana Nadova, Baber Abbasi, Stella Biderman, David Stap, Duygu Ataman, Fabian Schmidt, Hila Gonen, Jiayi Wang, and

David Ifeoluwa Adelani. 2026. Global piqa: Evaluating commonsense reasoning across 100+ languages and cultures.

Lucas Georges Gabriel Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candance Ross, et al. 2025. Findings of the third babylm challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Cristiano Chesi, Achille Fusco, Guido Formichi, Sarah Rossi, Hanna B. Cygan, Tommaso Sgrizzi, and Asya Zanollo. 2026. Exploring linguistic parameters in lms. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Po-Han Chiang. 2026. Urchin: A horizontal spiking language model for data-constrained pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop.

Chenyi Ding, Zachary Nicholas Houghton, and Masoud Jasbi. 2026. Reading acquisition order out of babylm representations: Child-scale training produces developmentally aligned semantic spaces. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Lukas Edman and Alexander Fraser. 2026. Qwenfant: Train big character-level babylms on concise parallel text. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Tingshuo Fan, Hongtao Mu, Tianyu Zhou, Hansen Liu, and Tao Ji. 2026. Looped gpt-bert: Trading parameters for computation in small language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Catherine Finegan-Dollak. 2026. What helped with huge data hurt with small data: Standardization, representational geometry, and data scale. In *Proceedings of the Second BabyLM Workshop*, pages 379–393. Association for Computational Linguistics.

Sam Furr. 2026. Four kilobytes of memory: A 10m-parameter fixed-state recurrent lm for babylm strict-small. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Serdar Gülbahar, Alexander Fraser, and Lukas Edman. 2026. Recursive transformers for babylm 2026. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Mozafar Haider and Burcu Can. 2026. Arabic babylm: Effects of diglossia and tokenisation. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Patrick Haller, Filipe Laitenberger, Jonas Golde, and Alan Akbik. 2026. From puzzle-solver to language model: Recurrent inductive bias, data scale, and benchmark validity in babylm. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Daichi Hotta, Yohei Kobashi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. 2026. Loop before you leap: What recurrent depth can and cannot buy in data-limited pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Zhuojing Huang, Luise Pohlmann, and Lisa Beinborn. 2026. Introducing code-switched contexts to cognitively-inspired bilingual model training. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Imrul Huda. 2026. How far is the training corpus from real infant input? auditing babylm against day-long home recordings, and what an age-of-acquisition curriculum does not buy. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

JiWoo Hwang, Lukas Edman, and Alexander Fraser. 2026. Language exposure matters: Distribution and order in multilingual babylm pre-training. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Sadri Islam, Tagore Rao Kosireddy, Kevin Li, and Dylan Gaines. 2026. Learning from correction: Data-efficient language model training through adaptive teacher-model feedback. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Soham Jain, Harsh Singh, Divija Dewan, and Atul Dev. 2026. Xpertgpt: Multi-scale sparse expert routing for data-constrained language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, Suchir Salhan, Susana Zhou, Yurii Paniv, Ziyin Zhang, Arianna Bisazza, Alex Warstadt, and Leshem Choshen. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Jeremy Kalfus and Sharvin Goyal. 2026. Babblerlm: Targeting closed-system self-generation at a 10m-word model's own reducible-loss spans. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Wiktor Kamzela, Ondrej Dusek, and Mateusz Lango. 2026. Scaling compute instead of parameters: Exploring iterative transformers for data-efficient language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Oscar Kemme, Janek Haberer, Ali Hojjat, and Olaf Landsiedel. 2026. Hydrallm: Adaptive width slicing for elastic text and vision-language models. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Heesu Kim, Shota Sato, and Mamoru Komachi. 2026. Retrieval without relevance: Prompt dependence in small machine translation models. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Filipe Laitenberger, Patrick Haller, Jonas Golde, and Alan Akbik. 2026. What survives the babylm budget? a controlled study of token mixers and generative objectives. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Zy Li, Phu Han, Trinh Tran, Lam Duong, and Andrew Fang. 2026. Do association-reweighted bpe merge rules help data-efficient language modeling? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Hanyu Liao, Yijing Chen, and Tao Ji. 2026. When the weakest language keeps moving: Armor-dp for multilingual babylm. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in Chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Yash Maheshwari. 2026. Halved clm exposure mitigates late-training degradation in small recurrent language models. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Francois Meyer. 2026. Subword segmental babylms: Learning to tokenise for sample-efficient pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Aftab Hossain Mikat and Tamjid Hasan Fahim. 2026. Dense supervision, gentle decay: Curriculum-guided rebalancing of electra-style rtd and mlm for small-data pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Sebastian Miles. 2026. Weight-tied looped depth for data-constrained pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Ujjwal Mishra and Sudipta Roy. 2026. Spend the wasted epochs: Preference post-training from the saturation checkpoint for sample-efficient language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Masato Mita, Max Müller-Eberstein, Misaki Yamaguchi, and Yohei Oseki. 2026. Linguistically inspired multilingual pre-pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Naoki Mori, Seiji Kyo, and Goda kohki. 2026. Language allocation and targeted data design for a multilingual babylm. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Rafael Mosquera-Gómez, Juan Felipe Rodriguez, Mateo Restrepo, Rodrigo Meireles de Oliveira, and Andrés Felipe González Giraldo. 2026. Parameter-efficient architectures for developmentally plausible pretraining: Optimizing embedding and projection layers under babylm strict-small constraints. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Noor Eidirhn Mustafa, Valdemar Melkersson, and Pierre Nugues. 2026. Moep+: Enhancing modular expert paths for data-efficient language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Doğa Oytaç. 2026. Trlm: Shared-weight recursive reasoning for non-autoregressive language generation. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Ahmet Rıfat Öztürk, Yağız Ekrem Dalar, Nedim Mutlu Sezer, Feyzi Arda Salihoğlu, and Ömer Faruk Aksoy. 2026. Sentence-level foresight in small language models: A diagnostic study of learning signals and calibration. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Alonso Palomino. 2026. Exploring auxiliary objectives through multi-task learning for small-data language model pretraining: A babylm 2026 submission. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Prashanna Mani Paudel and Shivanand Venkanna Sheshappanavar. 2026. Parameter golf: What really works? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Sabijn Perdijk, Gijs Wijnholds, Suzan Verberne, and Max J. van Duijn. 2026. Stories for the win: Creating and evaluating a synthetic Dutch children's storytelling dataset. In *Proceedings of the Second BabyLM Workshop*, pages 450–469. Association for Computational Linguistics.

Max Ploner and Alan Akbik. 2026. Can randomly permuting the lm decoding order boost data-efficiency? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Venkata Naga Sai Vishnu Rohit Pulipaka and Libby Barak. 2026. Can translation bridge the data gap for low-resource languages? a case study of hindi and telugu. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Vinit Ravishankar, Memduh Gökırmak, Lilja Øvrelid, and Erik Velldal. 2019. Multilingual probing of deep pre-trained contextual encoders. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 37–47, Turku, Finland. Linköping University Electronic Press.

Pengfei Ren. 2026. Does typological distance affect cross-lingual transfer? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Elizabeth Granda Rodriguez, David Escobar Ruiz, Adrián Rojas Fernandez, and Andres Felipe Gomez Rojas. 2026. Growing smarter, not bigger: Curriculum pretraining and layer duplication in small language models. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Juan Diego Rodriguez. 2026. Tokens of meaning: Semantically-informed morphological tokenization in low-resource language modeling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A. Haggag, Snegha A, Alfonso Amayuelas, Azril Hafizi Amirudin, Viraat Aryabumi, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Aditya Kumar Dalmia, Abraham Diress, Sharad Duwal, Daniil Dzenhaliou, Daniel Fernando Erazo Florez, Fabian Farestam, Joseph Marvin Imperial, Shayekh Bin Islam, Perttu Isotalo, Maral Jabbarishiviari, Börje F. Karlsson, Eldar Khalilov, Christopher Klamm, Fajri Koto, Dominik Krzemiński, Gabriel Adriano de Melo, Syrielle Montariol, Yiyang Nan, Joel Niklaus, Jekaterina Novikova, Johan Samir Obando Ceron, Debjit Paul, Esther Ploeger, Jebish Purbey, Swati Rajwal, Selvan Sunitha Ravi, Sara Rydell, Roshan Santhosh, Drishti Sharma, Marjana Prifti Skenduli, Arshia Soltani Moakhar, Bardia Soltani Moakhar, Ran Tamir, Ayush Kumar Tarun, Azmine Toushik Wasi, Thenuka Ovin Weerasinghe, Serhan Yilmaz, Mike Zhang, Imanol Schlag, Marzieh Fadaee, Sara Hooker, and Antoine Bosselut. 2024. Include: Evaluating multilingual language understanding with regional knowledge.

Dries Rooryck, Alexander Dazhen Cai, Yonatan Belinkov, David Alvarez-Melis, and Kianté Brantley. 2026. Feeding babylms macaroni: Code-switching curricula cause cross-lingual convergence. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Suchir Salhan, Hongyi gu, Donya Rooein, Diana Galvan-Sosa, Gabrielle Gaudeau, Andrew Caines, Zheng Yuan, and Paula Buttery. 2025. Teacher Demonstrations in a BabyLM's Zone of Proximal Development for Contingent Multi-Turn Interaction. In *Proceedings of the 3rd BabyLM Challenge at the 2025 Conference on Empirical Methods in Natural Language Processing*.

Aditya Sasidhar. 2026. When does depth recursion pay? a parameter-matched study of weight-tied hybrid transformers on 100m unique words. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Siddhi Satheesh, Jorge C. Murillo, and Yubei Chen. 2026. Babysteps at babylm 2026: Stronger optimization and causal baselines for data-constrained pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Noam Siegelman, Sascha Schroeder, Cengiz Acarturk, Hee-Don Ahn, Svetlana Alexeeva, Simona Amenta, Raymond Bertram, Rolando Bonandrini, Marc Brysbaert, Daria Chernova, Sara Fonseca, Nicolas Dirix, Wouter Duyck, Argyro Fella, Ram Frost, Carolina Gattei, Areti Kalaitzi, Nayoung Kwon, Kaidi Lõo,

and Victor Kuperman. 2022. Expanding horizons of cross-linguistic research on reading: The multilingual eye-movement corpus (meco). *Behavior Research Methods*, 54:2843–2863.

Siyuan Song, Zhiheng Qian, Yunhao Zhang, Linyang He, Xiaozhe Ji, Yingxin Lin, Hongao Zhu, Chongtian Shao, Chuhan Lang, Luan Li, et al. 2026. The first chinesebabylm challenge: training data-efficient and cognitively plausible language models for chinese. *arXiv preprint arXiv:2607.10745*.

Arnav Srivastav. 2026. Causal, hybrid, and ternary transformers under a restricted data budget. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Josh Taylor. 2026. Hard-wiring the induction head: The surprising effectiveness of copy mechanisms in data-efficient language modelling. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Nikitas Theodoropoulos, Maria Lymperaiou, and Giorgos Filandrianos. 2026. Cross-linguistic effects in bilingual phoneme babylms. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Filip Trhlík, Andrew Caines, and Paula Buttery. 2026. Bias dynamics in BabyLMs: Towards a compute-efficient sandbox for democratising pre-training debiasing. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 33356–33377, San Diego, California, United States. Association for Computational Linguistics.

Edoardo Vaira, Georgios Chavales, and Pierre Nugues. 2026. Different corpora, different skills? a paired study of small language models. In *Proceedings of the Second BabyLM Workshop*, pages 404–416. Association for Computational Linguistics.

Dheeraj Varghese. 2026. Tinylibrary: Do age-graded curricula help small vision-language models? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Yana Veitsman, Jonas Mayer Martins, Jonathan Lautenschlager, and Lisa Beinborn. 2026. Structural priors for data-efficient language learning. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Yupei Wang. 2026. What does input-dependent depth routing buy in data-limited language modeling? In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*.

Adam Zachary Wasserman and David Beauchemin. 2026. Right tool, right job: Native-language evaluation, tokenizer sensitivity, and methodological findings from a French-only BabyLM. In *Proceedings of the Second BabyLM Workshop*, pages 360–378. Association for Computational Linguistics.

Zichao Wei. 2026. Olmo2 with hybrid mlm-clm training for the babylm 2026 multilingual challenge. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Shaoxiang Wu, Alexander Fraser, and Lukas Edman. 2026. Pairing original text with meaning-preserving rewrites for sample-efficient small-data language model pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Fuming Yang, KAI-HAO YANG, Hsuan Lien, Lukas Edman, and Alexander Fraser. 2026a. Optimizing multilingual babylm: The roles of tokenization, data packing, and curriculum learning. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Yuwei Yang, Shihao Pu, Gefei Fung, and Yiming Yang. 2026b. Learning what can combine and what can substitute: Relational inductive biases for babylm pretraining. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

yizhe xu, Ming Song, DANNI HE, Jianghao Liu, Yitong Wang, and Qing Cai. 2026. Babyconceptlm: Exploring dynamic concept computation under restricted language exposure. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Wen Zhang. 2026. Not all irregularity is equal: Causally isolating a rare failure mode in japanese morphological inflection. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Barbora Štěpánková, Vojtěch Lanz, and Jakub Szymanik. 2026. The impact of preference optimization data selection: Teaching practical life skills to babylm. In *Proceedings of the 2nd BabyLM Workshop at EMNLP 2026*, Budapest, Hungary.

Exploring Auxiliary Objectives through Multi-Task Learning for Small-Data Language Model Pretraining: A BabyLM 2026 Submission

Alonso Palomino
mail@alonsopg.com

Abstract

Pretraining optimizes a language model with self-supervised objectives over raw text, shaping the representations later measured by diagnostic evaluations. Masked language modeling (MLM) predicts hidden tokens from context, combining the distributional hypothesis with the cloze task. Less is known about alternative objectives under small data budgets. Using BabyLM 2026's 10M-token Strict-Small setting, this paper asks whether auxiliary objectives jointly improve grammar, world-knowledge plausibility, entity tracking, and reading prediction, or instead benefit some areas at the expense of others. The experiments compare an MLM-only baseline with several multi-task objective mixtures while holding the corpus, architecture, tokenizer, training budget, and evaluation pipeline fixed. Results are selective: MLM-only is strongest on grammatical minimal-pair judgments, while auxiliary objectives improve entity-state tracking and reading prediction. Within the controlled fast evaluation, single-mask semantic cloze gives the clearest directional world-knowledge plausibility gain; grammar repair does not recover grammatical performance, and broad plausibility classification produces no positive EWoK delta. The central finding is objective-interface alignment: auxiliary objectives transfer most clearly when their loss format resembles the diagnostic interface used at evaluation time. Auxiliary objectives therefore act as targeted steering signals rather than uniform improvements over MLM.

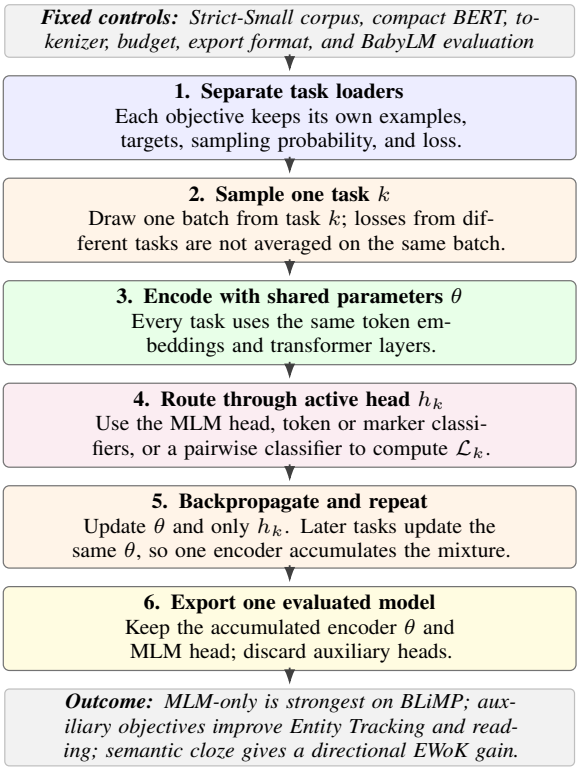


Figure 1: Multi-task training and export. The comparison holds the corpus, model, tokenizer, budget, checkpoint format, and evaluation fixed. Tasks remain separate at the data, target, head, and loss levels, but alternating losses update the same encoder; the submitted artifact retains that encoder and the MLM head.

1 Introduction

The distributional hypothesis motivates pretraining by characterizing linguistic units through context (Harris, 1954). Masked language modeling (MLM) operationalizes this idea by recovering hidden tokens (Devlin et al., 2019), a modern form of the cloze task (Taylor, 1953). Alternative cloze objectives can target function words, discourse connectives, agreement, semantic affordances, and plausibility. Under small data budgets, objective choice determines which contrasts a compact model sees often enough to learn, yet these effects remain underexplored.

Small-data pretraining matters when web-scale corpora or compute are unavailable, including for low-resource languages (Joshi et al., 2020), specialized domains (Gururangan et al., 2020), and on-device deployment (Lu et al., 2025). BabyLM formalizes this concern by evaluating language models trained under restricted, human-scale data

Proceedings of the Second BabyLM Workshop, pages 493–503
October 29, 2026 ©2026 Association for Computational Linguistics

budgets (Warstadt et al., 2023). Its premise is that progress should include closing the data-efficiency gap between human and computational learners, not only scaling data. The 2026 English release provides a 100M-token *Strict* corpus and a 10M-token *Strict-Small* corpus (Choshen et al., 2026); this paper uses Strict-Small throughout.[1] The fixed 10M-token budget makes sample efficiency the experimental target and prevents improvements from being attributed to additional text.

Prior BabyLM submissions explore architectures, data filtering, curricula, optimization, and multi-task supervision for sample-efficient pretraining (Warstadt et al., 2023; Haller et al., 2024; Kamzela et al., 2025). This study instead holds the corpus, architecture, tokenizer, training budget, and evaluation interface fixed while varying the objective mixture. A compact 11.2M-parameter BERT-style model is trained without downstream fine-tuning under MLM-only or multi-task mixtures containing selective MLM objectives, classification objectives, and ranking objectives. The objective mixture is therefore the main intervention. The design asks: *Under the BabyLM Strict-Small data budget, how do auxiliary pretraining objectives change diagnostic performance relative to MLM-only pretraining?* **(RQ)**

The answer is selective improvement. Auxiliary objectives improve entity tracking and reading prediction, and semantic cloze gives the clearest directional world-knowledge plausibility gain in the controlled fast evaluation, but MLM-only remains strongest on grammatical minimal pairs. Targeted tasks also transfer unevenly: grammar repair does not recover grammatical performance, while broad plausibility classification produces no positive fast-evaluation EWoK delta. The contribution is therefore a controlled account of objective-interface alignment: auxiliary objectives steer a compact model toward behaviors aligned with their loss and prediction interface rather than uniformly improving MLM. Figure 1 summarizes this thesis.

2 Related Work

This work builds on three strands of prior research. First, masked language modeling and cloze-style prediction provide the standard pretraining interface for BERT-style encoders (Taylor, 1953; Devlin et al., 2019). MLM is attractive in the BabyLM setting because it requires only raw text and produces checkpoints that can be scored directly by masked-LM diagnostic suites. However, MLM observes only a sampled subset of token prediction targets at each step, so auxiliary losses may change which linguistic contrasts receive frequent supervision.

Second, alternative pretraining objectives show that the shape of the training signal matters. ELECTRA-style replaced-token detection trains discriminators to identify corrupted tokens rather than only reconstructing masked tokens (Clark et al., 2020). Domain-adaptive and task-adaptive pretraining further show that continued training can shift model behavior toward particular domains or tasks (Gururangan et al., 2020). These results motivate the use of targeted auxiliary objectives, while also making interference a central risk: adding a useful loss can reduce exposure to the original MLM objective or train through a head that is not used at evaluation time.

Third, BabyLM work has explored ways to improve learning under restricted data. The original BabyLM Challenge emphasized developmentally plausible, sample-efficient pretraining (Warstadt et al., 2023), while later systems examined compact recurrent architectures and other training recipes (Haller et al., 2024). Most directly related to this paper, Kamzela et al. (2025) study multi-task pretraining for BabyLM using LLM-designed study plans and many generated tasks. That work motivates the same broad question, namely whether small models benefit from task-diverse supervision, but it uses a teacher-assisted design process for constructing the training plan. The present work is complementary rather than competitive with that line: it does not use an external teacher model to design a full curriculum or replace the corpus. Instead, it asks a narrower controlled question: what happens when a compact BERT-style MLM is trained on the same Strict-Small corpus and evaluated through the same BabyLM interface while only the objective mixture is varied?

3 Background and Objective Families

Within BabyLM Strict-Small, the pretraining corpus is limited to 10M tokens. This creates pressure to choose not only model architecture and data processing, but also the pretraining signal. The baseline objective in this study is MLM, as used by BERT-style encoders (Devlin et al., 2019). MLM trains a model to reconstruct masked tokens from context and remains a strong default for compact

[1] https://babylm.github.io/

encoder-only models.

Auxiliary objectives are attractive because they can make specific contrasts more frequent than they would be under random token masking. Replaced-token detection, for example, gives dense corruption-discrimination supervision related to ELECTRA-style pretraining (Clark et al., 2020). Discourse and function-word tasks can highlight connectives and determiners. Pairwise objectives can ask the model to prefer natural collocations, grammatical sentences, or semantically plausible contexts. These signals may help diagnostics that depend on discourse tracking, reading alignment, or conceptual plausibility.

The risk is that auxiliary supervision may interfere with MLM. Some tasks use classifier heads rather than the MLM head used by evaluation, and even MLM-compatible tasks can change which behaviors are emphasized during training. For this reason, aggregate scores are less informative than the pattern of gains and losses across benchmarks. The analysis therefore treats each objective family as a possible steering signal for different diagnostic behaviors. The remainder of this section is organized definitions-first. It first groups the auxiliary tasks by loss interface, then defines the task names used in the paper, and only then describes how examples are generated. The objectives fall into three implementation types. *Selective MLM* keeps the MLM head but applies it to targeted tokens, such as function words or agreement-bearing words. *Classification* uses token-level, marker-position, or `[CLS]` classifiers to label corruptions, discourse/reference markers, collocations, grammaticality, or plausibility. *Ranking* compares two candidate completions or sentence alternatives by asking the model to assign the preferred option a higher score. This taxonomy is used below to separate the linguistic motivation of a task from the loss interface through which it trains the model.

4 Experimental Setup

Pretraining corpus. All experiments use the official BabyLM 2026 Strict-Small training set, `BabyLM-community/BabyLM-2026-Strict-Small`.[2] The dataset card describes it as a detoxified 10M-token English training set and reports six source files: BNC spoken, CHILDES, Gutenberg, OpenSubtitles, Simple Wikipedia, and Switchboard. In the training code, examples are loaded from the `text` column, empty rows are filtered, and the remaining rows are split with a fixed seed into a training portion and 2,000 held-out validation examples. The code repository contains the training, evaluation, and paper artifacts used for the submission.[3]

Tokenization and validation. All approaches use the same `bert-base-uncased` WordPiece tokenizer. After tokenization, the text is concatenated and grouped into fixed-length sequences of 128 tokens. The held-out corpus split is used for validation loss, checkpoint selection, early stopping, and MLM calibration diagnostics; it is not used as a reported BabyLM benchmark score.

Auxiliary-data setup. The auxiliary datasets are built only after fixing this corpus split and model interface. Corpus-derived objectives reuse the Strict-Small text and statistics from it, while template-derived objectives instantiate controlled grammar, plausibility, semantic-cloze, and relational contrasts. Official BabyLM diagnostic items remain evaluation-only: the training procedure does not use BLiMP, BLiMP Supplement, EWoK, Entity Tracking, or reading-prediction benchmark examples. This setup keeps the underlying data source fixed while varying how the same small-data setting is converted into pretraining supervision.

4.1 Objective inventory and generation

Table 1 summarizes the objective families tested. All approaches keep the same compact BERT-style architecture and tokenizer. Most auxiliary data are generated from the BabyLM training corpus or from small synthetic templates. No official BLiMP, EWoK, Entity Tracking, or reading-prediction items are used to construct the auxiliary tasks.

Terminology. Key task names are defined before the generation procedure. *RTD* is replaced-token detection, where a token classifier predicts whether each token is original or corrupted. *Connective* and *definiteness* examples replace discourse connectives or articles with special markers and classify the missing relation or reference type. *Semantic plausibility* is pairwise classification over plausible and implausible event or property contexts. *Semantic cloze* ranks plausible versus implausible

[2] https://huggingface.co/datasets/BabyLM-community/BabyLM-2026-Strict-Small

[3] https://github.com/alonsopg/BabyLM-2026-submission

Family	Loss interface	Task examples	Expected target	Observed effect
MLM-only	Selective MLM	Random masked tokens	General baseline; BLiMP	Strongest BLiMP, weakest Entity and reading scores.
Core multi-task	Classification + MLM	RTD, connective, definiteness, collocation	Entity, reading, discourse/function words	Large Entity and reading gains, large BLiMP loss.
Syntax repair	Selective MLM	Function-word recovery, agreement prediction	BLiMP recovery	Only small BLiMP recovery; Entity and reading remain strong.
Grammar repair	Classification	BLiMP-style generated minimal pairs	BLiMP recovery	Does not recover BLiMP; weak positive fast-evaluation EWoK delta.
Plausibility	Classification	Selectional, affordance, part-whole, agency templates	EWoK and Entity	Improves Entity, but has a negative fast-evaluation EWoK delta.
MLM pair ranking	Ranking	Plausible vs. implausible contexts	EWoK and reading	Improves reading most strongly, but has a negative fast-evaluation EWoK delta.
Semantic cloze	Ranking	Single-mask plausible vs. implausible completion	EWoK	Clearest directional fast-evaluation EWoK gain; still trades off against BLiMP.
Relational cloze	Ranking	Spatial and relational masked-token contrasts	EWoK and Entity	Best Entity score, but lower fast-evaluation EWoK than semantic cloze 5% final.

Table 1: Auxiliary objective families tested in the BabyLM Strict-Small experiments. The loss-interface column groups the many objectives into the three broad mechanisms used in the study: selective MLM, classification, and ranking.

single-token completions with the MLM head, and *relational cloze* applies the same masked-token ranking format to spatial or relation templates.

Generation. Corpus-derived objectives are generated offline from the BabyLM Strict-Small training split. The generator builds token and bigram frequency statistics, masks eligible tokens for MLM, creates RTD and collocation negatives by replacing words with alternatives from the same coarse word class and frequency bucket, and creates connective and definiteness examples by replacing a fixed connective inventory or articles with special markers. The syntax-repair tasks use conservative local filters to mask function words and agreement-bearing tokens. The grammar, conceptual-plausibility, semantic-cloze, and relational-cloze objectives are template-based: templates instantiate grammatical contrasts, plausible versus implausible event roles, affordances, part-whole relations, physical properties, typical locations, and spatial or relational contrasts. For pairwise tasks, the plausible or grammatical option is randomly placed first or second, and the label records which side should be preferred.

Objective families and loss interfaces. The used tasks are grouped into five linguistic families and three loss-interface types, with detailed per-task examples in Appendix A.

1. **Corruption and discrimination objectives** turn contextual fit into a reconstruction or anomaly-detection problem, following the distributional and cloze motivation behind MLM (Harris, 1954; Taylor, 1953; Devlin et al., 2019) and RTD-style pretraining (Clark et al., 2020). For example, MLM predicts *apple* in "The child ate the `[MASK]`," while RTD marks *sky* as corrupted in "The cat sat on the sky."

2. **Discourse and function objectives** target closed-class words whose form is tied to discourse relations, reference, or syntactic structure (Halliday and Hasan, 1976; Prasad et al., 2008; Gundel et al., 1993; Fries, 1952). For example, connective prediction recovers *so* in "It was raining, `[unused1]` the family stayed inside."

3. **Grammatical repair objectives** emphasize agreement and controlled acceptability contrasts (Corbett, 2006; Chomsky, 1957; Warstadt et al., 2020). For example, an agreement item predicts *are* in "The dogs `[MASK]` running," and a minimal-pair item prefers "The dogs run" over "The dogs runs."

4. **Semantic plausibility objectives** train pairwise choices over plausible and implausible contexts, motivated by selectional preferences and world-knowledge plausibility (Resnik, 1996; Ivanova et al., 2025). For example, a classifier prefers the context "The apple was a fruit" over "The apple was a stone" for "The child ate the apple."

5. **MLM-head cloze and ranking objectives** keep semantic supervision close to the masked-LM scoring interface by ranking plausible completions at masked positions (Salazar et al., 2020). For example, semantic cloze ranks *fly* over *read* in "A bird can

[MASK]," while relational cloze ranks *on* over *under* in "The picture is [MASK] the wall."

The semantic and relational cloze objectives use a masked-token ranking loss. For these examples, let s_g and s_b be the MLM log-probabilities of the plausible and implausible tokens at the masked position. The ranking loss is

$$\mathcal{L}_{cloze} = \log(1 + \exp(-(s_g - s_b))). \quad (1)$$

This objective changes pretraining while preserving a normal AutoModelForMaskedLM evaluation interface.

4.2 Training and evaluation protocol

Figure 1 summarizes the experimental control structure. The fixed layers define the comparison space: the corpus, model family, tokenizer, training budget, checkpoint interface, and evaluation suite remain constant. The objective-condition layer is the intervention, so score differences are interpreted as effects of pretraining objective mixture rather than changes in architecture or evaluation protocol.

Model and data control. All comparisons use a compact BERT-style masked language model with hidden size 256, 4 transformer layers, 4 attention heads, intermediate size 1024, maximum position embeddings 512, and the bert-base-uncased WordPiece tokenizer. The exported semantic-cloze checkpoint has 11,201,338 parameters. All runs use the shared corpus and preprocessing described in Section 4; no condition changes the underlying BabyLM Strict-Small data source.

How separate tasks form one model. The tasks are not collapsed into one label space, and separately trained models are not ensembled. Instead, they are integrated by alternating updates to one shared encoder: Thus, "multi-task" refers to the training process, not to the submitted evaluation interface: only one objective is active on a given update, but all updates modify the same encoder.

1. **Keep task data separate.** Each objective has its own JSONL file, data loader, sampling probability, target format, and loss.
2. **Select one task per step.** The trainer samples a task from the configured mixture and draws one batch only from that task. Losses from different tasks are therefore not averaged on the same batch.
3. **Update shared parameters.** The batch passes through the same token embeddings and BERT encoder, then through the head required by the selected task. Backpropagation updates the shared encoder and active head; inactive heads receive no update on that step.
4. **Export one model.** After many alternating steps, the shared encoder has accumulated updates from every sampled objective. Auxiliary heads are removed, leaving that encoder and the MLM head as one standard AutoModelForMaskedLM checkpoint.

The shared encoder is therefore the point of multi-task fusion: task knowledge is combined in one parameter set across training steps, not by combining predictions at evaluation time.

Task-to-head routing. Selective MLM objectives, including ordinary MLM, function-word recovery, and agreement prediction, train through the MLM head on selected masked positions. Ranking objectives, including MLM-head pair ranking, semantic cloze, and relational cloze, also use the MLM head but compare two candidate completions or contexts. Classification objectives use auxiliary heads: RTD uses a token classifier; connective and definiteness prediction use marker-position classifiers; and collocation, grammar minimal-pair, and conceptual plausibility tasks use a two-way classifier over [CLS]. These auxiliary classifiers are training scaffolds and are not included in the submitted checkpoint. The selected semantic-cloze run alternates MLM, RTD, connective, definiteness, collocation, grammar minimal-pair, and semantic-cloze batches; semantic cloze has a configured 5% probability during the mixed-task phase, followed by an MLM-only calibration phase. The semantic cloze 5% final checkpoint is the single official submission artifact; other checkpoints are diagnostic comparisons rather than separate submitted approaches.[4] Figure 1 visualizes this update-and-export process.

Training. Unless otherwise noted, runs use batch size 96, maximum sequence length 128, AdamW

[4]Primary model artifact: https://huggingface.co/alonsopg/babylm-2026-semantic-cloze-strict-small. Additional rerun artifact: https://huggingface.co/alonsopg/babylm-2026-semantic-cloze-aoa-rerun.

with learning rate 5×10^{-4}, weight decay 0.01, linear warmup for 6% of steps, gradient clipping at 1.0, fp32 training, and a 10,000-step budget. Several multi-task runs use a final MLM-only calibration phase or early stopping after step 9000. The semantic cloze 5% run stops at step 9500; its best validation checkpoint occurs at step 5000, while its final checkpoint is the strongest fast-evaluation EWoK checkpoint.

Evaluation benchmarks and metrics. The empirical analysis reports verified local outputs from the official BabyLM strict fast-evaluation suite for each experimental checkpoint.[5] The fast suite is a subsampled diagnostic evaluation used for checkpoint comparison (Warstadt et al., 2023). For candidate-ranking benchmarks, let $s(c_{ij})$ be the summed token log-probability assigned by the MLM backend to candidate j for item i, and let y_i be the correct candidate. Accuracy is $100N^{-1}\sum_i \mathbb{I}[\arg\max_j s(c_{ij}) = y_i]$. The reported benchmarks are:

1. **BLiMP** tests grammatical minimal pairs; the grammatical sentence must receive the higher MLM score (Warstadt et al., 2020).
2. **BLiMP Supplement** uses the same ranking metric for additional BabyLM contrasts such as hypernymy, question answering, inversion, and turn taking (Warstadt et al., 2023).
3. **EWoK** tests conceptual and everyday-world plausibility by asking the model to rank the supported completion above a controlled alternative (Ivanova et al., 2025).
4. **Entity Tracking** evaluates discourse-state consistency: after entity-state changes, the model must prefer the continuation with the correct resulting state (Kim and Schuster, 2023).
5. **Eye Tracking** scores whether model surprisal improves regressions of human gaze durations beyond lexical and contextual controls (de Varda et al., 2024).
6. **Self-Paced Reading (SPR)** applies the same normalized incremental-R^2 scoring to self-paced reading times, including previous-word spillover controls (Frank et al., 2013).

All comparisons keep the model architecture, tokenizer, corpus, and evaluation pipeline fixed. Table 2 summarizes the objective mixtures behind the reported rows. Rows labeled *best* use the held-out MLM-validation checkpoint unless otherwise stated; rows labeled *final* use the terminal or early-stopped checkpoint. The best-by-metric table is a retrospective diagnostic readout, not a checkpoint-selection rule. The controlled comparisons use fast evaluation throughout. Official full evaluation of the selected submission yields 50.45 EWoK accuracy, but no matched full-evaluation baseline is used; this value is therefore descriptive and is not treated as evidence of an EWoK improvement.

Choice of objective weights. The task probabilities in Table 2 are fixed pilot allocations rather than hyperparameters tuned on BabyLM diagnostic scores. They were chosen to preserve ordinary MLM as the majority update source while giving each auxiliary family enough updates to be visible within a 10,000-step budget. The core multi-task run keeps MLM at 60% and assigns the remaining 40% to dense corruption discrimination and two small discourse/reference classifiers. Later variants keep this conservative structure and substitute only a small auxiliary share for the task being tested: syntax repair adds selected function-word and agreement MLM losses, plausibility variants allocate 15% to semantic supervision, and semantic-cloze sweeps vary the cloze share between 2.5% and 10%. This design is intentionally simple, but it creates an unavoidable confound: an auxiliary gain may reflect the auxiliary objective itself, the reduced number of MLM updates, or their interaction. For that reason, the paper interprets the results as diagnostic trade-offs rather than as optimized multi-task recipes. The probabilities should therefore be read as experimental conditions that expose the model to different loss interfaces under one compute budget, not as recommended default weights.

Why sample one task per step. The trainer samples one active objective per update because the tasks have heterogeneous input formats, labels, and heads. Some use token-level MLM labels, some use token corruption labels, some use marker-position classifiers, and some use pairwise sentence classifiers. Sampling one task per step keeps each loss well-defined and avoids hand-tuned loss-scale coefficients across unlike objectives. The cost is higher gradient variance and a potentially spikier optimization path than a design that mixes several losses on each input batch. The final MLM-only

[5] https://github.com/babylm-org/babylm-eval

calibration phase in several runs was included as a simple way to restore exposure to the submitted MLM head after mixed training, but it does not fully remove this limitation.

5 Results

MLM-only is compared against auxiliary multi-task training across all diagnostic families. Two supporting analyses help explain the observed changes: comparisons among selective MLM, classification, and ranking objectives test whether transfer depends on objective format, while comparisons between best-validation and final checkpoints examine whether targeted gains persist across training. These analyses are diagnostic and are not estimates of robust checkpoint-selection performance.

Table 3 gives the raw fast-evaluation scores for representative checkpoints. The main pattern is selective improvement. The key findings are: (i) MLM-only remains strongest for grammar, with the best BLiMP score; (ii) auxiliary approaches strongly improve some non-BLiMP diagnostics, especially Entity Tracking and reading prediction; (iii) objective format matters, since semantic supervision helps fast-evaluation EWoK only when expressed as masked-token cloze ranking; and (iv) no approach dominates all metrics, so the results are a trade-off profile rather than a single uniform improvement. The strongest BLiMP result is the MLM-only baseline at 64.33. Every auxiliary approach loses at least 7.88 BLiMP points relative to that baseline, including objectives explicitly designed to repair grammatical preference. This makes BLiMP the clearest cost of auxiliary training.

At the same time, the auxiliary approaches produce large gains for Entity Tracking and reading. The calibrated multi-task approach improves Entity Tracking from 15.56 to 34.52, Eye from 2.47 to 7.46, and SPR from 2.42 to 3.19. Most later auxiliary variants preserve these gains. The best Entity score is achieved by the relational cloze best checkpoint, 40.68, and the best Eye score by the 2.5% semantic cloze final checkpoint, 8.82.

Within the controlled fast evaluation, EWoK is more selective. The semantic results show that task format matters more than broad semantic motivation. The conceptual plausibility classifier and MLM-head pair-ranking objective are both motivated by EWoK-style plausibility, yet neither produces a positive fast-evaluation EWoK delta. In contrast, semantic cloze ranking directly trains the MLM head to prefer plausible single-token completions over implausible alternatives, matching the masked-token evaluation interface more closely. This is the clearest example of objective-interface alignment in the experiments: unlike broad plausibility classification, semantic cloze trains the same MLM scoring interface later used by EWoK-style candidate ranking. The 5% semantic cloze final checkpoint reaches 52.00 in fast evaluation, 1.45 points above the fast-evaluation MLM baseline and 1.00 above the previous best non-cloze result. Because these are single-seed runs and EWoK scores remain close to chance, this fast-evaluation difference is interpreted as a directional effect rather than a stable estimate of expected improvement. More generally, checkpoint comparisons are used diagnostically rather than as estimates of expected generalization. The semantic-cloze weight sweep and relational variant clarify the trade-off: increasing semantic cloze to 10% gives the strongest non-selected sweep EWoK score of 51.64, while relational cloze produces the strongest Entity Tracking score, suggesting that relation-focused masked-token supervision may strengthen discourse tracking more than EWoK-style plausibility.

Table 4 shows the same pattern as deltas from the MLM baseline. The consistent sign structure is the main finding: fast-evaluation BLiMP deltas are negative for every auxiliary approach, Entity and reading deltas are strongly positive for nearly every auxiliary approach, and the EWoK delta becomes positive only for cloze-shaped objectives. Thus, the central result is that auxiliary objectives move gains toward particular diagnostic families. More specifically, the result supports an objective-interface alignment view: task content alone is not enough, since the transfer pattern also depends on whether the auxiliary loss is expressed through a scoring interface similar to the diagnostic.

To address **RQ**, the results show that auxiliary objectives selectively improve diagnostic performance rather than producing a uniform overall gain. They substantially improve Entity Tracking and reading-aligned metrics, while semantic cloze produces a directional positive EWoK delta in fast evaluation; however, every auxiliary approach reduces BLiMP accuracy relative to MLM-only. The supporting comparisons suggest that objective format helps explain this pattern: grammar minimal-pair classification does not recover BLiMP, conceptual-

Approach	Active objective mixture	Reported checkpoint(s)	Selection note
MLM-only	Random MLM only	final	Baseline MLM checkpoint.
Calibrated multi-task	MLM .60, RTD .20, connective .10, definiteness .10	validation-best	Final 2k steps use MLM-only calibration.
Syntax repair	MLM .60, RTD .10, connective .075, definiteness .075, collocation .05, function-word .05, agreement .05	final; best discussed	Final checkpoint has strongest BLiMP/reading for this run.
BLiMP-pair repair	MLM .60, RTD .10, connective .075, definiteness .075, collocation .05, grammar-minpair .10	validation-best	Final 2k steps use MLM-only calibration.
Conceptual plausibility	MLM .55, RTD .10, connective .075, definiteness .075, collocation .05, conceptual-plausibility .15	validation-best	Classifier-style semantic objective.
MLM-head pair ranking	Same base mixture as conceptual plausibility, replacing the .15 classifier task with MLM-head pair ranking	validation-best	MLM-compatible ranking over plausible/implausible contexts.
Semantic cloze 5%	MLM .60, RTD .10, connective .075, definiteness .075, collocation .05, grammar-minpair .05, semantic-cloze .05	validation-best and final	Final checkpoint gives the strongest observed fast-evaluation EWoK and SPR.
Semantic cloze sweep	Same base as semantic cloze, with cloze weight .025, .075, or .10 and MLM weight adjusted inversely	selected final/best rows	Used to test whether the 5% result is weight-specific.
Relational cloze	MLM .60, RTD .10, connective .075, definiteness .075, collocation .05, grammar-minpair .05, semantic-cloze .025, relational-cloze .025	validation-best and final	Tests whether relation-focused cloze transfers to EWoK or Entity.

Table 2: Configuration summary for reported approaches. Probabilities are fixed task-sampling weights during mixed training, not tuned recommendations. Rows marked validation-best use the held-out MLM-validation checkpoint, not the BabyLM diagnostic score.

Approach	BLiMP	Sup.	EWoK	Entity	Eye	SPR	Main takeaway
MLM-only baseline	**64.33**	54.00	50.55	15.56	2.47	2.42	strongest BLiMP
Calibrated multi-task	54.80	**54.80**	50.27	**34.52**	**7.46**	**3.19**	Entity and reading gains
Syntax repair final	55.51	52.40	49.55	**32.76**	**8.28**	**3.63**	repair does not recover BLiMP
BLiMP-pair repair best	54.80	51.60	**51.00**	**27.54**	**8.02**	**3.05**	weak positive fast EWoK delta
Conceptual plausibility best	55.46	**56.40**	49.64	**37.10**	**7.87**	**3.22**	high Entity, negative EWoK delta
MLM pair ranking best	55.95	47.20	49.55	**33.50**	**8.73**	**3.67**	strong reading, negative EWoK delta
Semantic cloze 5% best	54.10	**57.60**	**50.73**	**36.61**	**7.56**	**3.11**	high Supplement/Entity
Semantic cloze 5% final	55.26	**54.40**	**52.00**	**27.44**	**8.18**	**3.71**	best fast EWoK and SPR
Semantic cloze 2.5% final	54.34	**54.80**	**50.73**	**22.79**	**8.82**	**3.06**	best Eye
Semantic cloze 7.5% best	53.57	52.80	**51.00**	**40.48**	**7.85**	**3.13**	high Entity
Semantic cloze 10% final	55.09	52.80	**51.64**	**33.59**	**8.01**	**3.39**	best non-selected sweep EWoK
Relational cloze best	54.78	53.20	**51.00**	**40.68**	**7.69**	**3.29**	best Entity
Relational cloze final	54.83	52.00	**50.82**	**26.80**	**8.09**	**3.26**	relation final lowers Entity

Table 3: Official strict fast-evaluation results for representative approaches. BLiMP, Supplement, EWoK, and Entity Tracking are accuracy-style percentages. Eye and SPR are official reading-prediction scores. Bold black entries indicate scores that improve over the MLM-only baseline, except for BLiMP where the MLM-only baseline is the strongest observed score.

plausibility classification does not produce a positive EWoK delta, and the strongest fast-evaluation EWoK result appears when semantic supervision is expressed as masked-token cloze ranking through the MLM head.

6 Analysis

Objective/evaluation alignment. The main conclusion is not that auxiliary objectives simply help or hurt. They reallocate a small pretrained model toward the decisions emphasized during training. Under a fixed small-data budget, auxiliary objectives are therefore useful steering signals, but they are not neutral extra supervision. They consume updates that would otherwise train ordinary MLM, and they shape the representation through the interface that receives the loss. This makes objective design a question of objective-interface alignment: an auxiliary task is most useful when its loss, prediction head, and contrast structure resemble the diagnostic behavior one hopes to improve. The strongest gains appear when the auxiliary training signal resembles the diagnostic mechanism used at evaluation time. Semantic cloze is associated with the clearest directional fast-evaluation EWoK gain and uses the MLM head with a masked-token plausibility contrast. MLM-head pair ranking improves reading-oriented scores, plausibly by changing token-level expectations, but does not by itself produce a positive fast-evaluation EWoK delta. Conceptual plausibility improves Entity Tracking but not fast-evaluation EWoK, suggesting that a pairwise classifier over contexts may not transfer to the masked-LM scoring interface.

Approach	Δ BLiMP	Δ Sup.	Δ EWoK	Δ Entity	Δ Eye	Δ SPR
Calibrated multi-task	-9.53	**+0.80**	-0.28	**+18.96**	**+4.99**	**+0.77**
Syntax repair final	-8.82	-1.60	-1.00	**+17.20**	**+5.81**	**+1.21**
BLiMP-pair repair best	-9.53	-2.40	**+0.45**	**+11.98**	**+5.55**	**+0.63**
Conceptual plausibility best	-8.87	**+2.40**	-0.91	**+21.54**	**+5.40**	**+0.80**
MLM pair ranking best	-8.38	-6.80	-1.00	**+17.94**	**+6.26**	**+1.25**
Semantic cloze 5% final	-9.07	**+0.40**	**+1.45**	**+11.88**	**+5.71**	**+1.29**
Semantic cloze 10% final	-9.24	-1.20	**+1.09**	**+18.03**	**+5.54**	**+0.97**
Relational cloze best	-9.55	-0.80	**+0.45**	**+25.12**	**+5.22**	**+0.87**

Table 4: Representative fast-evaluation deltas from the MLM-only baseline. Positive values indicate higher fast-evaluation scores than MLM-only and are shown in bold black; EWoK deltas are interpreted directionally.

Objective effect versus MLM displacement. The experiments do not fully disentangle two mechanisms. One mechanism is *objective content*: an auxiliary task may teach a contrast that ordinary MLM sees rarely, such as discourse connectives or event plausibility. The other is *MLM displacement*: every auxiliary update replaces an update that would otherwise train ordinary MLM. The BLiMP drop is consistent with both explanations. However, the contrast between broad plausibility classification, MLM-head pair ranking, and semantic cloze suggests that the loss interface also matters: all three reduce MLM exposure, but only the single-mask cloze format gives a positive directional fast-evaluation EWoK delta. The safest interpretation is therefore not that a particular loss is generally superior, but that auxiliary objectives produce benchmark-specific trade-offs whose direction depends on both content and interface.

Targeted objectives do not always transfer. The non-transfer cases are as informative as the gains. Function-word and agreement repair improve BLiMP only slightly. A direct BLiMP-style grammatical minimal-pair classifier also does not recover BLiMP, suggesting that template pair classification is not enough to shape the sentence pseudo-likelihood preferences used by BLiMP scoring. Similarly, broad conceptual plausibility supervision shows no positive fast-evaluation EWoK delta unless it is recast as a single-mask cloze ranking objective. The format of the objective matters, not just its linguistic label.

7 Conclusion

Auxiliary objectives in BabyLM Strict-Small are useful, but their benefit is selective. Under a fixed small corpus and compact BERT-style architecture, they improve several non-BLiMP diagnostics: multi-task objectives substantially improve Entity Tracking and reading-aligned metrics, and direct semantic cloze ranking produces the clearest directional EWoK gain in the controlled fast evaluation. The main cost is grammatical minimal-pair performance, where MLM-only remains strongest on BLiMP. Thus, the strongest conclusion is a trade-off: auxiliary objectives do not uniformly improve the model, but they can steer small-data pretraining toward the diagnostic behaviors their losses emphasize. The broader lesson is objective-interface alignment: auxiliary objectives should be designed and evaluated with their target diagnostic interface in mind, because pairwise classifiers, MLM-head ranking, and single-mask cloze losses do not transfer in the same way. Returning to the research question, auxiliary pretraining objectives do change diagnostic performance relative to MLM-only pretraining, but not as a uniform improvement. The practical next step is to test whether the favorable Entity, reading, and semantic-cloze trends persist when mixture weights are swept systematically, MLM exposure is matched more tightly, and multiple seeds are run for the strongest objective formats.

References

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4: Call for papers for the 2026 BabyLM workshop.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Greville G. Corbett. 2006. *Agreement*. Cambridge Text-

books in Linguistics. Cambridge University Press, Cambridge.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

J. R. Firth. 1957. A synopsis of linguistic theory, 1930–1955. In *Studies in Linguistic Analysis*, pages 1–32. Blackwell, Oxford.

Stefan L. Frank, Irene Fernandez Monsalve, Robin L. Thompson, and Gabriella Vigliocco. 2013. Reading time data for evaluating broad-coverage models of English sentence processing. *Behavior Research Methods*, 45(4):1182–1190.

Charles C. Fries. 1952. *The Structure of English: An Introduction to the Construction of English Sentences*. Harcourt, Brace, New York.

Jeanette K. Gundel, Nancy Hedberg, and Ron Zacharski. 1993. Cognitive status and the form of referring expressions in discourse. *Language*, 69(2):274–307.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 82–94, Miami, FL, USA. Association for Computational Linguistics.

M. A. K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(2–3):146–162.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Wiktor Kamzela, Mateusz Lango, and Ondrej Dusek. 2025. You are an LLM teaching a smaller model everything you know: Multi-task pretraining of language models with LLM-designed study plans. In *Proceedings of the First BabyLM Workshop*, pages 469–487, Suzhou, China. Association for Computational Linguistics.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Barbara Landau and Ray Jackendoff. 1993. What and where in spatial language and spatial cognition. *Behavioral and Brain Sciences*, 16(2):217–238.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Wei Liu, Jian Luan, Xiwen Zhang, Nicholas D. Lane, and Mengwei Xu. 2025. Demystifying small language models for edge deployment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14747–14764, Vienna, Austria. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61(1–2):127–159.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Leonard Talmy. 2000. *Toward a Cognitive Semantics, Volume 1: Concept Structuring Systems*. MIT Press, Cambridge, MA.

Wilson L. Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 English

sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*, 56:5190–5213.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

A Detailed Task Examples

This appendix expands the task inventory summarized in the main text. The citations identify the linguistic or psycholinguistic constructs that inspired each generated objective; they are not sources of training examples.

1. **Masked language modeling.** Random corpus tokens are replaced with `[MASK]` and predicted with the MLM head, following distributional and cloze-style word recovery (Harris, 1954; Taylor, 1953; Devlin et al., 2019). Example: "The child ate the `[MASK]`." → *apple*.

2. **Replaced-token detection.** A classifier marks tokens that were replaced by plausible alternatives, following ELECTRA-style corruption discrimination (Clark et al., 2020). Example: "The cat sat on the *sky*." marks *sky* as corrupted.

3. **Connective prediction.** A connective is replaced with `[unused1]` and predicted from context, reflecting discourse cohesion and explicit discourse-relation markers (Halliday and Hasan, 1976; Prasad et al., 2008). Example: "It was raining, `[unused1]` the family stayed inside." → *so*.

4. **Definiteness prediction.** An article is replaced with `[unused2]` and classified as definite or indefinite, motivated by reference and discourse status (Gundel et al., 1993). Example: "A boy found a ball. He kicked `[unused2]` ball." → definite.

5. **Collocation naturalness.** The model chooses the more natural phrase in a pair, motivated by habitual lexical co-occurrence (Firth, 1957). Example: *strong tea* is preferred over *powerful tea*.

6. **Function-word recovery.** Closed-class function words are masked and predicted with the MLM head, targeting syntactic structure encoded by function words (Fries, 1952). Example: "She went `[MASK]` school." → *to*.

7. **Agreement prediction.** Agreement-bearing words are masked and predicted with the MLM head, targeting morphosyntactic dependencies (Corbett, 2006). Example: "The dogs `[MASK]` running." → *are*.

8. **Grammar minimal-pair ranking.** A classifier chooses between generated grammatical and minimally corrupted alternatives, inspired by controlled acceptability contrasts and BLiMP-style minimal pairs (Chomsky, 1957; Warstadt et al., 2020). Example: prefer "The dogs run" over "The dogs runs."

9. **Conceptual plausibility choice.** A classifier chooses the plausible context for a target sentence, motivated by selectional preference and EWoK-style plausibility (Resnik, 1996; Ivanova et al., 2025). Example: for "The child ate the apple," prefer "The apple was a fruit" over "The apple was a stone."

10. **MLM-head pair ranking.** Plausibility pairs are scored through MLM pseudo-likelihood rather than a classifier (Salazar et al., 2020). Example: "The child ate the `[MASK]`." ranks *apple* above *stone*.

11. **Semantic cloze ranking.** The model ranks plausible and implausible single-token completions at one masked position, combining cloze completion with plausibility contrasts (Taylor, 1953; Resnik, 1996; Ivanova et al., 2025). Example: "A bird can `[MASK]`." ranks *fly* above *read*.

12. **Relational semantic cloze.** The same single-mask ranking loss is applied to relation and spatial templates, motivated by spatial and relational semantics (Landau and Jackendoff, 1993; Talmy, 2000). Example: "The picture is `[MASK]` the wall." ranks *on* above *under*.

MoEP+: Enhancing Modular Expert Paths for Data-Efficient Language Modeling

Noor E. Mustafa
Lund University
no6827mu-s@student.lu.se

Valdemar Melkersson
Lund University
va7575me-s@student.lu.se

Pierre Nugues
Lund University
pierre.nugues@cs.lth.se

Abstract

We present MoEP+, an enhanced version of the Modular Expert Paths (MoEP) decoder. MoEP+ retains MoEP's top-k routing over parallel expert paths, simplifies its dimensional mapping layers, and modernizes the backbone with RoPE, RMS normalization, and SwiGLU. It also introduces a projection-specialized routing scheme in which different experts apply nonlinear residual transformations to distinct attention projections. We evaluate MoEP+ on the BabyLM 2026 strict-small track, which is limited to 10 million words, and present it as a combined architectural recipe rather than attributing the observed gains to any single component.

Under a matched training budget, MoEP+ outperforms the unmodified MoEP baseline on all eight zero-shot benchmarks. On five of these benchmarks, the advantage persists even when the baseline is trained for five times as many epochs. Most notably, MoEP+ is substantially more data-efficient: after only two epochs, it surpasses the unmodified MoEP's ten-epoch BLiMP score. Despite its smaller size, it also remains competitive with the official BabyLM 2026 baselines, with its advantages concentrated in the reading-based measures.

1 Introduction

Transformer-based large language models (LLMs), building on the architecture introduced by Vaswani et al. (2017); Radford et al. (2018), require large amounts of training data to approximate language distributions, resulting in substantial computational costs. Sparse Mixture-of-Experts (MoE) architectures, first introduced by Shazeer et al. (2017), can reduce these costs by decomposing a large network into parallel subnetworks that are activated sparsely. Modular Expert Paths (MoEP), introduced by Tapaninaho (2025) as a decoder-only model for the 2025 BabyLM Challenge, is a recent instance of this approach. In this work, we adopt the MoEP implementation as our starting point and investigate how it can be improved.

We revisit MoEP with two observations in mind. First, its backbone uses earlier Transformer components, including learned positional embeddings, LayerNorm, standard MLP feed-forward blocks, and linear query, key, and value projections. Since then, RoPE (Su et al., 2024), RMSNorm (Zhang and Sennrich, 2019), and SwiGLU feed-forward blocks (Shazeer, 2020) have become common in modern decoder architectures. In parallel, Karbevski (2026) and Karbevski and Mijoski (2025) showed that replacing the linear query projection with a nonlinear residual function,

$$Q(X) = \frac{X + f_\theta(X)}{2},$$

can outperform a linear baseline with more nonembedding parameters. Second, MoEP does not explicitly encourage its parallel experts to assume distinct functional roles.

We address both observations by introducing **MoEP+** [1] an enhanced version of MoEP that

1. Simplifies the dimensional mapping layers,
2. Modernizes the backbone with RoPE, RMSNorm, and SwiGLU (see Table 1), and
3. Introduces *projection-specialized experts*.

Different routed paths apply nonlinear residual transformations to different attention projections: some specialize in queries, whereas others specialize in keys or values, as illustrated in Figure 1. Under top-k routing, the model can therefore combine, for example, the output of a nonlinear-query expert with that of a nonlinear-key expert without introducing symmetric residual transformations within

[1] The source code and instructions for obtaining and preprocessing the training data are available at https://github.com/vame-lth/MoEPPlus/tree/main.

Proceedings of the Second BabyLM Workshop, pages 504–513
October 29, 2026 ©2026 Association for Computational Linguistics

the same attention head. This design avoids the unweighted, parameter-free $XX^\top$ term that Karbevski (2026) seeks to prevent by restricting nonlinearity to the query projection alone. This term would otherwise bias the attention matrix toward self-similarity.

Crucially, we introduce these modifications as a unified architectural package and evaluate the resulting model as a whole. Because several components are modified simultaneously, *we do not claim that any single component is responsible for the observed gains*; isolating their individual contributions is left to future work. Our results on the BabyLM strict-small track show that the combined design consistently outperforms the unmodified baseline and is substantially more data-efficient: MoEP+ matches or exceeds the baseline's best BLiMP score after only a fraction of the training.

2 Related Work

2.1 Modern Decoder Components

Several components used in recent decoder-only language models differ from the original Transformer formulation (Vaswani et al., 2017). Rotary Position Embeddings (RoPE) replace learned absolute positional embeddings with a relative position mechanism applied directly to the query and key representations (Su et al., 2024). RMSNorm simplifies LayerNorm by normalizing only with the root mean square of the activations, reducing normalization overhead while preserving training stability (Zhang and Sennrich, 2019). Gated feed-forward networks such as SwiGLU extend the standard Transformer MLP by introducing a multiplicative gating mechanism, which has been shown to improve Transformer performance (Shazeer, 2020).

MoEP+ incorporates these components as part of a single architectural update to the original MoEP backbone. We therefore do not evaluate RoPE, RMSNorm, or SwiGLU as isolated modifications. Instead, they are treated as part of the complete MoEP+ configuration, together with projection-specialized routing and nonlinear attention projections.

2.2 BabyLM and Small-Scale Language Models

The BabyLM challenge is a shared-task, focused on small language models, providing several tracks: The small-track with a 100M word budget, and the strict-small track with a 10M word budget,

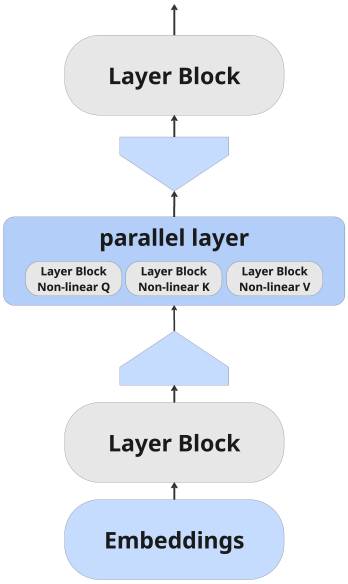


Figure 1: Overall architecture of MoEP+. The model follows a decoder-only Transformer structure with an embedding layer, an initial global Transformer block, a routed parallel layer, a final global Transformer block, and an output layer. The global blocks operate at the full model dimension $d_{\text{model}} = 384$, while the routed parallel layer operates at the reduced parallel dimension 192. The first global block applies nonlinearity to the query projection, the final global block applies nonlinearity to the key projection, and the parallel blocks alternate between query-, key-, and value-specialized nonlinear configurations.

among others. Findings from the BabyLM challenge (Warstadt et al., 2023; Choshen et al., 2026) have demonstrated that, when subject to the mentioned word-budget constraints, architectural innovations can potentially improve performance.

2.3 Modular Expert Paths (MoEP)

The Modular Expert Paths (MoEP) architecture, introduced by Tapaninaho (2025), is a novel language model presented in the third iteration of the BabyLM Challenge and an extension of the Papaformer architecture proposed by Tapaninaho and Oussala (2025). Operating within the mixture-of-experts (MoE) paradigm, MoEP employs top-k routing across parallel computational paths.

While MoEP provides the foundation for our architecture, we introduce several modifications. First, we simplify the dimensional mapping layers. Whereas the original MoEP employs multiple routed linear layers to reduce and restore the dimensionality of representations entering and exiting the parallel blocks, our architecture uses a single

linear projection for each of the down- and up-projections. Furthermore, we introduce nonlinearity into the queries, keys, and values attention projections, SwiGLU feed-forward, RMSNorm and incorporate rotary position embeddings (RoPE) (Su et al., 2024).

2.4 Beyond Linearity

Our exploration of nonlinear attention pathways is motivated by Karbevski (2026), who propose replacing the redundant linear query projection matrix W_Q with a nonlinear residual function of the form

$$Q(X) = \frac{X + f_\theta(X)}{2},$$

where f_θ is a bottleneck MLP with $d^2 + O(d)$ learnable parameters, implemented as

$$f_\theta(X) = \mathrm{LN}(\mathrm{GELU}(\mathrm{LN}(X)W_1)\, W_2)\,,$$

with $W_1 \in \mathbb{R}^{d\times r}$, $W_2 \in \mathbb{R}^{r\times d}$, and $r = d/2$, and where GELU is implemented using the standard hyperbolic tangent approximation,

$$\mathrm{GELU}(x) \approx \tfrac{x}{2}\big(1+\tanh[\sqrt{\tfrac{2}{\pi}}\,(x+0.044715x^3)]\big).$$

This gives $2dr = d^2$ matrix parameters, while the normalization layers add $O(d)$ additional parameters. Their experiments on GPT-3–small–style models showed that this nonlinear query projection outperforms a baseline that uses 12.5% more non-embedding parameters. However, to the best of our knowledge, the interaction between such nonlinear projections and MoE models, such as MoEP, has not previously been investigated.

Furthermore, Karbevski (2026) restricts nonlinearity exclusively to the query projection. The author argues that applying the same residual structure symmetrically to the key projection would introduce an unweighted, parameter-free $XX^\top$ term, thereby biasing the attention matrix toward self-similarity.

Rather than adopting this restriction, MoEP+ distributes nonlinearity across projection-specialized experts, as illustrated in Figures 2 and 1. By assigning query, key, and value nonlinearities to different routed blocks, the model gains access to nonlinear key and value processing without introducing the symmetric $XX^\top$ term within a single attention computation.

Note that our implementation of f_θ differs in that we use LayerNorm in place of the inner RMSNorm and the $\tanh$ approximation of GELU.

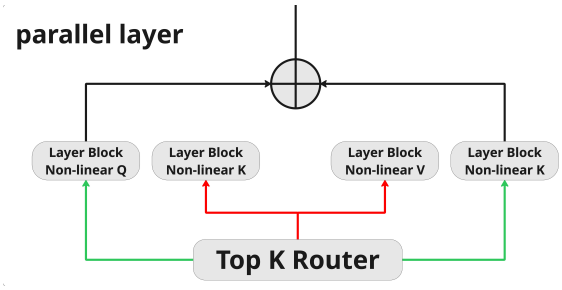


Figure 2: Top-k routing in the MoEP+ parallel layer. With $k = 2$, the router selects two projection-specialized expert blocks per token and combines their outputs using learned routing weights. This allows different nonlinear projection roles, such as query and key or query and value, to be combined without placing all nonlinear projections in the same attention block.

3 Methodology

The projection-specialized routing illustrated in Figures 2 and 1 distributes nonlinearity across the attention projections by assigning distinct projection roles to different blocks. Rather than applying the same nonlinear residual structure to queries, keys, and values within a single attention block, MoEP+ allows different routed components to specialize in different projections. The first large global layer block, see Figure 3, uses a nonlinear query projection while keeping keys and values linear. The final large global layer block uses a nonlinear key projection, while keeping queries and values linear. Inside the intermediate *Parallel-Layer*, see Figure 1, the parallel blocks cyclically alternate between query, key, and value nonlinear configurations:

For a selected projection $P \in \{Q, K, V\}$, the nonlinear residual projection is defined as:

$$P_{\text{nonlin}}(X) = \frac{1}{2}\,(X + f_\theta(X))\,, \qquad (1)$$

while the remaining projections are kept linear. For example, in a key-specialized block,

$$K(X) = \frac{1}{2}\,(X + f_\theta(X))\,, \qquad (2)$$

with

$$Q(X) = XW_Q, \qquad V(X) = XW_V. \quad (3)$$

Top-k sparse routing has previously been used to distribute tokens across a limited subset of experts while maintaining conditional computation (Shazeer et al., 2017; Lepikhin et al., 2021).

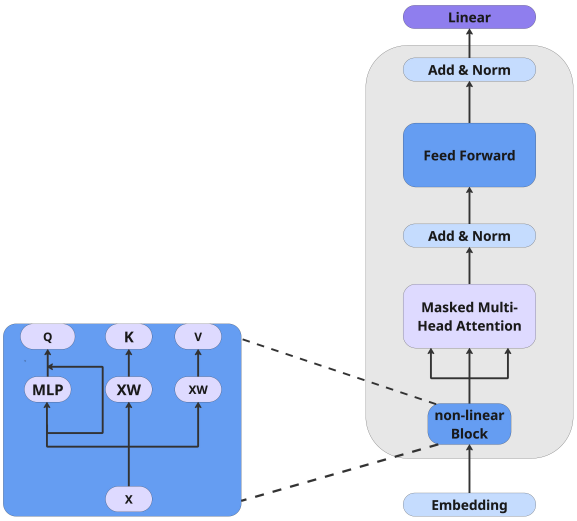


Figure 3: Projection-specialized attention block used in MoEP+. Each block keeps the standard query, key, and value structure, but applies the nonlinear residual projection to only one selected projection path. The shown example applies the nonlinear transformation to the query projection, while keys and values remain linear. Other blocks use the same structure but specialize the nonlinear projection to keys or values instead.

Our intuition is that top-k routing with $k = 2$ can combine outputs from experts with different projection specializations. One selected expert may contribute a nonlinear query representation, while another selected expert may contribute a nonlinear key or value representation; see Figure 2. Since the routed expert outputs are weighted and combined, the model can learn to use different nonlinear attention features without forcing both query and key to use the same residual structure inside a single attention computation. This gives the model access to nonlinear query and key processing while avoiding a direct symmetric residual construction that would explicitly introduce the problematic $XX^\top$ term.

4 Training and Experimental Setup

4.1 Data

Our models are trained on the BabyLM 2026 *Strict-small* (10M-word) track corpus (Choshen et al., 2026). The corpus contains roughly 10M words drawn from six developmentally plausible, predominantly transcribed-speech sources: CHILDES (child-directed speech), the spoken portion of the British National Corpus (BNC), Switchboard, OpenSubtitles, Simple English Wikipedia, and Project Gutenberg. No additional data is used for training, and the same corpus is used for MoEP+ and Unmodified MoEP from Tapaninaho (2025).

4.2 Tokenization

We use the original MoEP article's single shared subword tokenizer for all experiments: a **GPT-2** style byte-pair encoding (BPE) tokenizer (Sennrich et al., 2016) with a vocabulary of 16,384 tokens, trained at the 16K level on the 2025 BabyLM Strict-small track corpus. Sequences are packed to a fixed length of 512 tokens.

4.3 Training

Each variant (MoEP and MoEP+) is trained with the configuration in Table 3. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a peak learning rate of 3×10^{-4} and a cosine schedule with linear warmup over 800 steps. MoEP and MoEP+ are trained for 10 and 2 epochs over the 10M-word corpus. For the MoEP+ model, we stopped training after two epochs, as the model started to overfit. Training is performed in full 32-bit precision on a single Radeon RX 7900 XTX (24 GB) GPU and takes approximately 1.5 hours per model for 10 epochs. Tables 2 and 3 show additional hyperparameters.

To enable analysis of how performance evolves over the course of training, we save checkpoints at fixed word-count budgets following the BabyLM thresholds: every 1M words up to 10M, and then every 10M words up to 100M. Each configuration is trained for three independent runs, differing in their random initialization. The values reported in Tables 4, 5, and 6 are averages over the three runs for both MoEP+ and the unmodified MoEP baseline.

5 Results

The evaluation pipeline follows the BabyLM Challenge 2026 setup and includes both zero-shot and fine-tuned evaluations. Our models were evaluated on the following benchmarks: BLiMP (Warstadt et al., 2020), BLiMP Supplement, COMPS (Misra et al., 2023), Entity Tracking (Kim and Schuster, 2023), EWoK (Ivanova et al., 2025), Reading Eye Tracking, Reading Self-Paced, and Reading Overall. For fine-tuning, we evaluate on tasks drawn from GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a), including BoolQ, MNLI, MRPC, MultiRC, QQP, RTE, and WSC.

Variant	Pos. emb.	Norm	FFN	Params
Unmodified (baseline)	Learned	LN	MLP (multiplier 4)	28.4M
MoEP+	RoPE	RMS	SwiGLU (multiplier 8/3)	28M

Table 1: The baseline (Unmodified MoEP) uses learned positional embeddings, LayerNorm (LN), a standard MLP feed-forward block, and linear $Q/K/V$ projections. Our model replaces these with rotary positional embeddings (RoPE), RMSNorm, and a SwiGLU feed-forward block, and additionally alters the query, key, and value projections.

Hyperparameter	Value
Vocabulary size	16,384
Model dimension d_{model}	384
Parallel dimension	192
Layers	12
Attention heads	6
Head dimension	64
Parallel heads	3
Parallel blocks	4
Number of experts	4
Top-k routing	2
Max sequence length	512
Dropout (embd/resid/attn)	0.1

Table 2: Shared architectural hyperparameters used for MoEP+ and the unmodified MoEP baseline. These settings define the model size, attention configuration, routing setup, context length, and regularization parameters used in the matched-budget experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	3×10^{-4}
Adam β	$(0.9, 0.999)$
Adam ϵ	1×10^{-6}
Weight decay	0.1
Batch size	16
Gradient accumulation steps	1
Training epochs	MoEP: 10, MoEP+: 2
LR scheduler	Cosine
Warmup steps	800
Sequence length	512
Precision	FP32
GPU	RX 7900 XTX
VRAM	24 GB
Runs per configuration	3

Table 3: Training setup and optimization parameters used for MoEP+ and the unmodified MoEP baseline. The configuration includes the optimizer, learning rate, and others.

5.1 Evaluation Protocol

We report results under two comparison settings. First, we compare MoEP+ and the unmodified MoEP baseline under a matched training budget, where both models are evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. This comparison isolates the performance of the two architectures under the same amount of training data.

Second, we compare MoEP+ after two epochs with longer-trained checkpoints of the unmodified MoEP baseline. In this setting, the baseline is evaluated after two, three, six, and ten epochs, corresponding to 20M, 30M, 60M, and 100M words seen during training. This comparison is intended to measure data efficiency: if MoEP+ after two epochs matches or exceeds the unmodified baseline after more training, this suggests that the combined architectural modifications allow the model to reach similar performance with fewer training updates.

All reported MoEP+ and unmodified MoEP values are averages over three independent runs. The unmodified MoEP results reported in this paper are obtained from our own retraining and evaluation setup, using the same data, tokenizer, training schedule, and evaluation pipeline as MoEP+. Therefore, these values may differ from the official MoEP scores reported on the BabyLM 2025 leaderboard (BabyLM-community, 2025). We use our retrained unmodified MoEP model as the primary baseline because it provides a controlled comparison under matched experimental conditions.

For the BabyLM baseline comparisons, we use the scores reported for the `BabyLM Baseline (Interaction)` and `BabyLM Baseline GPT-2` models on the BabyLM 2026 leaderboard as external reference points (BabyLM Community, 2026). These leaderboard scores are not obtained from retraining by us and are therefore used only to contextualize the performance of MoEP+ relative to the official BabyLM 2026 baselines. In contrast, comparisons between MoEP+ and unmodified MoEP are based on our own three-run averages.

5.2 MoEP+ vs. MoEP

As shown in Table 4, we compare MoEP+ with the baseline MoEP model, referred to here as unmodified MoEP. When both models are trained for two epochs (20M words), MoEP+ outperforms unmodified MoEP on all eight zero-shot benchmarks,

including BLiMP and BLiMP Supplement. The advantage largely persists when comparing MoEP+ with unmodified MoEP trained for longer. MoEP+ outperforms the three-epoch baseline (30M words) on seven of eight benchmarks, and it outperforms both the six-epoch (60M words) and ten-epoch (100M words) baselines on five of eight benchmarks. At 60M words, the unmodified baseline exceeds MoEP+ only on COMPS (by 0.27), Entity Tracking (by 0.56), and EWoK (by 0.02). At 100M words, it exceeds MoEP+ on the same three benchmarks, by 0.45, 1.42, and 0.46, respectively.

For the fine-tuning evaluation, both models were fine-tuned from their two-epoch checkpoints. As shown in Table 5, the two models achieve largely comparable results, although our model obtains an average improvement of 0.29 points.

Table 6 compares MoEP+ with two official BabyLM 2026 baselines trained on the same 10M-word strict-small corpus. We focus primarily on the GPT-2 baseline, which obtains the higher fine-tuned macro average of the two baselines. MoEP+ obtains higher scores than this baseline on BLiMP Supplement ($+1.18$) and all three reading-based measures: Reading Overall ($+1.65$), Eye Tracking ($+0.54$), and Self-Paced ($+2.75$). On the remaining benchmarks, MoEP+ matches the GPT-2 baseline on EWoK (50.63), remains within one point on Entity Tracking (-0.32) and COMPS (-0.19), and trails it by 2.23 points on BLiMP (63.00 vs. 65.23). On the fine-tuning tasks, MoEP+ achieves a macro average of 63.60, which is 0.21 points lower than the GPT-2 baseline's score of 63.81.

Relative to the Interaction baseline, MoEP+ obtains higher scores on five of the eight zero-shot benchmarks, including BLiMP Supplement, COMPS, and all three reading-based measures. The two models also achieve nearly identical fine-tuned macro averages (63.60 vs. 63.62). Differences below roughly one point should be interpreted with caution, as we do not report variance estimates for the leaderboard baselines. This applies particularly to the EWoK, Entity Tracking, and COMPS margins against the GPT-2 baseline, as well as the BLiMP margin against the Interaction baseline (63.00 vs. 63.07).

Note: The MoEP+ scores reported on the BabyLM 2026 leaderboard may differ from those presented in our tables because the leaderboard reports the best-performing run, whereas the values in our tables are averaged over three independent runs.

6 Discussion

6.1 Data Efficiency

The clearest and most robust finding is improved data efficiency. After only two epochs, it already exceeds the BLiMP score of the unmodified baseline after 6 epochs, (63.00 vs. 62.12). Thus, MoEP+ achieves a higher BLiMP score using one-third of the training steps. Nevertheless, this faster convergence does not generalize uniformly across tasks: although MoEP+ leads on the reading measures early in training, reading performance declines during extended training, whereas the BLiMP advantage persists. Faster convergence should therefore be interpreted as a BLiMP-specific result rather than a general property across all benchmarks.

6.2 Comparison with the BabyLM 2026 Baselines

Despite using substantially fewer parameters (28M vs. 98.4M), **MoEP+** matches or outperforms the official BabyLM 2026 GPT-2 baseline on five of the eight reported zero-shot benchmarks, as shown in Table 6. MoEP+ outperforms the baseline on BLiMP Supplement and all three reading-based benchmarks, while matching it on EWoK. The largest advantages are observed on Reading Self-Paced and Reading Overall, with gains of 2.75 and 1.65 points, respectively.

On the remaining zero-shot benchmarks, MoEP+ trails the baseline by 0.19 points on COMPS, 0.32 points on Entity Tracking, and 2.23 points on BLiMP. On the fine-tuning tasks, the models achieve similar macro averages (63.60 vs. 63.81). MoEP+ scores higher on MRPC, QQP, and RTE, matches the baseline on WSC, and scores lower on BoolQ, MNLI, and MultiRC. These results suggest that the strengths of MoEP+ are concentrated in reading-based zero-shot measures, whereas its weaknesses are more visible on grammatical acceptability.

The comparison with the Interaction baseline follows a similar pattern, which suggests that this profile reflects a property of MoEP+ rather than a peculiarity of one particular baseline. A fuller investigation of this pattern is left for future work.

6.3 Projection-specialized Routing

Our architecture is motivated by the intuition that top-k routing can combine experts with different nonlinear projection specializations, such as query-, key-, and value-specialized blocks. How-

Average metric	MoEP+ @20M	MoEP @20M	Δ 20M	MoEP @30M	Δ 30M	MoEP @60M	Δ 60M	MoEP @100M	Δ 100M
BLiMP Average	**63.00**	59.14	**+3.86**	60.40	**+2.60**	62.12	**+0.88**	62.04	**+0.96**
BLiMP Supplement Average	**58.43**	55.94	**+2.49**	56.49	**+1.94**	57.86	**+0.57**	58.07	**+0.36**
COMPS Average	51.62	51.02	**+0.60**	51.16	**+0.46**	**51.89**	-0.27	**52.07**	-0.45
Entity Tracking Average	18.78	17.74	**+1.04**	18.80	-0.02	**19.33**	-0.56	**20.20**	-1.42
EWoK Average	50.63	49.15	**+1.48**	49.81	**+0.82**	**50.65**	-0.02	**51.09**	-0.46
Reading Eye Tracking Average	**10.17**	9.50	**+0.67**	9.74	**+0.43**	9.58	**+0.59**	9.38	**+0.79**
Reading Self-Paced Average	**4.39**	4.05	**+0.34**	4.14	**+0.25**	3.84	**+0.55**	3.66	**+0.73**
Reading Overall Average	**7.28**	6.77	**+0.51**	6.94	**+0.34**	6.71	**+0.57**	6.52	**+0.76**
Wins	–	**8/8**		**7/8**		**5/8**		**5/8**	

Table 4: Zero-shot comparison between MoEP+ and the unmodified MoEP baseline. MoEP+ is evaluated at a single training point (20M words, i.e., two epochs), while the baseline is evaluated at four checkpoints (20M, 30M, 60M, and 100M words) to measure data efficiency. Bold Δ values indicate that MoEP+ outperforms the corresponding baseline checkpoint. All scores are averages over three independent runs.

Model	Zero-shot Tasks									Fine-tuned Tasks							Macro
	BLiMP	Supp.	EWoK	Entity	COMPS	Reading	Eye Track.	Self-Paced	AoA	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg
MoEP Unmodified	59.14	55.94	49.15	17.74	51.02	6.77	9.50	4.05	0	65.67	47.72	**82.97**	60.16	**62.14**	**60.43**	**64.10**	28.15 63.31
MoEP+	**63.00**	**58.43**	**50.63**	**18.78**	**51.62**	**7.28**	**10.17**	**4.39**	0	**65.71**	**48.82**	82.28	**64.95**	61.93	58.03	63.46	**29.37** **63.60**

Table 5: Comparison between MoEP+ and the unmodified MoEP baseline on zero-shot and fine-tuned tasks under a matched training budget. Both models are evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. In the Macro Avg column, the upper value is the zero-shot average and the lower value is the fine-tuning average. Reported values are averages over three independent runs.

ever, we do not directly verify this behavior in the present work. In particular, we do not analyze whether the router consistently selects complementary projection-specialized experts, or whether specific combinations such as nonlinear queries with nonlinear keys or values emerge during training. A more detailed analysis of routing patterns, expert utilization, and projection-specialization behavior is therefore left for future work.

6.4 Architectural Takeaways

First, modernizing the MoEP backbone with contemporary components such as RoPE, RMSNorm, and SwiGLU, together with projection-specialized experts, yields consistent gains over the original MoEP, particularly during the early stages of training. Second, projection specialization provides a practical approach to nonlinear attention in a routed model: By assigning query, key, or value nonlinearity to different experts, the model gains access to nonlinear key and value processing without introducing the symmetric, parameter-free $XX^\top$ term that in part motivated prior work to restrict nonlinearity to the query projection. Third, parallel models can achieve performance comparable to, and in some cases better than, the official BabyLM 2026 baselines.

7 Future Work

Several directions remain open for extending this work:

- Scale the model by testing deeper architectures with more parallel experts. Another direction is to set top-k to three, which could allow the router to select three nonlinear projection paths corresponding to queries, keys, and values.
- Test whether the same data-efficiency effect holds when the model is trained on a larger dataset, such as the 100M-word setting.
- Further analyze the routing mechanism to determine whether the router consistently combines specific nonlinear projection types, such as keys and values or queries and keys.
- Add a skip-connection mechanism in which less important tokens skip heavy attention computation, while larger or more important tokens receive more attention.
- Test projection-specialized routing with and without nonlinear projections in order to determine whether the observed data-efficiency gains are caused by the nonlinear projections themselves or by the routing specialization.

Model	Zero-shot Tasks									Fine-tuned Tasks							Macro
	BLiMP	Supp.	EWoK	Entity	COMPS	Reading	Eye Track.	Self-Paced	AoA	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg
MoEP+	63.00	**58.43**	50.63	18.78	51.62	**7.28**	**10.17**	**4.39**	0	65.71	48.82	**82.28**	64.95	61.93	**58.03**	**63.46**	**29.37** 63.60
BabyLM Baseline (Interaction)	63.07	58.13	**51.45**	**19.62**	51.29	5.10	8.50	1.69	0	**68.01**	**51.20**	81.0	65.31	**64.31**	53.96	61.54	28.76 63.62
BabyLM Baseline GPT-2	**65.23**	57.25	50.63	19.1	**51.81**	5.63	9.63	1.64	-12.15	67.71	49.84	81.37	**65.76**	61.67	56.83	**63.46**	27.64 **63.81**

Table 6: Comparison between MoEP+ and BabyLM baseline models on zero-shot and fine-tuned tasks. MoEP+ is evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. BabyLM baseline scores are taken from the BabyLM 2026 leaderboard and constitute an evaluation after 10 epochs; table names abbreviate the leaderboard entries `BabyLM-2026-Baseline-Strict-Small-Interaction`, `BabyLM-2026-Baseline-GPT2-Strict-Small`. In the Macro Avg column, the upper value is the zero-shot average and the lower value is the fine-tuning average. Reported MoEP+ values are averages over three independent runs.

8 Conclusion

This paper introduced MoEP+, a decoder-only Transformer architecture that extends the unmodified MoEP baseline with a set of architectural modifications: nonlinear attention projections, SwiGLU activations, RMSNorm, and Rotary Position Embeddings (RoPE). The central idea is to combine token-adaptive parallel routing with projection-specialized experts, so that different routed components can be assigned different query, key, or value projection roles.

The empirical results show that this combined architecture improves data efficiency in the BabyLM 2026 strict-small setting. Under a matched training budget, MoEP+ outperforms the unmodified MoEP baseline on all eight zero-shot benchmarks after two epochs of training. More importantly, MoEP+ surpasses the baseline's 6-epoch BLiMP score after only two epochs, showing that the full set of modifications allows the model to reach stronger grammatical performance with substantially less training.

MoEP+ also performs competitively against the official BabyLM 2026 baselines, despite using substantially fewer parameters. Relative to the GPT-2 baseline, it outperforms the baseline on four of the eight zero-shot benchmarks and matches it on EWoK. Relative to the Interaction baseline, it obtains higher scores on five of the eight zero-shot benchmarks. At the same time, MoEP+ does not uniformly outperform either baseline, and its gains are concentrated in specific benchmarks rather than across the full evaluation suite.

Overall, the results suggest that the MoEP+ modification package is a promising direction for improving sample efficiency in small language models. However, the present experiments do not isolate which individual component is responsible for the observed improvements. In particular, we cannot conclude whether the gains are caused by the nonlinear projections, the routing specialization, the updated Transformer components, or the interaction between them. Future work should therefore focus on larger-scale experiments, controlled ablation studies, and detailed routing analysis.

Limitations

Unverified expert specialization. The projection-specialized routing mechanism is motivated by the assumption that different experts learn to represent distinct projection nonlinearities and that the router combines them appropriately. However, we do not directly verify whether such specialization emerges. In particular, we report no routing statistics, expert-utilization metrics, or analyses of which experts are selected for different input types.

Limited scale and scope. All experiments are conducted at a single model scale of approximately 28M parameters, using one training corpus from the 10M-word strict-small track and a single tokenizer. It therefore remains unclear whether the observed improvements, particularly the gains in data efficiency, generalize to larger model sizes or training budgets.

Acknowledgments

This work was partially supported by *Vetenskapsrådet*, the Swedish Research Council, registration number 2021-04533.

References

BabyLM-community. 2025. BabyLM Leaderboard 2025: All Tasks. https://huggingface.co/spaces/BabyLM-community/

babylm-leaderboard-2025-all-tasks. Hugging Face Space, accessed 8 July 2026.

BabyLM Community. 2026. BabyLM Leaderboard 2026. https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026. Hugging Face Space, accessed 26 June 2026.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Marko Karbevski. 2026. Beyond linearity in attention projections: The case for nonlinear queries. In *ICLR 2026 Workshop GRaM*.

Marko Karbevski and Antonij Mijoski. 2025. Key and value weights are probably all you need: On the necessity of the query, key, value weight triplet in decoder-only transformers. *CoRR*, abs/2510.23912.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. GShard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2920–2941. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Preprint.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Noam Shazeer. 2020. Glu variants improve transformer. *Preprint*, arXiv:2002.05202.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*. OpenReview.net.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Joonas Tapaninaho. 2025. MoEP: Modular expert paths for sample-efficient language modeling. In *Proceedings of the First BabyLM Workshop*, pages 540–547, Suzhou, China. Association for Computational Linguistics.

Joonas Tapaninaho and Mourad Oussala. 2025. Papaformer: Language model from pre-trained parallel paths. *Preprint*, arXiv:2508.00544.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Trans. Assoc. Comput. Linguistics*, 8:377–392.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12360–12371.

Four Kilobytes of Memory: A 10M-Parameter Fixed-State Recurrent LM for BabyLM Strict-Small

Sam Furr
samneillfurr@gmail.com

Abstract

How much language competence can a small recurrent model acquire when it must compress its context into a fixed-size state? We study this question with StateHead, a gated recurrent language model without attention or positional embeddings. Our BabyLM 2026 Strict-Small submission has 9.97M parameters and 2,160 recurrent state values, equivalent to 4.2 KiB in bf16. After 1.1 hours of training on a laptop, it scores 68.2 on BLiMP, compared with 65.2 for the official 98.4M-parameter GPT-2 baseline on the same benchmark suite. This advantage is specific to grammatical discrimination: StateHead is weaker on reading-time prediction and scores near or below chance on several semantic and entity-tracking evaluations. We characterize its computation and training trajectory, and compare recipes that reset or carry state across training chunks. The apparent entity-tracking gain is confined to items whose correct answer is "nothing"; both recipes score below chance on the revised benchmark. These results document both the useful capabilities and the limits of a language model with a very small recurrent memory.

1 Introduction

The BabyLM Challenge asks how much language competence can be learned from a limited amount of experience (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). Its 2026 Strict-Small track allows a corpus of at most 10M words and at most 100M word exposures during model training (Choshen et al., 2026). This setting makes it possible to study architectural choices with modest resources, while testing grammatical, semantic, and processing-related generalization.

We examine one such choice: compressing preceding context into a recurrent state whose size does not grow with the sequence. A transformer with full attention can retain a token-indexed record of its context. A recurrent model instead updates a fixed set of values as each token arrives. This constraint reduces persistent inference memory, but also requires the model to retain useful distinctions without direct access to earlier token representations.

Incremental comprehension provides one motivation for studying this constraint. The Now-or-Never account emphasizes that transient linguistic input must be encoded as it arrives (Christiansen and Chater, 2016), and recurrent networks have long been used to study learning from sequential input (Elman, 1990). The analogy has limits. Human readers can revisit a passage, and a fixed-state text model does not reproduce human memory or eye movements. We treat bounded state as a computational restriction to investigate, rather than evidence of cognitive fidelity. Reading-time prediction is an empirical test of alignment, and our model performs poorly on that test.

StateHead combines a gated recurrent update with a residual language-model backbone. Its recurrence admits parallel scanning during training and carries a vector of state values per layer. We study a 10M-parameter submission and a 98M-parameter probe. Recurrent syntax learning has prior support (Gulordava et al., 2018); our question is how far this particularly small state can support the broader BabyLM suite.

We document the capabilities and limitations of this system, quantify its computation and persistent memory, and report its training trajectory. We also trace an apparent benefit from a state-carry recipe to entity-tracking items removed by an evaluator revision. Our contribution is empirical: it characterizes what this restricted model learns, rather than proposing a new recurrent-cell principle.

Proceedings of the Second BabyLM Workshop, pages 514–524
October 29, 2026 ©2026 Association for Computational Linguistics

	GPT-2	SH-10M	SH-98M
Parameters	98.4M	9.97M	98.7M
Forward FLOPs/token	233.6M	20.0M	197.5M
Attention/scan share	16.4%	0.22%	0.09%
Training FLOPs	—	9.7×10^{15}	9.7×10^{16}
State at 1,024 (bf16)	36.0 MiB	4.2 KiB	17.4 KiB
State at 4,096 (bf16)	144.0 MiB	4.2 KiB	17.4 KiB
Training time	—	1.1 h	5.1 h
Training tokens/s	—	39.8k	9.0k

Table 1: Analytic costs and measured StateHead training times. State sizes assume two bytes per value and exclude parameters, training activations, and workspace. The 4,096-token GPT-2 cache is a size extrapolation beyond its 1,024 learned positions. Training FLOPs use three times forward cost. GPT-2 training time was not measured.

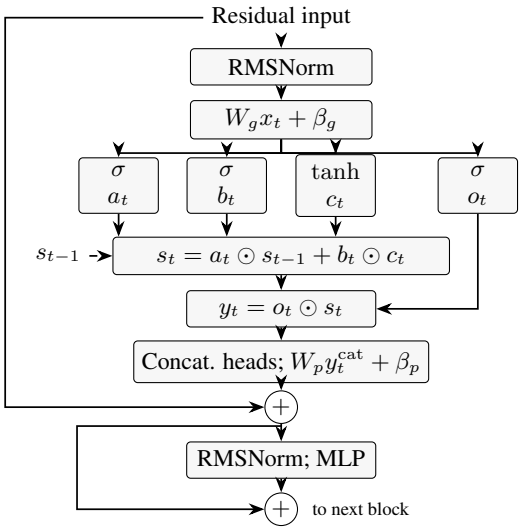


Figure 1: One StateHead block. The fused projection splits into retention, write, candidate, and output branches. Each head updates a vector state; y_t^{cat} concatenates the gated head outputs. The dashed arrow supplies the previous state, and s_t is retained for the next token. The outer paths are residual connections. Stacking L blocks requires Ld recurrent values.

2 StateHead

2.1 Architecture and recurrent update

StateHead uses pre-normalized residual blocks. Each block contains a recurrent token mixer followed by a feed-forward multilayer perceptron (MLP), with a residual connection around each sublayer. The input and output embeddings are tied (Press and Wolf, 2017). There are no positional embeddings: order enters through the recurrent update. Figure 1 shows the block and its gates.

Let d denote the residual width, H the number of heads, and $d_h = d/H$ the width of each head. For a normalized token representation $x_t \in \mathbb{R}^d$, one affine projection produces all four sets of preactivations:

$$\begin{bmatrix} u_{a,t} \\ u_{b,t} \\ u_{c,t} \\ u_{o,t} \end{bmatrix} = W_g x_t + \beta_g, \qquad W_g = \begin{bmatrix} W_a \\ W_b \\ W_c \\ W_o \end{bmatrix}. \quad (1)$$

Here $W_g \in \mathbb{R}^{4d\times d}$ vertically stacks four $d \times d$ matrices; it is not an additional projection before those matrices. The learned bias β_g is partitioned in the same way. Each output is then split into H heads. Each head maintains $s_t \in \mathbb{R}^{d_h}$ and produces $y_t \in \mathbb{R}^{d_h}$. With σ denoting the logistic sigmoid and $\odot$ elementwise multiplication, and suppressing the head index, the update is

$$\begin{aligned} a_t &= \sigma(u_{a,t}), & b_t &= \sigma(u_{b,t}), \\ c_t &= \tanh(u_{c,t}), & o_t &= \sigma(u_{o,t}), \\ s_t &= a_t \odot s_{t-1} + b_t \odot c_t, & y_t &= o_t \odot s_t. \end{aligned} \quad (2)$$

The retention gate a_t scales the previous state, while $b_t \odot c_t$ adds new information. The output gate controls which state coordinates are exposed to the rest of the block. All state operations are elementwise. Head outputs are concatenated into $y_t^{\text{cat}} \in \mathbb{R}^d$ and mixed by the affine projection $W_p y_t^{\text{cat}} + \beta_p$ before the residual addition, with $W_p \in \mathbb{R}^{d\times d}$ and learned bias $\beta_p \in \mathbb{R}^d$.

Retention biases start at 2, giving an initial sigmoid value of about 0.88 at zero input; the other gate biases start at zero. Each layer has a learned initial state used at resets. Each head's gates use all d input coordinates despite its diagonal state transition. At fixed width, parameters, and flattened state, the head partition leaves the mathematical forward computation unchanged; it determines the gate-matrix grouping for head-wise Muon (Section 4.2).

Once the gates are computed, the update is affine in s_{t-1}. Affine updates compose associatively, permitting a parallel prefix scan (Blelloch, 1990; Martin and Cundy, 2018). The implementation uses scan chunks of 64 tokens and fp32 accumulation. This scan chunk size is an implementation parameter, distinct from the 1,024-token training windows described below.

2.2 Computation and persistent memory

Let N be the number of unique trainable parameters, L the number of layers, and T the context length. We count a multiply-accumulate as two

floating-point operations (FLOPs). Table 1 summarizes the accounting; Appendix B gives formulas and a profiler cross-check.

Most computation is in the gate projections, output projections, MLPs, and vocabulary projection. Under serial-equivalent arithmetic accounting, the elementwise recurrence and gate activations contribute approximately 0.22% of forward FLOPs for SH-10M and 0.09% for SH-98M. Our accounting gives a total per-token cost close to $2N$. This follows the dense-matrix convention of Kaplan et al. (2020), but our N includes the tied embedding/output matrix and our cost includes vocabulary logits. These percentages describe the mathematical update, not the exact instructions of a parallel kernel, which can repeat operations. Scan dependencies and memory traffic also affect runtime, so operation share should not be read as runtime share.

StateHead carries Ld values independently of T. SH-10M therefore needs 2,160 values, or 4,320 bytes (4.2 KiB) in bf16. This is persistent sequence state, not total model memory: weights, activations, and temporary buffers are additional. The scan accumulates internally in fp32 and returns state in the numerical precision used for execution. Running in fp32 uses 8.4 KiB for the same state; the exported configuration defaults to fp32.

Full-attention GPT-2 retains $2LTd$ key and value entries. The baseline uses 36 MiB of bf16 cache at $T = 1,024$, and attention adds context-dependent computation. The large FLOP difference between SH-10M and GPT-2 is primarily due to parameter count; removing full attention accounts for a smaller additional saving. Fixed-window attention can also bound memory as a function of total sequence length. StateHead's more restrictive property is that it stores no token-indexed cache at all.

3 Related Work

Gated recurrent models. QRNNs combine temporal convolutions with elementwise recurrent pooling (Bradbury et al., 2017). StateHead's cell uses the same update form as QRNN *ifo* pooling: independent retention and write gates, a bounded candidate, and an output gate. Here the projections use only the current token, equivalent to width-one convolutions, within a pre-normalized residual/MLP backbone. RG-LRU also uses a diagonal transition, but parameterizes retention through a learned decay and scales writes by $\sqrt{1 - a_t^2}$ (De et al., 2024). StateHead does not use that write scaling. Its state is a vector of d scalars per layer; matrix-memory models such as mLSTM store associations through outer-product updates (Beck et al., 2024).

Recurrent models in BabyLM. BLaLM is a close predecessor (Haller et al., 2025). It replaces self-attention with mLSTM and studies short convolutions, sliding-window attention, and other additions. Its 270M-parameter Strict-Small submission reports BLiMP 67.0, EWoK 50.6, and COMPS 50.5. This profile resembles our stronger grammatical discrimination and weak performance on broader semantic tests. The numbers cannot establish a ranking: BLaLM uses a curated 2025 corpus, a larger model, and an earlier evaluator. Its entity-tracking score of 33.7 predates the filtering change in Section 5.3.

Haller et al. also provide controlled evidence about optimization. In their Strict-track experiment, Muon reduces perplexity from 11.21 ± 0.11 to 7.95 ± 0.15 across three runs, while mean zero-shot performance changes more modestly, from 35.75 ± 1.74 to 36.24 ± 1.16. Our use of Muon follows that precedent. We do not establish an additional benefit from our head-wise update rule, which was not ablated.

Context length and recall. Salhan et al. (2025) study OPT and Mamba on the 100M-word Strict corpus and find task-dependent, non-monotonic effects of training length. Their Table 2 distinguishes a training-optimal length, balancing cost and performance, from the length with the highest score. For Mamba entity tracking these are 1,024 and 128, respectively; reading prediction peaks at 64. For OPT, the highest entity score instead occurs at 8,192. Their results therefore do not support a universal preference for long recurrent training contexts. Our 1,024-token setting is within a previously useful range, but its suitability must be checked for StateHead and each task.

Lee et al. (2025) find complementary recall behavior in Mamba–attention hybrids. Sequential hybrids tend to favor shorter-context recall, whereas parallel hybrids with attention-based merging perform better at longer contexts. Those experiments use 430M–1.3B parameters and 100B SlimPajama training tokens, providing architectural evidence rather than a Strict-Small baseline. Their findings motivate studying explicit retrieval. StateHead examines the more restrictive setting without any lo-

cal or global attention cache.

4 Experimental Setup

4.1 Data, model sizes, and training windows

Both models train for ten epochs on the BabyLM 2026 Strict-Small corpus, nominally 100M word exposures. Byte-pair tokenizers (Sennrich et al., 2016) are trained only on that corpus. SH-10M uses an 8,192-token vocabulary; SH-98M uses 16,384 tokens. Tokenizer training does not count toward the model-training exposure allowance (organizer Mustafa Omer Gul, personal communication, July 9, 2026). Training logs record 161.2M and 163.2M token exposures, respectively.

Table 2 gives the configurations. The larger probe approximates the parameter count of the official GPT-2 baseline. The smaller configuration targets an approximately 10M-parameter system and was selected after development experiments with alternative widths, depths, vocabulary sizes, and MLP ratios. It was not chosen through a controlled scaling experiment. Because parameter allocation and tokenization both change, the models test two practical configurations rather than isolate parameter count alone.

The 1,024-token window controls packing, the number of tokens processed together, and the distance through which gradients propagate. Linear-time recurrence does not remove training activation costs or the need to choose a backpropagation window. Inference state can continue beyond that length, but this does not demonstrate accurate recall at arbitrary distances. Development runs also examined windows of 512 and 2,048 tokens; Appendix C describes their narrower scope. Dropout was fixed at zero and was not ablated. We make no claim that either setting is optimal for this data budget.

4.2 Training recipe and checkpoint selection

Training uses MLX (Hannun et al., 2023) on an Apple M5 MacBook Air with a 10-core GPU and 32 GB of unified memory. The microbatch is one window, with 32 accumulation steps, yielding 32,768 tokens per full optimizer update. Parameters and activations use bf16 where supported, and the scan accumulates in fp32. Reported times measure the training loop and exclude tokenizer preparation, initial data loading, checkpoint export, and evaluation.

Setting	SH-10M	SH-98M
Layers L	6	12
Heads H	6	12
Residual width d	360	744
Head width d_h	60	62
MLP expansion r	2	4
Vocabulary V	8,192	16,384
Unique parameters N	9,971,640	98,658,120
Training window T	1,024	1,024
Dropout	0	0
State between windows	Reset	Carry

Table 2: Model configurations. The GELU MLP maps $d \to rd \to d$, so r is the hidden expansion ratio. Both models use RMSNorm, tied embeddings, and learned initial states, without positional embeddings. State carry is detached and resets at document boundaries.

SH-10M concatenates corpus rows without separators and divides the stream into windows. It resets to the learned initial state at every window, including windows that cross a row boundary. The loss therefore covers transitions between rows. SH-98M carries detached state across consecutive chunks within documents and resets at document boundaries. Both shift the packing offset between epochs to vary window boundaries. These differences also confound comparison of the two scales.

Hidden weight matrices and $H \times d_h$ learned initial-state matrices use Muon (Jordan et al., 2024). Token embeddings, tied output weights, normalization scales, and biases use AdamW (Loshchilov and Hutter, 2019). For W_g, the implementation groups update rows by head and applies Muon to H matrices of shape $4d_h \times d$. Each group contains that head's rows from all four gates. Ordinary Muon would process the complete $4d \times d$ matrix. Unit tests verify this grouping against explicit slices, but we did not compare the variants experimentally. Head-wise Muon is a disclosed recipe choice rather than a demonstrated improvement.

We save 19 checkpoints at nominal word-exposure milestones: every 1M through 10M, then every 10M through 100M. Labels follow optimizer-update progress, not a whitespace-word counter. SH-10M is submitted at its final checkpoint. The main SH-98M result uses a weight average of its 60M, 80M, and 100M milestones (Izmailov et al., 2018), selected before full evaluation.

4.3 Evaluation and comparison scope

We use the official 2026 benchmark suite at its July revision (BabyLM Organizing Committee, 2026). Zero-shot tasks include BLiMP (Warstadt

Setting	Value
Schedule	Cosine; 200 warmup steps
AdamW learning rate	5×10^{-4} to 5×10^{-5}
Muon learning rate	2×10^{-2} to 2×10^{-3}
Weight decay	0.1
Muon momentum	0.95, Nesterov
Newton–Schulz iterations	5
Updates (10M / 98M)	4,920 / 4,990

Table 3: Optimization settings for the two main runs. Partial accumulation windows at epoch boundaries account for the small difference between update counts and token exposures divided by 32,768.

Benchmark	GPT-2	SH-10M	SH-98M
Parameters	98.4M	9.97M	98.7M
BLiMP	65.2	68.2	69.9
BLiMP Supplement	57.3	55.3	56.9
EWoK	50.6	50.6	50.1
Entity tracking	19.1	18.8	18.4
COMPS	51.8	50.8	50.9
Reading (ET / SPR)	9.6 / 1.6	0.2 / 0.1	—
Global-PIQA parallel	—	26.2	—
Global-PIQA nonparallel	—	45.0	—
AoA	—	−0.17	—
GLUE (mean accuracy)	—	56.6	—

Table 4: Full evaluation on the July 2026 benchmark suite. SH-10M is the submitted final checkpoint; SH-98M is the 60M/80M/100M weight average. Dashes indicate results not reported under this protocol. Entity tracking uses the revised full set (five choices; chance 20). Global-PIQA chance is 25 for parallel and 50 for nonparallel. Reading scores are normalized incremental $R^2 \times 100$; AoA is acquisition-curve fitness. GLUE uses the official scorer's mean per-task accuracy.

et al., 2020), its Supplement, EWoK (Ivanova et al., 2025), entity tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), and Global-PIQA (Chang et al., 2025). The suite also tests reading-time prediction (de Varda et al., 2024), word-level age of acquisition (Chang and Bergen, 2022), and fine-tuned classification on (Super)GLUE (Wang et al., 2019b,a).

Official dataloaders and scorers drive our MLX zero-shot evaluator. Its scan operations are parity-tested against the PyTorch export. GLUE uses an MLX reimplementation with official task defaults; a check against the official RTE path gives similar results. AoA and the GPT-2 baseline use the official PyTorch evaluator. These checks support the comparison but do not independently reproduce every task in both runtimes.

We evaluate the released GPT-2 checkpoint on the same benchmark suite. Our BLiMP score of 65.2 agrees with the published 65.23. The baseline was not retrained with our tokenizer or optimizer, so the comparison concerns complete systems rather than isolating recurrence. All training runs use one seed. Development used the pipeline's fixed reduced ("fast") subsets; final results use full sets. Fast and full scores are labeled throughout.

5 Results

5.1 Grammatical discrimination and broader evaluation

Table 4 reports full-set evaluation. SH-10M scores 68.2 on BLiMP, 3.0 points above the official GPT-2 checkpoint with approximately one tenth the parameters. The SH-98M weight average scores 69.9. These are descriptive single-run comparisons, without an estimate of variation across training seeds.

The advantage is concentrated in BLiMP. GPT-2 scores higher on the Supplement, COMPS, and entity tracking, although the latter two remain close to chance for all three systems. StateHead's EWoK and Global-PIQA scores also remain close to or below their nominal chance levels. The results support grammatical discrimination under a small parameter budget, but do not establish broad language understanding.

Reading-time prediction gives the clearest contrasting result. GPT-2 reaches 9.6 on eye tracking (ET) and 1.6 on self-paced reading (SPR), while SH-10M reaches 0.2 and 0.1. Recurrence and competent BLiMP performance are therefore insufficient to produce surprisal estimates that predict the measured reading times well in this setup. We have not identified whether architecture, training, tokenization, or their interaction explains this gap.

Fine-tuned classification is also weak. SH-10M obtains 56.6 mean accuracy and exhibits majority-class behavior on several tasks, including near-zero Matthews correlation on BoolQ, QQP, and MultiRC. Its causal objective and single-token pooling are possible contributors, but neither was ablated. Hybrid masked/causal objectives offer a relevant precedent (Charpentier and Samuel, 2024); our results do not determine which change would improve classification.

5.2 Changes across training

Figure 2 shows BLiMP on the fast subsets over 19 milestones. SH-98M rises to 72.6 near four epochs, then declines to 67.0 at the final individual checkpoint. SH-10M reaches 69.0 and finishes at 67.8,

remaining between 67.8 and 69.0 from the nominal 40M milestone onward. Full-set scores differ from fast scores because they evaluate different samples.

This contrast is consistent with overtraining under repeated exposure, but it does not identify a capacity effect: tokenizer, depth, width, MLP ratio, and state handling all differ. A final-checkpoint comparison can therefore conceal substantial differences in trajectory. The smaller model offered a practical submission with good final performance and low training cost. We do not infer that small models are generally more stable or that the larger model's best fast-subset checkpoint would be best on the full suite.

Original entity-tracking scores vary sharply, including jumps from 21.6 to 17.2 to 41.4 over adjacent early SH-98M milestones. These scores predate the filtering change discussed next. We retain the milestone record in Appendix D, but do not interpret its old entity scores as evidence of changes in tracking ability.

5.3 Reassessing a state-carry recipe

State handling determines what can persist across training windows. Resetting at each window discards earlier state; carrying detached state within a document preserves forward information without backpropagating through the preceding chunk. This resembles cached segments in Transformer-XL (Dai et al., 2019) and truncated backpropagation through time (Williams and Peng, 1990).

We compared two earlier 98M training recipes, distinct from the averaged SH-98M model reported in Table 4. Both used cosine decay, but the carry recipe also introduced learned initial states and epoch packing shifts. The reset recipe used a fixed zero state and did not use those shifts. This comparison therefore does not isolate state carry. The recipes gave similar fast-subset language scores: reset/carry results were 66.6/66.2 on BLiMP, 56.0/55.2 on the Supplement, and 48.6/49.0 on EWoK. On the original fast entity benchmark, however, the macro-average rose from 17.0 to 42.0. Taken alone, that difference would suggest a benefit from the combined recipe.

The July 8, 2026 evaluator update (revision `7d0a6c8`) excludes every item for which any candidate answer contains "nothing" (BabyLM Organizing Committee, 2026). In the original fast *regular* split, it removes 914 of 3,152 items (29.0%). We partitioned saved predictions by that exact criterion. Table 5 shows that the gain is confined to the removed group; accuracy on retained items is almost unchanged.

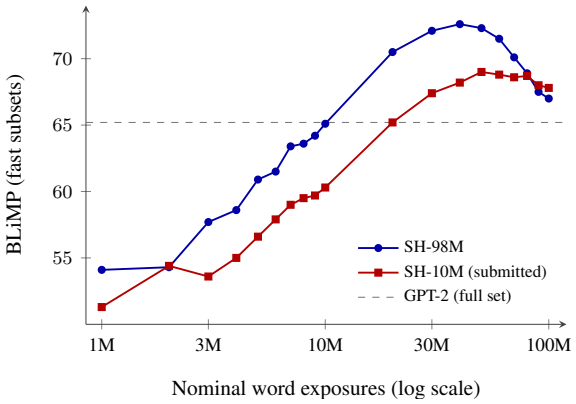


Figure 2: BLiMP on fixed fast subsets across nominal word-exposure milestones. The dashed reference is GPT-2's full-set score, shown for orientation because the samples differ. SH-98M points are individual checkpoints, not the weight average in Table 4.

Original fast-split group	Items	Reset	Carry
Retained by July filter	2,238	18.1	18.0
Removed by July filter	914	15.2	94.9
All original items	3,152	17.3	40.3

Table 5: Per-item accuracies use the same original predictions, without rerunning either model. The headline scores, 17.0/42.0, average six operation-count subtasks. "Reset" and "Carry" denote combined recipes, not a one-factor ablation.

All 914 removed items have "nothing" as the correct answer. The carry model selects it on 867 of them (94.9%). One item describes seven boxes, including "Box 6 contains nothing," then asks "Box 6 contains . . . " Its answers are "nothing," "the suit," "the bread," "the medicine," and "the egg." A rule that chooses "nothing" whenever offered succeeds on the entire removed group without tracking any contents. The increase is therefore compatible with a preference for that answer form. The audit identifies this confound; it does not establish why the training recipe changed the preference or whether answer length caused it.

Fresh evaluation on the revised full set of 6,780 items gives 17.8 for reset and 18.8 for carry. SH-10M scores 18.8 and GPT-2 scores 19.1. None exceeds the nominal five-choice chance level of 20.0. Revised fast scores are 18.4 for both recipes. These are new evaluations with subtask averaging, distinct from the retained-item partition in Table 5.

The revised local evaluation includes an index-alignment fix to the convenience scorer: it must

filter gold items in the same way as the dataloader filters predictions. The patch and upstream revision accompany the reproduction code. The issue was reported to the organizers; leaderboard scoring uses a server-side path.

The evidence removes the apparent tracking benefit of the combined recipe. It does not show that resets are intrinsically better or that state carry is useless. We chose per-window resets for the submission because the extra machinery had no demonstrated benefit in these evaluations. The comparison remains single-seed, and small revised differences should not be interpreted as a reliable ranking.

6 Discussion

StateHead complements BabyLM work on larger recurrent models by showing that a small vector state can support many BLiMP contrasts without positional embeddings. BLaLM's similar qualitative separation between BLiMP and EWoK/COMPS suggests that weak semantic results are not unique to this cell, while differences in data and model size prevent a stronger cross-paper conclusion.

The experiments also delimit what can be attributed to memory. On revised entity tracking, the transformer fails alongside StateHead. These data cannot establish that diagonal recurrence causes the failure. Explicit key–value structure remains a plausible way to improve binding, consistent with research on associative recall and copying (Schlag et al., 2021; Arora et al., 2024; Jelassi et al., 2024). Testing that explanation requires a task and training regime in which a suitable control succeeds.

Sequence length also needs task-specific interpretation. Salhan et al. show that longer training contexts do not uniformly improve reading prediction. Whether longer windows would improve StateHead remains an open question requiring a direct ablation. Lee et al.'s hybrid results suggest benefits from explicit retrieval at larger scales, but do not predict how much benefit would survive the Strict-Small budget. These comparisons motivate controlled follow-up experiments rather than supply explanations established by our data.

7 Conclusion

StateHead shows that a 9.97M-parameter recurrent language model can exceed the official 98.4M GPT-2 checkpoint on BLiMP after approximately 1.1 hours of laptop training. Its recurrent state contains only 2,160 values, and its elementwise update contributes a small fraction of arithmetic operations. The capability profile is uneven: reading-time prediction is poor, and several semantic and tracking evaluations remain near chance. The recipe comparison further shows why a large score difference must be checked at the item level. The resulting submission is a reproducible point of comparison for studying which language capabilities fit within a very small persistent memory.

Limitations

All runs use one seed; the 3.0-point BLiMP margin therefore has no estimate of across-seed variation. Development used fast benchmark subsets, so model and recipe selection were influenced by them. The baseline was evaluated as released rather than retrained with the same tokenizer and optimizer. The two main StateHead configurations differ in more than parameter count, and the earlier reset/carry recipes also change initialization and packing. None of these comparisons isolates an architectural cause.

Historical entity trajectories use the original benchmark and cannot be compared directly with revised full-set results. The item audit identifies an answer-form confound rather than the mechanism by which models acquired their preferences. The older SH-98M GLUE score of 60.0 used a pre-fix fine-tuning pipeline and is omitted, as is its older AoA evaluation. Submission GLUE results use our MLX implementation; every task has not been independently replicated with the official PyTorch trainer.

Head-wise Muon, dropout, training objective, and classification pooling were not separately ablated. Near-zero reading-time performance remains unexplained. Arithmetic and memory figures are analytic estimates with explicit storage conventions. Wall-clock time is measured only for StateHead and does not establish a speedup over GPT-2. Experiments cover one English corpus and one recurrent family, and do not establish long-context accuracy or scaling behavior.

Acknowledgements

The author thanks the BabyLM organizers and reviewers. OpenAI Codex assisted with literature search, text revision, figures, and verification scripts. The author retains full responsibility.

References

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2024. Zoology: Measuring and improving recall in efficient language models. In *International Conference on Learning Representations*.

BabyLM Organizing Committee. 2026. BabyLM 2026 evaluation pipeline. Software repository. Revision 7d0a6c8, 8 July 2026.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2024. xLSTM: Extended long short-term memory. In *Advances in Neural Information Processing Systems*, volume 37, pages 107547–107603. Curran Associates, Inc.

Guy E. Blelloch. 1990. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. Quasi-recurrent neural networks. In *International Conference on Learning Representations*.

Tyler A. Chang, Catherine Arnett, Abdelrahman Sadallah, et al. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *arXiv preprint arXiv:2510.24081*.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Morten H. Christiansen and Nick Chater. 2016. The now-or-never bottleneck: A fundamental constraint on language. *Behavioral and Brain Sciences*, 39:e62.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988. Association for Computational Linguistics.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. 2024. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 English sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*, 56(5):5190–5213.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205. Association for Computational Linguistics.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*, pages 175–191. Association for Computational Linguistics.

Awni Hannun, Jagrit Digani, Angelos Katharopoulos, and Ronan Collobert. 2023. MLX: Efficient and flexible machine learning on Apple silicon. Software.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired

framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*. AUAI Press.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. Repeat after me: Transformers are better than state space models at copying. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 21502–21521. PMLR.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks. Online technical note.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855. Association for Computational Linguistics.

Hyunji Lee, Wenhao Yu, Hongming Zhang, Kaixin Ma, Jiyeon Kim, Dong Yu, and Minjoon Seo. 2025. Understanding and enhancing Mamba-Transformer hybrids for memory recall and language modeling. In *Proceedings of the First BabyLM Workshop*, pages 380–398. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Eric Martin and Chris Cundy. 2018. Parallelizing linear recurrent neural nets over sequence length. In *International Conference on Learning Representations*.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163. Association for Computational Linguistics.

Suchir Salhan, Richard Diehl Martinez, Zébulon Goriely, and Paula Buttery. 2025. What is the best sequence length for BabyLM? In *Proceedings of the First BabyLM Workshop*, pages 130–146. Association for Computational Linguistics.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. 2021. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366. PMLR.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Ronald J. Williams and Jing Peng. 1990. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501.

A Reproducibility

The submitted model, tokenizer, and milestone revisions are available on Hugging Face (haybales/statehead-10m). The code repository contains training and evaluation implementations, configurations, tokenizers, and evaluator patches. The original review snapshot is also available.

The PyTorch export supports the Hugging Face causal-language-model interface with remote model code enabled. Reproduction requires the model-specific configuration and tokenizer rather than current repository defaults. The camera-ready source includes scripts and input hashes for the item audit and FLOP cross-check. Preserve the original training recipes and evaluator revision when reproducing the submitted results.

B FLOPs and Memory Accounting

For L layers of width d, vocabulary size V, and MLP expansion r, the dominant per-token forward cost is

$$F_{\mathrm{SH}} \simeq L\big[(10+4r)d^2 + (4\alpha+4)d\big] + 2dV. \qquad (3)$$

Per layer, the four gate projections cost $8d^2$, the output projection $2d^2$, and the MLP $4rd^2$. The vocabulary projection incurs computation even with tied embeddings. We approximate each nonlinear activation by $\alpha = 4$ operations. The counting tool additionally includes GELU and normalization costs: $L(\alpha rd + 8d) + 4d$. Bias additions and nonlinear-kernel details are not exact hardware operation counts. Vocabulary-output softmax and training loss are outside the forward-logit comparison.

Analytic totals are 19,974,240 and 197,478,432 FLOPs/token for SH-10M and SH-98M. Gate activations and recurrence contribute 43,200 and 178,560, respectively. Full attention adds about $4Td$ matrix-multiplication FLOPs/token/layer, plus a smaller softmax term. The GPT-2 estimate uses 12 layers, width 768, vocabulary 16,384, and 1,024 learned positions.

Profiler cross-check. We instantiated the preserved PyTorch exports with random weights and profiled forward logits on CPU in fp32, using 32 tokens for SH-10M and eight for SH-98M. PyTorch 2.12.1 with `with_flops=True` reports 19,921,680 and 197,191,248 supported FLOPs/token. These differ from the analytic totals by 0.27% and 0.15%, respectively. The matrix-product counts agree exactly. The profiler does not count every elementwise operator, and this check does not measure the repeated work or runtime of the MLX scan kernel.

Training is estimated as three times forward cost per token, giving $3FD$ for D tokens. The runs process exactly 161,198,080 and 163,196,928 tokens. This approximates forward and backward arithmetic, excluding optimizer operations, memory traffic, and system energy.

One sequence requires Ld recurrent values. BF16 storage occupies $2Ld$ bytes: 4,320 for SH-10M and 17,856 for SH-98M. FP32 storage doubles these amounts. The full-attention cache has $2LTd$ values. These counts exclude parameters, training activations, and temporary workspace.

C Context-Length Development Runs

Development runs used windows of 512, 1,024, and 2,048 tokens with the same 98,658,120-parameter configuration and 16k tokenizer. Accumulation factors were 64, 32, and 16, preserving 32,768 tokens/update. The runs used cosine decay, a 4,650-update horizon, and approximately 152.2M token exposures each. All used flat row packing, no separators, and per-window resets, unlike the main document-carry SH-98M probe.

Original fast evaluations included the subsequently removed entity items. Applying the July filter retrospectively to the saved final predictions gives retained-item accuracies of 17.1, 15.9, and 16.5 for the three lengths, respectively. These are item averages from old predictions, not fresh evaluations under the revised pipeline. They provide no evidence that increasing training length to 2,048 rescued entity tracking in those runs. They also do not establish an optimal length for the submitted 10M model; that requires a controlled evaluation on its own recipe and replication across seeds.

D Full Milestone Trajectories

	SH-10M				SH-98M			
Milestone	BLiMP	Sup.	EWoK	Ent.	BLiMP	Sup.	EWoK	Ent.
1M	51.3	50.0	49.5	44.1	54.1	46.4	48.4	43.1
2M	54.4	54.0	47.5	43.1	54.3	55.6	49.1	43.4
3M	53.6	55.6	47.8	43.3	57.7	55.6	50.1	40.5
4M	55.0	56.8	51.5	42.9	58.6	55.2	50.4	41.4
5M	56.6	57.6	50.2	40.1	60.9	58.8	50.1	29.3
6M	57.9	57.2	49.1	32.7	61.5	58.4	50.1	21.6
7M	59.0	55.6	51.4	40.9	63.4	56.4	49.9	17.2
8M	59.5	55.6	49.3	28.1	63.6	58.0	48.1	41.4
9M	59.7	57.2	50.6	31.9	64.2	54.0	49.9	37.5
10M	60.3	56.8	48.9	40.3	65.1	56.4	50.4	41.0
20M	65.2	53.6	48.1	35.2	70.5	54.0	49.7	38.0
30M	67.4	55.6	50.1	35.4	72.1	60.4	50.2	38.6
40M	68.2	54.8	49.6	38.6	72.6	58.4	49.5	35.7
50M	69.0	51.6	50.0	37.8	72.3	59.2	49.7	34.3
60M	68.8	54.4	51.2	35.6	71.5	56.8	49.1	36.2
70M	68.6	56.4	51.8	37.4	70.1	59.6	51.3	31.2
80M	68.7	57.6	49.5	38.0	68.9	58.4	50.1	27.5
90M	68.0	56.8	49.5	38.0	67.5	54.8	51.5	25.0
100M	67.8	58.8	51.1	36.0	67.0	54.0	50.7	24.7

Table 6: All 19 milestones on the original fast subsets. Labels denote nominal cumulative word exposures scheduled by optimizer updates. Entity scores predate the July filter and are historical measurements, not revised-benchmark tracking results. The 98M columns refer to individual checkpoints.

Weight-Tied Looped Depth for Data-Constrained Pretraining

Sebastian Miles
miless@chalmers.se

Abstract

The BabyLM Strict-Small track fixes the training corpus at 10M words and the budget at 10 epochs. Data cannot be added, so any gain has to come from the architecture. We ask whether weight-tied looped depth, applying the same transformer block stack twice per forward pass, helps under that constraint. On a 50M-parameter model built from `nanochat` it does. Held-out bits-per-byte (`val_bpb`, a per-byte cross-entropy that is comparable across tokenizers) falls from 1.3222 to 1.3175 as a three-seed mean, and every looped run lands below every baseline run, at no parameter cost. Downstream the effect points the same way, but we cannot resolve it: averaged over five zero-shot benchmarks under the official harness, the looped mean is 50.00 against the baseline's 49.90, a gap of 0.10 points against a standard error of 0.17. The second pass is paid for in computation. Training takes $1.71\times$ as long, and a 100M-parameter model trained for a comparable wall-clock time still scores better on both metrics. What looping buys here is parameter efficiency, not compute efficiency.

The baseline these experiments run on was tuned by a human-directed agentic loop. The researcher proposed each hypothesis; an agent implemented it, trained for the fixed budget around the clock on one GPU, and kept or reverted the change under a fixed rule on `val_bpb`. We treat the loop as methodology and report its rejected hypotheses alongside the accepted ones.

1 Introduction

The BabyLM challenge (Charpentier et al., 2025) asks how well a language model can perform when it is given a developmentally plausible amount of text. The gap between that budget and the one large language models enjoy is enormous. As the challenge organizers note, Chinchilla (Hoffmann et al., 2022) sees about 1.4 trillion tokens during training, well over 10,000 tokens for every single word a 13-year-old child has heard in their entire life. The Strict-Small track limits the corpus to 10M words and training to 10 epochs, so a model sees at most 100M words in total (Choshen et al., 2026).

That budget leaves the model underprovisioned. Scaling laws (Kaplan et al., 2020) show that model size and data have to grow together, and a model whose size grows alone will overfit. A 50M-parameter model such as ours would ideally see on the order of a billion tokens, roughly twenty per parameter (Hoffmann et al., 2022). Ten epochs over 10M words is an order of magnitude short of that, and re-reading a fixed corpus returns less with each pass (Muennighoff et al., 2023). The same bottleneck is now visible at the frontier: the stock of high-quality human-generated text is finite and, on current trends, is projected to be largely exhausted within the decade (Villalobos et al., 2024). With the corpus fixed, the levers that remain are the architecture and the training recipe.

Weight-tied looped depth. Our main contribution is empirical. In this data-constrained setting, *weight-tied looped depth* lowers validation loss at no parameter cost. The construction is simple: the same transformer stack is applied P times per forward pass, so an L-layer model computes at an effective depth of PL while its parameter count stays that of the L-layer model. It is also old, running from Universal Transformers (Dehghani et al., 2019) and ALBERT (Lan et al., 2020) to a recent wave of looped language models (Giannou et al., 2023; Bae et al., 2025). What we add is the measurement in this regime: re-applying the stack with the same weights improves held-out bits-per-byte at every seed we ran, and the downstream benchmarks move in the same direction, though by less than the spread between seeds. Recent at-scale studies report the opposite trade (Saunshi et al., 2025; Zhu

Proceedings of the Second BabyLM Workshop, pages 525–538
October 29, 2026 ©2026 Association for Computational Linguistics

et al., 2025b): there looping buys a better reasoning inductive bias and gives up perplexity. We attribute the difference to the data budget. Those models are trained close to compute-optimal, where the loss is limited by capacity; ours is mildly underfit on a corpus it has already read ten times, and extra computation still has something to do.

Routing over depth did not help on top of our baseline. We also pursued, at length, a way of enriching how information moves between layers (Pagliardini et al., 2024; Zhu et al., 2025a). In the form we tried, *delta attention residuals* (Luo et al., 2026), each layer attends through learned routing weights over the updates that earlier sublayers wrote into the residual stream ($\mathbf{v}_i = \mathbf{h}_{i+1} - \mathbf{h}_i$). This looked especially promising under the loop, since the second pass can route over the first pass's updates. It was neutral or worse in every run we did, single-pass and looped alike, costing validation loss with no compensating benchmark gain and adding substantially to training time in our implementation (Section 5). We read the result narrowly. Routing has little left to add *on this baseline*, which already carries two cheaper paths that do similar work; that is not evidence against depth routing in the low-data regime in general.

Found and validated by a human-agent research loop. The baseline itself came out of a loop we built by modifying Karpathy's `autoresearch` framework (Karpathy, 2025a), which scheduled training runs and evaluations automatically, around the clock, on a single GPU. The division of labour was fixed: the human researcher proposed the architectural directions, and the agent implemented each change, ran the fixed 10-epoch budget, and reported both validation bits-per-byte and the downstream benchmark results. The default rule was mechanical. A change was kept if `val_bpb` improved beyond a noise threshold and reverted otherwise. The human intervened when `val_bpb` and the benchmarks disagreed, and in those cases the benchmarks decided. We treat the loop as methodology (Section 3.3). It is not the contribution. What it produced is the tuned baseline of Section 4.1, together with a record of everything that failed along the way.

Findings. On a fixed 50M-parameter model and a fixed 10-epoch budget over the 10M-word corpus, weight-tied looped depth lowers `val_bpb` from 1.3222 to 1.3175 as a three-seed mean, and every looped run falls below every baseline run, at *zero* additional parameters. Downstream the picture is weaker. On the average of our five zero-shot benchmarks (BLiMP, BLiMP-supplement, EWoK, COMPS, and entity tracking), scored under the official harness, the looped model gains 0.10 points (50.00 against 49.90, both three-seed means) with a standard error of 0.17. The direction agrees with `val_bpb`, but three seeds cannot resolve an effect that size, and no individual benchmark separates the two models (Section 4.2). Nor does the loop substitute for capacity: a 100M-parameter, depth-12 model reaches 50.19 over two seeds and keeps a clear `val_bpb` advantage (1.2982 against 1.3175), for a comparable wall-clock cost. What a second pass buys, then, is a small and repeatable gain in validation loss that costs parameters nothing and computation a great deal. Against it stands a long list of *rejected* hypotheses: depth routing over sublayers, deep supervision, and many dataset modifications including cleaning, deduplication, and register re-weighting. Each of these matched the baseline or hurt it.

2 Related Work

Weight-tied and looped transformers. Applying one block stack repeatedly to grow effective depth at fixed parameter count has a long lineage. Universal Transformers (Dehghani et al., 2019) recur a single layer in depth with a shared weight matrix, and ALBERT (Lan et al., 2020) shares parameters across all layers to cut model size without losing quality. More recent work studies the loop explicitly: Looped Transformers (Giannou et al., 2023) show that a weight-tied stack iterated in depth can express iterative computation, and Relaxed Recursive Transformers (Bae et al., 2025) recover most of the quality lost to hard weight sharing by adding small layer-wise adapters. A recent wave scales the idea to full language models. Saunshi et al. (2025) and Ouro (Zhu et al., 2025b) pretrain looped models at the billion-parameter scale, and both report the same trade at fixed effective depth: looping strengthens the reasoning inductive bias and pays for it in memorization and perplexity. We measure the opposite sign on perplexity, and Section 1 gives our reading of why.

Dense depth routing and residual mixing. We also studied, and rejected, a new technique to move information between layers. DenseFormer (Pagliardini et al., 2024) adds learned depth-weighted av-

eraging of all previous layer outputs, and Hyper-Connections (Zhu et al., 2025a) generalize the single residual stream to multiple learned connections. Our own model already carries a scalar residual-and-embedding mix at each layer, in the ResFormer and value-residual family (Zhou et al., 2025). The delta attention residuals we tried replace that scalar mix with a per-token softmax over the individual sublayer updates. Under the loop they can also route over an earlier pass's updates, which is a source no single-pass model has. We report the line as a negative result (Section 5): the new cross-pass sources did not turn into a gain, and in this data-constrained regime the plain residual stream proved hard to beat.

Sample-efficient pretraining. The BabyLM challenge (Charpentier et al., 2025) constructs the data-constrained setting and the accompanying zero-shot evals (BLiMP, Warstadt et al., 2020; the BLiMP supplement, Charpentier et al., 2025; EWoK, Ivanova et al., 2025; COMPS, Misra et al., 2023; and entity tracking, Kim and Schuster, 2023). GPT-BERT (Charpentier and Samuel, 2024) shows that mixing causal and masked-language-model objectives improves data efficiency, a technique we adopt in our baseline.

Optimizer and base-model components. We use the Muon optimizer (Jordan et al., 2024) for hidden matrices and AdamW for embeddings and scalars, rotary position embeddings (Su et al., 2024), SwiGLU feed-forward layers (Shazeer, 2020). Two of our baseline's retained wins are also established methods: stochastic weight averaging (Izmailov et al., 2018) and multi-token prediction (Gloeckle et al., 2024; DeepSeek-AI et al., 2025).

Agent-assisted research. The methodology adapts Karpathy's `autoresearch` (Karpathy, 2025a), part of a broader line of work on language-model agents in machine-learning research. We do not claim autonomy: ours is a human-in-the-loop process in which the researcher steers and the agent implements, runs, and analyzes. We use it as a tool for automatically scheduling training and evaluations around the clock.

3 Method and Setup

3.1 Base model and budget

The starting point is a ∼50M-parameter decoder-only transformer with depth L=8, model dimension d=512, head dimension d_h=128, RoPE, SwiGLU MLPs, and the `modded-nanogpt/nanochat` value-embedding mechanism (Karpathy, 2025b), inspired by ResFormer-style value residuals (Zhou et al., 2025). Since the value-embedding mechanism exists only as code in its source (to our knowledge), we state it explicitly. Every second attention layer, with the last layer always included, carries a lookup table $U_i \in \mathbb{R}^{|V|\times d}$ alongside the usual token embedding. In that layer's attention the values are updated as $v \leftarrow v + gu$, where $u := U_i[w]$ is the table entry for the current token id w, and $g := 2\,\sigma(W_i^g\, x_{[:32]}) \in (0,2)^H$ is a learned gate with one scalar per attention head. Here $x_{[:32]} \in \mathbb{R}^{32}$ denotes the first 32 channels of the layer input x, $W_i^g \in \mathbb{R}^{H\times 32}$ is a per-layer learned projection with H the number of attention heads, and σ is the logistic sigmoid; each head's gate scalar scales that head's slice of u. Because u depends on the token id alone, this adds a context-independent per-token component to the attention values, bypassing the residual stream. At each block the input is a learned scalar mix of the running residual x and the embedding skip x_0, the normalized token embedding,

$$\text{base}_i \;=\; \lambda_i\, x \;+\; \mu_i\, x_0, \qquad (1)$$

with learned scalars λ_i, μ_i; the embedding skip is likewise inherited from the `modded-nanogpt` baseline (Karpathy, 2025b). Hidden matrices are optimized with Muon; embeddings, projections, and per-layer scalars are optimized with AdamW. Training is a fixed budget of 10 full passes over the corpus at sequence length 2048. On a single RTX 3060 (12 GB) a single-pass run takes roughly 1 hour, and the submitted looped model roughly 1.9 hours (Section 4.2). Training uses the detoxified BabyLM 2026 Strict-Small corpus (Choshen et al., 2026) (9.99M words; CHILDES, Project Gutenberg, OpenSubtitles, Simple Wikipedia, BNC spoken, and Switchboard) with a fixed Morfessor-based (Virpioja et al., 2013) subword tokenizer (vocabulary 8192, byte fallback).

The submitted model keeps this backbone and differs from the initial baseline in five ways. First, training follows the GPT-BERT hybrid objective (Charpentier and Samuel, 2024), which trains one model on both a causal and a masked objective: 15% of microbatches are trained as a bidirectional masked LM and the remaining 85% causally. Second, a multi-token-prediction head (Gloeckle et al.,

2024) adds an auxiliary loss that predicts the token at $t+2$ from the same final hidden state, giving the representation a second, longer-range target to satisfy. Third and fourth, we hold weight decay constant on the hidden matrices instead of scheduling it, and apply decoupled weight decay to the value-embedding tables, which are large enough to overfit on their own (Section 4.1). Fifth, the weights we evaluate are not the final iterate but a stochastic weight average (Izmailov et al., 2018), the uniform mean of the weights over the final 40% of training. On top of these, the block stack is applied twice per forward pass (Section 3.2). How each of these changes was found, and what each contributes, is traced in Section 4. Full-harness evaluations of this configuration, and of every other configuration we scored that way, are in Appendix A, Table 4.

3.2 Architecture: looped depth, and the routing extension we tested

Weight-tied looped depth. Let the block stack be $B_0, \ldots, B_{L-1}$, where each block represents a standard attention sublayer followed by an MLP sublayer, and let x_0 be the normalized token embedding. A standard forward pass applies each block once. Looped depth instead applies the whole stack P times with the *same* weights, so the effective depth is $P \cdot L$ while the parameter count is exactly that of the depth-L model. At each block the input mix base_i of Section 3.1 is formed as usual, the block is applied to it, and the block's output becomes the running residual x seen by the next block; the P passes simply repeat this over the stack. Re-injecting the input at every iteration is standard in looped and recurrent-depth models (Bai et al., 2019; Geiping et al., 2025). The baseline uses $P{=}1$ and in our submitted model we use $P{=}2$.

Cross-pass delta attention residuals. On top of the loop we evaluated a routing mechanism based on delta attention residuals (Luo et al., 2026), intended to work together with the newly acquired model depth. We describe its form here because our analysis of *why* it fails (Section 5) depends on it. During the looped forward pass we maintain a growing set of *sources*, comprising the embedding x_0 together with every per-sublayer delta

$$v_j \;=\; x_j - x_{j-1}, \qquad j = 1, \ldots, 2LP, \tag{2}$$

where sublayers are indexed consecutively across all P passes and x_{j-1}, x_j denote the residual stream entering and leaving sublayer j. The definition is the same for both sublayer kinds and each sublayer adds its output to the stream, so v_j is the attention output when j is an attention sublayer and the MLP output when j is an MLP sublayer; every block thus contributes two separate deltas per pass, and each is routed over individually. Before routing sublayer s, with input h, a per-token softmax routes over the sources seen so far and adds the mix back into the sublayer input:

$$\begin{aligned} \ell_j &= q_s^\top \,\mathrm{norm}(v_j), \\ h' &= h + \sum_j \mathrm{softmax}_j(\ell)\, v_j, \end{aligned} \tag{3}$$

where v_j is the stored source value and $q_s \in \mathbb{R}^d$ the learned query of routing sublayer s, the only new parameters. Everything is zero-initialized so that the model starts exactly at the looped-depth baseline. The idea is that second-pass sublayers can route over every first-pass delta, including those of sublayers deeper than themselves, which no single pass can provide.

3.3 The human-agent research loop

The baseline above was built with the help of an agentic loop directed by a human. We describe it here as methodology. The loop operates on a small, fixed codebase split in two. `prepare.py` holds the data, the tokenizer, the dataloader, and the `evaluate_bpb` evaluator, which together define the task and the metric, and it is read-only. `train.py` holds the model, the optimizer, and the training loop, and it is the only file the agent may edit. Architecture, optimizer, hyperparameters, batch size, and model size are therefore all in scope; the evaluator and the dependencies are not.

Each iteration runs as follows:

(i) edit `train.py` with one hypothesis and commit

(ii) train for exactly 10 epochs

(iii) read `val_bpb` and the per-epoch validation curve from the run log

(iv) *keep* the change if `val_bpb` improves beyond a noise threshold ($\sim$0.005, picked by hand rather than estimated from repeated runs), otherwise `git reset` to the previous best

(v) document the result

Architectural hypotheses come from the human researcher. The agent applies the specified edits, schedules the training runs and evaluations, and logs the results. The point of the arrangement is scheduling: the human can read papers and implementations while the GPU stays busy, which on a single consumer card is most of the constraint.

Metric. The in-loop objective is validation bits-per-byte (`val_bpb`), a per-byte cross-entropy on held-out text. Measuring per byte instead of per token makes the number comparable across tokenizers, which matters here because the search changed the tokenizer as well as the model. In later iterations we also ran the zero-shot benchmarks (BLiMP, BLiMP-supplement, EWoK, COMPS, entity tracking) to check hypotheses that `val_bpb` might mislead us about. It is an imperfect proxy for them: some changes improve linguistic generalization while leaving the causal cross-entropy flat or slightly worse, and a rule reading `val_bpb` alone would reject them. The masked-LM objective of Section 3.1 is exactly such a case.

What the loop was and was not allowed to decide. Agentic search invites two failure modes, unvalidated code and a metric that drifts along with the search, and we constrained both. The evaluator, the data, and the metric live in the read-only half of the codebase, so no run can change what a number means and every configuration is scored by the same code. Each iteration is a single commit whose diff the human read before the run started, and every run log we quote from is released with the code. As a reproduction check, we re-ran the $P{=}1$ baseline at seed 42 from a later commit and recovered the original `val_bpb` to 4×10^{-5}.

We also separate two kinds of number in what follows. The headline comparison of Section 4.2 is confirmatory in intent: three seeds of each model, all five benchmarks under the official full harness, and an average over all five with no task dropped for being noisy. Everything else, including the fast evaluations and the single-seed screens of Section 5 and Appendix A, is the output of a greedy search and is labelled as such where it appears. We did not pre-register the protocol, and the search that produced the baseline was history-dependent by construction, so this is a separation we impose on the reporting and not a property of how the experiments were run.

Change (kept)	**Type**	**`val_bpb`**
Baseline (depth 8, $d{=}512$)	base	1.5598
Muon LR $0.04 \to 0.06$	optim	1.5468
Muon LR $0.06 \to 0.08$	optim	1.5325
Constant weight decay	reg	1.4996
Weight decay $0.2 \to 0.3$	reg	1.4815
Full-attention window (L)	arch	1.4738
Stochastic weight averaging	avg	1.3898†
Muon LR $0.08 \to 0.10$	optim	1.3824†
Value-embed weight decay 0.1	reg	1.3385†
Muon weight decay $0.3 \to 0.2$	reg	1.3312†
SWA start $0.5 \to 0.6$	avg	1.3266†
Sequence-length warmup	signal	1.3242†
Multi-token prediction ($K{=}2$)	signal	**1.3173**†

Table 1: The tuned baseline, in discovery order. Each kept change is an optimizer, regularization, averaging, or training-signal change on a fixed-capacity model. †From Stochastic weight averaging onward, `val_bpb` is evaluated on the SWA-averaged weights (Section 3.1), not the raw final-epoch weights; rows above are raw.

4 Results

4.1 A strong regularization baseline

Both architectural changes of Section 3.2 are evaluated on top of a heavily tuned baseline. We tuned first and changed the architecture second so that any improvement would be attributable to the loop or the routing, and not to optimization slack the baseline had left on the table. Table 1 lists the chain of *kept* changes the loop produced, from the initial model to that baseline. Most rows cleared the $\sim$0.005 noise threshold of Section 3.3 on their own. Two did not: the SWA start shift ($0.5 \to 0.6$) and sequence-length warmup were kept on human judgement, so the threshold was in practice applied softly. Over the whole chain `val_bpb` falls from 1.560 to 1.317.

The single largest step is stochastic weight averaging (Izmailov et al., 2018), worth -0.084. Rather than score the final, noisy iterate, it scores a uniform average of the weights over the last stretch of training, and it composes cleanly with the learning-rate increase that follows it. Every other large move was regularization. Switching to a constant weight-decay schedule removed an overfitting tail in the back half of training, and adding weight decay to the value-embedding tables changed the shape of the validation curve entirely: it had been U-shaped, bottoming out around epoch 6 and rising after, and it became monotonically descending across nearly the whole budget (Figure 1).

One final change completes the baseline. On top of the tuned configuration we add the GPT-

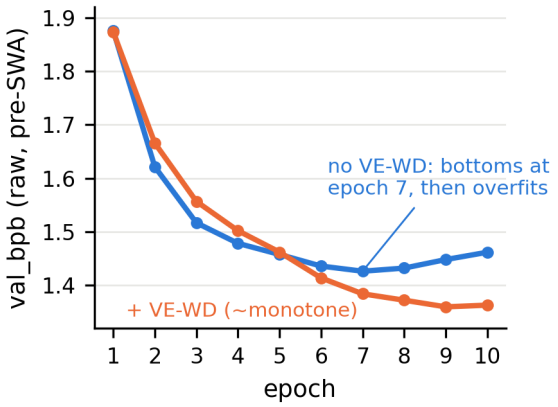


Figure 1: Per-epoch `val_bpb` (raw, pre-SWA) across the full 10-epoch budget for the last kept step before adding weight decay to the value-embedding tables (blue) and the step that adds it (orange). Without value-embedding weight decay the curve bottoms at epoch 7 and then rises as the 17M-parameter value tables overfit; with it the curve descends monotonically through epoch 9, with only a negligible uptick at the final epoch.

BERT masked-LM objective (Charpentier and Samuel, 2024) on 15% of microbatches. This raises `val_bpb` slightly, from 1.3173 to 1.3221, because the metric is measured causally and cannot see the bidirectional benefit, while improving downstream grammar. This MLM-augmented model, at `val_bpb` 1.3221, is the baseline for all architectural experiments below.

Finally we observe that the tuned model mildly *underfits* at 10 epochs: extending the budget to 14 epochs improves `val_bpb` by a further -0.008 before flattening (Appendix A.2).

4.2 Looped depth: a second pass over the block stack at zero parameter cost

Applying the block stack twice per forward pass (P=2) is our main confirmed result. It lowers `val_bpb` at all three seeds we ran. The three looped draws are 1.3148, 1.3178, and 1.3199 at seeds 42, 43, and 44, giving a mean of 1.3175 and a standard deviation of 0.0026. The same three seeds of the P=1 baseline span 1.3202 to 1.3249 (mean 1.3222, s.d. 0.0024; seeds 43 and 44 come from an earlier commit that reproduces seed 42's `val_bpb` to 4×10^{-5}). The two ranges do not overlap. The gap between the means is -0.0047 with a standard error of 0.0020. That gap is narrower than the $\sim$0.005 threshold of Section 3.3, which is worth stating plainly: the threshold is a single-run accept rule and does not bound a difference of means, so the two numbers are not in tension. The looped runs also reach a lower final training loss, so the gain is not an artifact of the averaging window.

Downstream, the direction agrees but the size does not resolve. Table 3 reports the full zero-shot suite for the same three seeds of each model, scored under the official harness. Looped depth raises the five-task average by 0.10 points (50.00 against 49.90) with a standard error of 0.17. Per task, four of five move up and one moves down: BLiMP-supplement $+0.64$, BLiMP $+0.19$, entity tracking $+0.16$, COMPS $+0.13$, EWoK -0.63. None of these is larger than its own seed spread. The baseline seeds span 0.51 points on the average (49.59–50.10) and the looped seeds 0.18 (49.93–50.11), so at three seeds the downstream effect is not resolvable, and we do not claim one. Two of the five tasks are also weak instruments at this scale. Entity tracking sits within a point or two of chance for every configuration we trained, looped or not, and the reading-time correlations that the harness also reports are near zero everywhere and are omitted from the average (Appendix A). Supervised fine-tuning tells the same story: on the BabyLM GLUE tasks the model lands near majority-class baselines (Appendix A). Whatever competence a 50M-parameter model acquires from 10M words shows up in the grammatical probes, and BLiMP and its supplement are the only two tasks where we would expect an architectural change of this size to be visible at all.

What the second pass costs. The looped model has exactly the same 50M parameters as the baseline and spends more of the fixed budget refining its representation in depth. Table 2 prices that. At $1.71\times$ the baseline's training time, the second pass is expensive, and two cheaper or comparable ways of spending the same compute do at least as well on `val_bpb`. Training the P=1 baseline for 14 epochs instead of 10 reaches 1.3140 in $1.24\times$ the baseline's time, below every looped draw. That run exceeds the track's epoch limit, so we report it as a probe and not as a submittable configuration, but it bounds what the loop is worth against the simpler option of training longer. A 100M-parameter depth-12 model reaches 1.2979 at $1.80\times$ the time, close to the looped model's cost and clearly ahead of it on both `val_bpb` and the downstream average (50.19 over two seeds against 50.00). We therefore state the claim as parameter efficiency and not compute efficiency: at a fixed parameter count the loop

Configuration (seed 42)	Par.	Time	val_bpb	BLiMP
Baseline P=1, 10 epochs	50M	1.00×	1.3215	71.27
Baseline P=1, 14 epochs‡	50M	1.24×	1.3140	71.19
Looped P=2, 10 epochs	50M	1.71×	1.3148	71.09
Looped P=3, 10 epochs	50M	2.45×	1.3142	–
Deep (depth 12), 10 epochs	100M	1.80×	1.2979	71.86

Table 2: Compute-matched comparison at seed 42, on one RTX 3060. "Time" is wall-clock training time relative to the baseline's 4031 s. Looping is not the cheapest way to buy val_bpb: a longer schedule gets below it for less, and a model with twice the parameters gets well below it for a comparable cost. ‡Exceeds the track's 10-epoch limit; reported as a probe only (Appendix A.2).

Model	BLiMP	Supp.	EWoK	COMPS	Ent.	Avg.
Baseline, s42	71.27	54.69	51.42	51.67	18.91	49.59
Baseline, s43	71.72	55.37	52.59	51.16	19.16	50.00
Baseline, s44	70.64	56.64	51.50	51.98	19.73	50.10
3-seed mean	71.21	55.57	51.84	51.60	19.27	49.90
Looped, s42	71.09	56.56	51.31	51.71	19.07	49.95
Looped, s43	71.42	56.69	50.89	51.27	19.36	49.93
Looped, s44	71.68	55.37	51.43	52.23	19.86	50.11
3-seed mean	71.40	56.21	51.21	51.74	19.43	**50.00**

Table 3: Weight-tied looped depth (P=2) against the tuned baseline (P=1), official zero-shot harness, three seeds each. "Avg." is the mean of the five columns to its left. Baseline seeds 43 and 44 are from an earlier commit (Section 4.2).

is a reliable gain, and at a fixed compute budget it is not the best available move. It also does not follow that the 14-epoch run is strictly better, since its full BLiMP is flat at 71.19 against 71.27 for the same configuration at 10 epochs (Appendix A.2): val_bpb and the benchmarks come apart there too.

Why two passes and not three. A third pass buys almost nothing. P=3 lands at 1.3142 at seed 42, 0.0006 below P=2's 1.3148 and well inside the seed-to-seed spread, while costing another full pass of block computation per token (Table 2). We can only speculate about why the returns fall off so sharply, and we did not run the experiments that would settle it. The reading we find most plausible is that the passes are not doing the same kind of work. The first pass has to turn embeddings into contextual representations, and the second refines representations that already exist, which is a cheaper problem and one the same weights can plausibly serve; by the third, the input to the stack is close to what the second pass already produced, and re-applying the same map to it has little left to change. The training budget also pushes in this direction. Each extra pass costs a fixed fraction of the total compute whether or not it helps, and with the data fixed at 10M words the model is only mildly underfit to begin with (Section 4.1), so there is a limited amount of loss for extra computation to remove. We fix P=2 for the submitted model and did not explore P>3.

5 Negative Results

Depth routing is dead weight, and looping does not revive it. The negative axis we explored most thoroughly is delta attention routing. We pursued it because it looked like it *should* work. We started with routing on its own, in an ordinary single forward pass, and tried four formulations: a replacement formulation over cumulative hidden states, a delta formulation, a gated-additive variant, and the paper-faithful per-sublayer version with $2L$+1 sources, each at depth 8 and depth 12. All were neutral to negative, on val_bpb and downstream alike.

The loop should have given routing something to work with. Once the stack runs twice, a second-pass sublayer can route over the first pass's per-sublayer deltas, which are new relative to anything in its own residual input, and no single-pass model can offer them. It did not help. val_bpb regressed at both seeds we ran, to 1.3224 against the loop's 1.3148 at seed 42 and 1.3242 against 1.3178 at seed 43. The downstream picture was flat rather than negative: on the fast evaluation, the harness's reduced-sample suite that we used as an in-loop screen (Appendix A), the five-task average came out at 50.52 and 51.48 against the loop's 51.34 and 50.56. We therefore rejected the extension on val_bpb and on cost, not on the benchmarks.

The learned router weights show why. Figure 2 plots them, and they collapse. Almost all of the routing mass concentrates on the token embedding emb, which is the residual-and-embedding skip the model already carries, plus one early source from within the same pass. The cross-pass deltas that the loop makes newly available, the whole justification for the extension, receive almost no weight. Given the choice, the router declines to use them and re-derives the skip it already had.

We offer one reading of this, and we offer it as a hypothesis: we have not established it. The baseline may have already saturated the dimension the routing operates in. It carries ResFormer-style per-token value residuals (Zhou et al., 2025), in the

form of the value-embedding mechanism of Section 3.1, and a learned residual-and-embedding mix at every block. Much of what a routed mix over sublayer deltas could contribute may already be reachable through those two paths, and the router's preference for exactly the skip that the scalars provide is what that account predicts. We have one piece of supporting evidence and it is indirect: removing the value-embedding tables from the looped model costs 0.0096 `val_bpb` and drops full BLiMP from 71.09 to 70.55 at seed 42, so that path is doing real work in the model. The experiment that would actually test the hypothesis is delta routing on a baseline stripped of value embeddings and of the embedding skip, and we did not run it. A baseline without those paths might well respond to routing differently, and we do not claim otherwise.

6 Conclusion

At the 10M-word budget, applying the same block stack twice per forward pass lowers `val_bpb` at every seed we ran, at zero additional parameters, on top of a baseline we had already tuned hard. That direction is the opposite of what recent at-scale work reports, where looping buys reasoning and gives up perplexity. Our reading is that the data budget accounts for the difference: a model that is mildly underfit on a fixed corpus can still spend computation on quality, while a compute-optimal one cannot. The gain is real but small, and it is not free. It costs $1.71\times$ the training time, and at comparable cost a model with twice the parameters does better (Section 4.2). Looping buys parameter efficiency, not compute efficiency, and we would not recommend it to anyone whose constraint is compute rather than parameters.

Delta attention residual routing, the other architectural direction we pursued, was neutral or worse everywhere we tried it (Section 5).

Two directions follow from the positive result. Exact weight tying already helps, so a small per-pass parameter budget (Bae et al., 2025) might let the two passes specialize into distinct functions at close to zero parameter cost. And because the second pass pays broadly while the third barely pays at all, per-token adaptive depth, in the spirit of the dynamic halting of Universal Transformers (Dehghani et al., 2019), would spend the extra computation only where it earns its keep.

On the methodology: the human-directed agentic loop was a useful way to keep a single GPU busy, and its mechanical accept rule made the failures legible after the fact. Its limitation showed up on the bidirectional objective, where the accept metric and the target evaluations disagreed and a human had to override the rule. Searches of this kind should either choose an acceptance metric that tracks the target evaluations or score against those evaluations directly. We release the code and the full training and evaluation logs.[1]

Limitations

Our study has several limitations.

Few seeds, one GPU. The headline looped-depth result replicates across three seeds, but most other configurations were trained once or twice. We also chose the $\sim$0.005 `val_bpb` accept threshold by hand: it is a guess at the run-to-run noise and we never measured that noise directly, so improvements near the threshold should be read as suggestive. Every run is on one consumer GPU (RTX 3060, 12 GB), which bounded the batch and model sizes we could explore at all and caps the seed budget at three.

The currency here is compute. The looped model costs $1.71\times$ the baseline's training time at the same parameter count. Two cheaper uses of that compute do at least as well on `val_bpb`: a longer schedule for the same model and a larger model at comparable cost (Table 2). We restrict the claim to parameter efficiency for that reason. Looping is not the best way to spend a fixed compute budget at this scale, and our own measurements are what say so.

Single-seed negative results. Most rejected hypotheses were evaluated at one or two seeds. We reject them because they failed to beat the baseline beyond the noise threshold at the point they were tried, not because we have excluded a smaller effect that a larger seed budget could resolve. One result deserves a specific flag: the fast-eval BLiMP of 72.46 from cross-pass routing (Table 5) is the highest BLiMP number in the project, and it is a single draw that falls to 71.20 at the next seed. We do not read it as evidence that the extension helps. We also cannot fully exclude a BLiMP-specific effect that a `val_bpb`-driven accept rule would never have kept.

[1] Code and experiment logs: https://github.com/quesswho/looped-transformer.

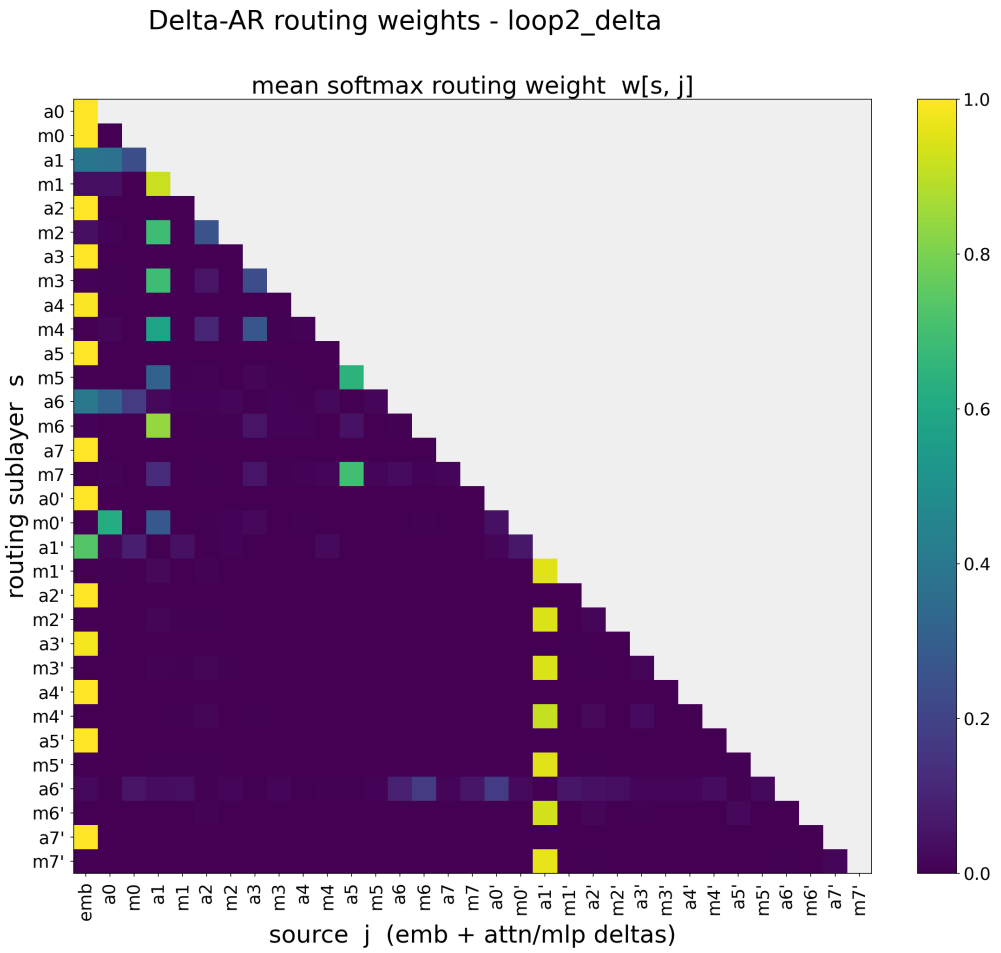


Figure 2: Mean softmax routing weights $w[s, j]$ for the cross-pass delta-routing variant (loop2_delta). Rows are the $2L$ routing sublayers of each of the two passes; columns are the available sources (`emb` plus each earlier attention/MLP delta). Despite having the first pass's per-sublayer deltas available, the router collapses onto the embedding skip `emb` and one early attention delta from within the same pass (`a1` in the first pass, `a1'` in the second), leaving the first pass's deltas near zero in the primed rows. The routing re-derives the model's existing skip rather than exploiting the new information the loop exposes.

Greedy, history-dependent search. The loop is greedy. It keeps any change that beats the current best and reverts the rest, so the final configuration is path-dependent. A technique rejected early, residual dropout or added depth for instance, might have helped on top of a later configuration, and the order in which we explored the changes conditions the outcome. Table 1 traces one path through the space; it is not an ablation of the final model, and no row there should be read as the isolated contribution of that change.

Proxy-metric optimization. The loop optimizes `val_bpb` and not the downstream benchmarks. The masked-LM objective is a case where the two came apart visibly enough for a human to catch it. Other divergences may exist that neither the loop nor our out-of-loop evaluation saw, particularly among changes the loop rejected on `val_bpb` without ever scoring them downstream.

Scope of conclusions. Our claims concern BabyLM Strict-Small: 10M words, ∼50M parameters, this corpus and this tokenizer. Whether any of it transfers to other data scales, model sizes, or corpora is untested, and establishing that would require replication we did not have the compute to run.

References

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. 2025. Relaxed recursive transformers: Effective parameter sharing with layer-wise LoRA. In *International Conference on Learning Representations*.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2019. Deep equilibrium models. In *Advances in Neural Information Processing Systems*, pages 688–699.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *Proceedings of the 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. DeepSeek-V3 technical report. *Preprint*, arXiv:2412.19437.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems*.

Angeliki Giannou, Shashank Rajput, Jy yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. 2023. Looped transformers as programmable computers. In *Proceedings of the 40th International Conference on Machine Learning*, pages 11398–11442.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Andrej Karpathy. 2025a. autoresearch. `https://github.com/karpathy/autoresearch`.

Andrej Karpathy. 2025b. nanochat. `https://github.com/karpathy/nanochat`.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Cheng Luo, Zefan Cai, and Junjie Hu. 2026. Delta attention residuals. *Preprint*, arXiv:2605.18855.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*.

Matteo Pagliardini, Amirkeivan Mohtashami, Francois Fleuret, and Martin Jaggi. 2024. DenseFormer: Enhancing information flow in transformers via depth weighted averaging. In *Advances in Neural Information Processing Systems*.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations*.

Noam Shazeer. 2020. GLU variants improve transformer. *Preprint*, arXiv:2002.05202.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Will we run out of data? limits of LLM scaling based on human-generated data. In *Proceedings of the 41st International Conference on Machine Learning*.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. Technical Report 25/2013, Aalto University, Helsinki, Finland.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Zhanchao Zhou, Tianyi Wu, Zhiyun Jiang, Fares Obeid, and Zhenzhong Lan. 2025. Value residual learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. 2025a. Hyper-connections. In *International Conference on Learning Representations*.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, and 14 others. 2025b. Scaling latent reasoning via looped language models. *Preprint*, arXiv:2510.25741.

A Full Evaluation Results

The main text reports a small set of numbers from the loop's overall record; the entire record is given here. Both evaluation modes come from the official BabyLM evaluation pipeline: *FULL* scores all 67 BLiMP paradigms and the complete supplement/EWoK/COMPS/entity sets (Table 4), while *fast* is the pipeline's fast suite, whose smaller per-task samples are intended for intermediate checkpoints; we used it as the in-loop screen, and it is correspondingly noisier (Table 5). It does not cover COMPS, so five-task fast averages quoted in the text take COMPS from the matching FULL evaluation. Entity tracking varies by only a few points across the configurations in this appendix and is essentially at chance for every variant we tried; it is nonetheless included in the main-text average (Section 4.2). The reading-time correlations are near zero for every run and are not tabulated here.

A.1 Fast-eval exploratory and rejected runs (higher variance)

Table 5 lists every in-loop fast evaluation; these numbers are comparable only within this table. The looped model ($P{=}2$) beats the seed-matched baseline at seed 42 (51.34 vs. 48.91) and ties at seed 43 (50.56 vs. 50.70); the delta-routing run's project-high 72.46 BLiMP at seed 42 falls to 71.20 at seed 43, which is why the main text rests nothing on it.

A.2 Other probes

Several measurements do not fit the zero-shot table and are summarized here.

GLUE fine-tuning. Fine-tuning the 50M MTP model for 10 epochs gives an accuracy-average of 57.7 across the BabyLM GLUE tasks (BoolQ 64.0, MNLI 35.7 $\approx$ chance, MRPC 68.1, MultiRC 57.6, QQP 62.8, RTE 54.0, WSC 61.5), with an official mixed-metric average of 42.3 (F1 collapses to 0 on MultiRC/QQP). These sit near majority/chance baselines, as expected for a 50M-parameter model on 10M words: competence here is zero-shot, not supervised.

Extended-budget probe (14 epochs). To test the underfitting claim of Section 4.1 directly, we re-ran the $P{=}1$ baseline with the budget extended from 10 to 14 epochs, which exceeds the track's 10-epoch limit and is therefore a probe only, stretching every

Configuration	val_bpb	BLiMP	Supp.	EWoK	COMPS	Ent.
50M base (d=512, depth 8)						
MTP (K=2) baseline	1.3173	70.45	55.35	52.05	51.91	17.96
SwiGLU causal baseline	1.3161	69.32	54.64	51.20	51.42	18.12
+ MLM (15%) *[arch baseline]*	1.3221	70.15	56.15	52.16	51.56	18.57
Baseline P=1, seed 42	1.3215	71.27	54.69	51.42	51.67	18.91
Baseline P=1, seed 43	1.3202	71.72	55.37	52.59	51.16	19.16
Baseline P=1, seed 44	1.3249	70.64	56.64	51.50	51.98	19.73
Scale (wider / deeper), +MLM 15%						
70M wide (d=640, depth 8)	1.3088	71.16	54.91	52.62	51.93	18.59
85M deep (d=640, depth 10) s42	1.3171	71.76	56.54	50.94	51.68	19.20
depth 10, seed 43	–	71.39	54.59	50.25	51.43	18.11
100M deep (d=640, depth 12) s42	1.2979	71.86	57.04	53.01	51.91	17.47
depth 12, seed 43	–	71.58	56.12	51.68	51.52	19.73
Looped depth and routing (50M)						
Looped P=2, seed 42	1.3148	71.09	56.56	51.31	51.71	19.07
Looped P=2, seed 43	1.3178	71.42	56.69	50.89	51.27	19.36
Looped P=2, seed 44	1.3199	71.68	55.37	51.43	52.23	19.86
Delta-AR (gated-additive)	1.3237	70.23	55.80	50.93	51.34	–
Delta-AR (per-sublayer, gated)	–	70.28	56.19	52.69	51.68	–
AttnRes DenseFormer routing	1.540	61.94	53.17	50.65	50.67	–

Table 4: All FULL evaluations, in discovery order. Accuracies are percentages; val_bpb lower is better. "–" = not measured.

progress-based schedule (learning rate, weight decay, sequence-length warmup, MTP) over the new budget. val_bpb improves from 1.3215 (baseline re-run) to 1.3140 (-0.0076), with the per-epoch validation curve flattening around epochs 11–12: the model is still gaining from optimization at 10 epochs, i.e. mildly underfit. The 14-epoch value is also below each of the three looped draws at 10 epochs (Section 4.2), yet its full BLiMP is flat (71.19 against 71.27 for the same configuration at 10 epochs): another case of val_bpb moving without the benchmarks.

B Hyperparameters

Table 6 lists the full configuration of the submitted looped model. The P=1 baseline is identical except for the number of block passes. All schedules are driven by training progress, defined as completed epochs divided by the 10-epoch budget.

Configuration (fast eval)	val_bpb	BLiMP	Supp.	EWoK	Ent.
Baseline trajectory					
Best original-data baseline	1.3266	69.03	57.20	51.09	19.37
+ sequence-length warmup	1.3242	69.17	55.60	51.73	21.15
+ MTP (K=2, "all")	1.3173	70.93	58.00	52.27	20.21
MTP K=3	–	70.87	56.80	52.18	19.56
GPT-BERT masked-LM sweep (causal backend)					
SwiGLU causal baseline	1.3161	69.59	56.40	52.09	18.64
+ MLM 5%	1.3156	70.20	56.40	53.18	17.66
+ MLM 15%	1.3209	70.54	57.20	55.36	20.44
15% rerun, seed 42	–	71.21	54.00	51.64	20.17
+ MLM 25%	1.3293	70.69	50.80	53.64	18.91
Tokenizer granularity (Morfessor vocab)					
vocab 4096 (finer)	–	70.81	54.40	50.18	19.72
vocab 8192 (default)	1.3173	70.93	58.00	52.27	20.21
vocab 16384 (coarser)	1.3147	69.40	56.40	50.00	19.49
Scale / context / data variants					
70M wide, MLM 0	1.306	71.08	56.80	50.64	19.03
50M, context cap 512	1.387	70.51	56.00	52.18	19.22
Corpus clean + refill	–	69.25	54.00	51.55	19.96
Corpus clean + dedup + refill	–	69.31	56.80	52.45	20.17
Corpus finalized (<10M)	–	69.61	57.20	52.82	18.77
Register tilt (written 56%)	–	69.31	55.20	51.82	19.53
P=1 *baseline, looped depth, and delta-routing family*					
Baseline P=1, seed 42	1.3215	71.38	53.60	49.00	18.88
Baseline P=1, seed 43	1.3202	71.74	57.20	53.00	20.41
Baseline P=1, seed 44	1.3249	71.17	56.00	51.91	20.39
Looped P=2, seed 42	1.3148	71.78	59.20	52.73	21.29
Looped P=2, seed 43	1.3178	71.53	58.00	52.18	19.80
Looped + delta route, seed 42	1.3224	72.46	56.40	52.64	19.65
Looped + delta route, seed 43	1.3242	71.20	61.60	52.45	20.04
Looped + delta route v2 (carry)	1.3437	69.14	–	–	–
Delta-AR per-sublayer (paper)	1.3255	70.96	59.60	51.18	19.98
Looped + curriculum	1.4047	70.59	–	–	–
Looped, VE ablation (33M)	1.324	70.83	–	–	–
100M depth-12, fast eval					
100M deep, seed 42	1.2979	72.27	58.80	52.27	18.28
100M deep, seed 43	1.2985	72.07	56.00	50.55	20.46

Table 5: All fast evaluations. "–" = not measured.

Setting	Value
Architecture	
Layers L / block passes P	8 / 2
Model dim d / heads	512 / 4
Head dim	128
MLP / norm / positions	SwiGLU / RMSNorm / RoPE
Attention window	full causal, every layer
Value-embedding tables	every 2nd layer + last
Vocabulary (Morfessor)	8192, byte fallback
Context length	2048
Muon (hidden matrices)	
Learning rate	0.10
Momentum (Nesterov)	$0.85 \rightarrow 0.95$ over 300 steps
Newton–Schulz / β_2	5 / 0.95
Weight decay	0.2, constant
AdamW (embeddings, scalars)	
Token-embedding LR	0.6
Unembedding LR	0.004
Per-layer scalar LR	0.5
(β_1, β_2)	$(0.8, 0.95)$
Value-embedding WD	0.1, decoupled
Budget and schedules	
Epochs	10
Tokens per optimizer step	2^{19} ($\approx$524K)
Device batch / accum.	8×2048 / 32
LR schedule	constant, then linear to 0
over final	50% of training
Sequence-length warmup	$256 \rightarrow 2048$ over first 50%
SWA start (uniform avg.)	60% of training
Multi-token prediction	K=2, aux. weight 0.3
Masked-LM microbatches	15%, 30% tokens masked
Seeds	42, 43, 44

Table 6: Full configuration of the submitted looped model (P=2); the P=1 baseline differs only in the number of block passes.

OLMo2 with Hybrid MLM-CLM Training for the BabyLM 2026 Multilingual Challenge

Zichao Wei
Saarland University
ziwe00001@stud.uni-saarland.de

Abstract

We present our submission to the BabyLM 2026 Multilingual Shared Task. Our system uses OLMo2, a modern decoder-only architecture with grouped-query attention, RoPE, and SwiGLU, trained from scratch on the official BabyLM trilingual corpus (English, Dutch, Chinese). We employ a hybrid training objective following (Charpentier and Samuel, 2024) that combines masked language modeling (87.5%) with causal language modeling (12.5%), enabling the decoder-only model to perform well on both bidirectional and autoregressive evaluation tasks. With a multilingual BPE tokenizer (32K vocabulary) and approximately 79M parameters, our model achieves a multilingual average of 47.12 on the official leaderboard. We release our model and training code.[1]

1 Introduction

The BabyLM Challenge (Warstadt et al., 2023) encourages the development of language models trained on limited, developmentally plausible data. The 2026 edition introduces a multilingual track requiring models to handle English, Dutch, and Chinese within a 100M-word budget. The previous edition (Charpentier et al., 2025) demonstrated that hybrid masked-causal training objectives are particularly effective in this data-constrained setting.

We explore whether modern architectural advances—specifically the OLMo2 architecture (Team OLMo et al., 2025)—can be effectively combined with the hybrid training paradigm (Charpentier and Samuel, 2024) in a multilingual setting. Our key contributions are:

- **Architecture choice:** We adopt OLMo2, a decoder-only transformer with grouped-query attention (GQA), rotary position embeddings (RoPE), SwiGLU activations, and RMSNorm. To our knowledge, this is the first application of OLMo2 to the BabyLM challenge.

- **Hybrid training objective:** Following Charpentier and Samuel (2024), we combine masked language modeling (MLM) with causal language modeling (CLM), using MLM for 87.5% of batches and CLM for 12.5%. This enables the decoder-only model to leverage bidirectional context for BLiMP-style evaluation tasks while retaining autoregressive capabilities.

- **Multilingual BPE tokenizer:** We train a byte-pair encoding tokenizer with Unicode script pre-tokenization, producing a 32K vocabulary that efficiently handles all three languages.

2 Model

2.1 Architecture

Our model uses the OLMo2 configuration with 8 layers, 8 attention heads (4 key-value heads via GQA), a hidden dimension of 512, and an intermediate dimension of 1,376. Word embeddings are tied between input and output. The total parameter count is approximately 79M. The maximum sequence length is 512 tokens.

2.2 Data and Tokenizer

We use the official BabyLM 2026 multilingual datasets with a 40/30/30 English/Dutch/Chinese split by document count, totaling approximately 100M words. We train a BPE tokenizer with NFKC normalization and lowercase folding, using Unicode script-based pre-tokenization to handle the distinct writing systems. The vocabulary size is 32,000 tokens.

[1]Model: https://huggingface.co/exusiaiw/emnlp-babylm-2026-v2

Proceedings of the Second BabyLM Workshop, pages 539–542
October 29, 2026 ©2026 Association for Computational Linguistics

Category	EN	NL	ZH
Zeroshot Avg	41.33	55.24	49.95
Finetune Avg	48.59	45.80	42.80
Language Avg	45.48	50.21	45.66
Multilingual Avg		**47.12**	

Table 1: Results on the BabyLM 2026 multilingual evaluation suite. Multilingual Average is the official leaderboard score.

2.3 Training

We train for 10 epochs using AdamW with a peak learning rate of 1×10^{-3}, weight decay of 0.1, warmup ratio of 0.06, and gradient accumulation over 2 steps (effective batch size of 64 sequences $\times$ 512 tokens = 32,768 tokens per step). Training uses FP16 mixed precision and takes approximately 7 hours on a single NVIDIA A100 GPU.

2.4 Hybrid Objective

Following Charpentier and Samuel (2024), for 87.5% of training batches, we apply masked language modeling: 15% of tokens are selected for prediction, of which 80% are replaced with a [MASK] token, 10% with a random token, and 10% left unchanged. The attention mask is set to be fully bidirectional. For the remaining 12.5% of batches, we apply standard causal language modeling with a triangular attention mask. This hybrid approach allows the decoder-only architecture to develop bidirectional representations while maintaining autoregressive generation capability.

3 Results

We evaluate using the official BabyLM 2026 multilingual evaluation pipeline (Biderman et al., 2026), comprising zero-shot tasks (BLiMP, MultiBLiMP, ZhoBLiMP, HellaSwag, Winogrande, XStoryCloze, XCOMPS, GlobalPIQA), fine-tuning tasks (ARC, Belebele, BMLama, MNLI, SIB-200, TruthfulQA, XNLI, INCLUDE, POS tagging), and MECO reading time prediction.

Our model achieves a multilingual average of 47.12 on the official leaderboard. Table 2 shows per-task results across languages.

3.1 Learning Curve

Figure 1 shows the training and evaluation loss over the course of training. The train loss decreases steadily from 10.4 to 3.7, while the eval loss plateaus around 3.9 after epoch 5, suggesting that additional training epochs yield diminishing returns at this model scale.

Task	EN	NL	ZH
Zero-shot			
BLiMP / ZhoBLiMP	71.01	77.91	78.12
MultiBLiMP	85.32	92.32	–
HellaSwag	26.80	26.56	27.78
Winogrande	51.28	50.87	50.70
XStoryCloze	49.92	49.23	49.69
XCOMPS	–	53.71	56.25
GlobalPIQA (par.)	21.36	25.24	23.30
GlobalPIQA (non-par.)	48.00	47.00	51.00
Fine-tuning			
ARC	27.50	26.67	20.83
Belebele	25.57	28.41	22.73
BMLama	24.67	24.42	16.06
MNLI	55.69	58.56	57.77
SIB-200	81.00	80.50	79.00
TruthfulQA	25.89	25.89	21.43
XNLI	55.75	–	52.10
INCLUDE	–	28.57	24.11
POS	92.62	93.42	91.20

Table 2: Per-task accuracy (%) across languages.

3.2 Checkpoint Evaluation

Figure 2 shows zero-shot task performance at intermediate checkpoints during training. Linguistic tasks (BLiMP, MultiBLiMP, ZhoBLiMP) show rapid improvement in the first 10M words and continue to improve throughout training, reaching 78–92% accuracy. Commonsense reasoning tasks (HellaSwag) plateau early and remain near chance level (27%), consistent with the limited world knowledge available in child-directed speech. XCOMPS and BLiMP-NL show steady improvement, with Dutch benefiting from vocabulary overlap with English.

3.3 Analysis

The model shows consistent performance across languages, with Dutch slightly outperforming English and Chinese on zero-shot tasks, likely due to favorable tokenizer fertility (Dutch shares many subwords with English in our BPE vocabulary). Chinese zero-shot performance benefits from the dedicated ZhoBLiMP evaluation (78.12%), suggesting that the hybrid training objective effectively captures Chinese syntactic patterns.

Fine-tuning performance varies by task type. Tasks like SIB-200 (topic classification) and POS tagging achieve strong results (81–93%), while knowledge-intensive tasks (BMLama, TruthfulQA) are near random, as expected for a model trained only on child-directed speech data. The Glob-

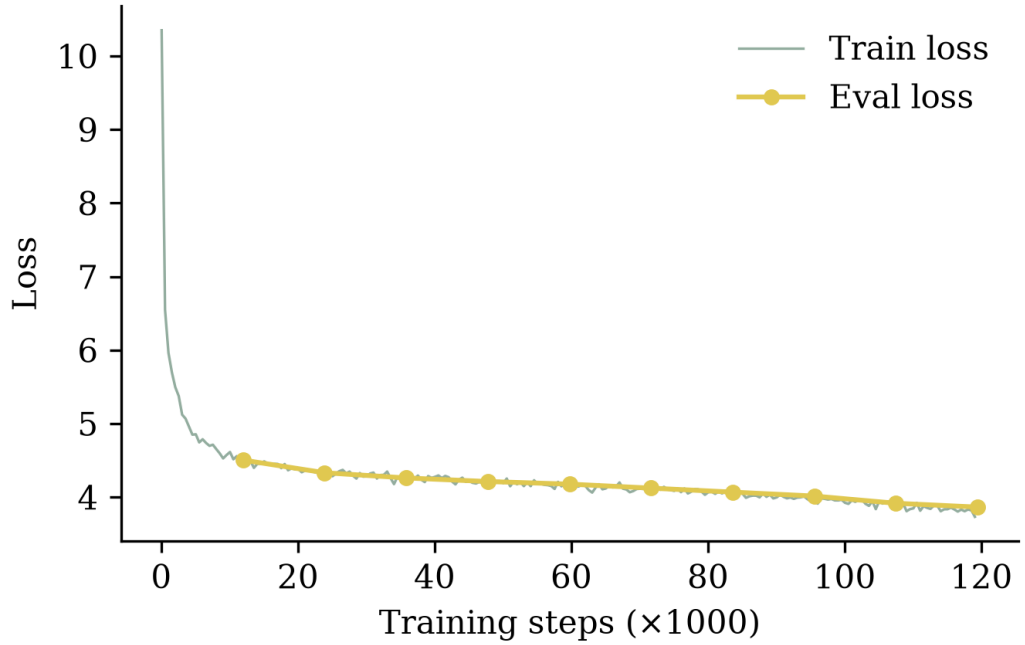


Figure 1: Training and evaluation loss over 10 epochs (119K steps).

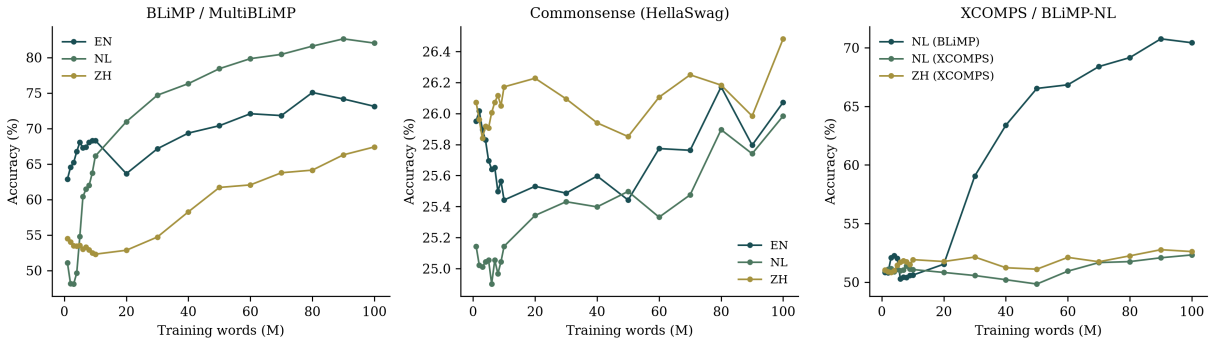


Figure 2: Zero-shot task accuracy at intermediate checkpoints (1M–100M training words). Left: grammatical acceptability. Center: commonsense reasoning. Right: semantic compatibility.

alPIQA parallel task is challenging for all models, while the non-parallel variant shows reasonable performance (47–51%).

The hybrid MLM-CLM objective is motivated by the evaluation setup: BLiMP-style tasks use log-probability comparisons that benefit from bidirectional context, while text generation tasks require autoregressive modeling. Our 87.5/12.5 split prioritizes the more numerous bidirectional evaluation tasks.

4 Conclusion

We presented a simple but effective approach for the BabyLM 2026 multilingual challenge based on OLMo2 with hybrid training. Our results suggest that modern architectural choices (GQA, RoPE, SwiGLU) combined with a hybrid training objective can achieve competitive performance in data-constrained multilingual settings. Future work could explore curriculum learning across languages and more sophisticated data mixing strategies.

5 Limitations

Our model is trained on a fixed 40/30/30 language split without tuning the mixing ratio. The hybrid objective ratio (87.5% MLM / 12.5% CLM) was also not systematically optimized. Additionally, our model has 79M parameters, which may be suboptimal for the 100M-word budget—smaller or larger models might achieve better data efficiency.

References

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri

Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, and 11 others. 2026. Lessons from the trenches on reproducible evaluation of language models. *Preprint*, arXiv:2405.14782.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 24 others. 2025. 2 OLMo 2 furious. *Preprint*, arXiv:2501.00656.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Loop Before You Leap: What Recurrent Depth Can and Cannot Buy in Data-Limited Pretraining

Daichi Hotta[1] **Yohei Kobashi**[2] **Takeshi Kojima**[2]
Yusuke Iwasawa[2] **Yutaka Matsuo**[2]
[1]Aoyama Gakuin University
[2]The University of Tokyo
daichi8120@gmail.com

Abstract

Multimodal models have often scored below text-only baselines in the BabyLM Challenge (Klerings et al., 2024; Takmaz et al., 2025). We test whether additional recurrent computation helps close this gap. Recurrence increases computation by reusing layers without adding parameters or training data. Using a single implementation, we compare standard and looped Transformers under a 100M-word budget, with three seeds per main condition. Each image is represented by one feature vector from a frozen DINOv2 encoder, used as a prefix token. Caption words count toward the budget, so multimodal training also changes the language data. In a pre-registered text-only comparison, a 12-layer stack repeated four times exceeds a standard 48-layer Transformer by 5.4 BLiMP percentage points at matched inference compute. However, multimodal training lowers BLiMP accuracy by about 3.8 points in both architectures, with no clear mitigation from recurrence. An exploratory loop-side no-image control, trained with the multimodal language data and batching schedule, scores 5.7 points below the text-only baseline. Adding the visual prefix raises accuracy by 1.85 points on average, but this contrast is uncertain with three seeds. These results suggest that changes in language data and its presentation are a major source of the drop. Diagnostic tests show that the models use visual information; its contribution to grammatical generalization remains uncertain under this single-vector interface.

1 Introduction

Children acquire language from far less linguistic input than modern language models, and they do so while perceiving the world. Motivated by this contrast, the BabyLM Challenge limits the amount of training text and asks how data, learning objectives, and model architectures affect language learning under that constraint (Charpentier et al., 2025; Choshen et al., 2026).

The 2024 and 2025 BabyLM challenges included a multimodal track to explore whether visual input could support language learning under a 100M-word budget (Hu et al., 2024; Charpentier et al., 2025). Yet visual input has not reliably improved language performance, and multimodal training can reduce scores on grammatical benchmarks (Klerings et al., 2024; Takmaz et al., 2025). This raises the question of what prevents models from using visual information to improve language learning.

One possibility is that integrating visual and linguistic information requires additional sequential computation. A looped Transformer increases this computation by repeatedly applying the same layers, without adding parameters or training data (Dehghani et al., 2019; Geiping et al., 2025). We therefore test whether recurrence reduces the language-performance gap between text-only and multimodal training.

To assess how recurrence uses computation, we compare looped Transformers with deeper standard models at matched inference compute. This comparison asks whether, under a fixed word budget, additional computation is better spent on layers with separate weights or on repeated use of the same layers.

In our multimodal setting, captions count toward the same word budget and replace part of the original text corpus. This changes the mix of language sources and genres, so the performance gap cannot be attributed to visual input alone. We therefore use no-image controls in both architectures to examine the contribution of the changed language data, and a batch-matched control to separate the visual-prefix contribution from the remaining data and batching changes in the looped model.

All conditions share a single implementation. The standard configuration applies the layers once, with text-input injection and visual refresh disabled. We compare both architectures with and without

Proceedings of the Second BabyLM Workshop, pages 543–562
October 29, 2026 ©2026 Association for Computational Linguistics

visual input, using the same visual input scheme for the main comparison between multimodal models. Figure 1 summarizes the three comparisons.

Each image is represented by a single feature vector from a frozen DINOv2 encoder pretrained on images alone (Oquab et al., 2024). The language model is trained from scratch on text and image captions, using the visual features to condition caption prediction.

RQ1. At matched inference compute, the looped model exceeds the 48-layer standard model by 5.4 BLiMP percentage points in text-only training and 3.7 points in multimodal training. Its gains over the parameter-matched 12-layer model are much smaller, at about one point, and remain uncertain with three seeds (Section 5.1).

RQ2. Recurrence does not reliably narrow the language-performance gap between text-only and multimodal training. The main comparison estimates a difference of -0.01 BLiMP points between the performance drops, with a 95% confidence interval of $[-1.44, 1.41]$. Tests with alternative connectors and visual injection schedules also provide no clear evidence of mitigation by recurrence (Section 5.2; uncertainty estimates in Limitations).

RQ3. Exploratory controls suggest that changes in language data and its presentation are a major source of the BLiMP drop. A batch-matched no-image control for the looped model scores 5.71 points below its original text-only baseline. The visual prefix recovers 1.85 points on average, with a 95% confidence interval of $[-1.67, 5.37]$. This exploratory decomposition combines language-data and batching changes in its nonvisual term. The matched decomposition is limited to the looped model, so we cannot compare these components between architectures (Section 5.3).

2 Related Work

Sample-efficient pretraining. High-performing BabyLM submissions have introduced changes to training data, learning objectives, and model architecture (Hu et al., 2024; Charpentier et al., 2025). The relationship between training compute and performance has been less consistent: it was strong in the 2024 analysis but varied across tracks in 2025. This motivates controlled comparisons of how computation is allocated under a fixed word budget.

Recurrent and weight-tied Transformers. Universal Transformers repeatedly apply a shared transformation across depth (Dehghani et al., 2019). Geiping et al. (2025) vary the number of repetitions of a shared block to increase computation at inference. Saunshi et al. (2025) find that looped language models can approach or exceed deeper non-looped models on reasoning tasks despite worse perplexity at matched compute. Schwethelm et al. (2026) find that replacing independent layers with shared recurrences increases validation loss at matched training compute.

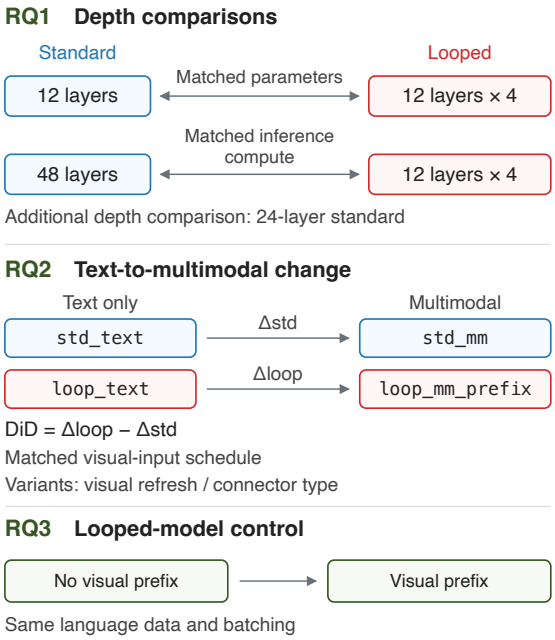


Figure 1: RQ1 compares standard and looped models at matched parameter count or inference compute. RQ2 compares their changes in accuracy from text-only to multimodal training. RQ3 compares the looped multimodal model with a no-image control matched in language data and batching.

Recurrence meets multimodality. RecursiveVLM extends a pretrained multimodal model with recurrent processing (Xu et al., 2026). HIVE combines a pretrained recurrent language model with visual features drawn from multiple encoder layers and injected across iterations (Zhang et al., 2026). We study recurrent computation when the language model is trained from scratch under a 100M-word budget, focusing on grammatical generalization.

Grounding at developmental scale. Amariucai and Warstadt (2023) independently vary text and image quantities to investigate whether visual input can compensate for limited language data. Takmaz et al. (2025) partially restore language performance by merging multimodal models with text-

only models. Ganescu et al. (2025) introduce token-wise gating to control the contribution of visual features during language processing. Our study tests whether recurrent computation narrows the language-performance gap between text-only and multimodal training. Controls using identical language data then distinguish the effect of visual input from that of changing the text corpus.

3 Method

3.1 One implementation, six conditions

We use a single implementation of a decoder-only Transformer (Vaswani et al., 2017) for both standard and looped models. In the six main conditions, the backbone consists of 12 layers. The standard model applies this stack once, whereas the looped model applies it four times, sharing weights across iterations.

The backbone follows a LLaMA-style design (Touvron et al., 2023), with pre-norm RMSNorm, rotary position embeddings (Su et al., 2024), and SwiGLU feed-forward layers. We omit bias terms and share weights between the input embedding and output projection (Press and Wolf, 2017).

Let $E(x)$ denote the token embeddings of text x, v the visual feature vector (multimodal cells only), and $P(\cdot)$ the connector that maps v into model space. The forward computation applies the shared core stack K times:

$$\begin{aligned} h^{(0)} &= [\,P(v);\,E(x)\,], \\ h^{(k)} &= \mathrm{Core}\Big(h^{(k-1)} \\ &\qquad + \mathbb{1}[k \le N_{\mathrm{inj}}]\,[\,P(v);\,\mathbf{0}\,]\Big) \\ &\quad + a\,[\,\mathbf{0};\,E(x)\,], \end{aligned} \tag{1}$$

for $k = 1, \ldots, K$. The looped models add the original text embeddings after every application of the shared core. We denote this input-injection setting by a, with $a = 1$ for looped models and $a = 0$ for standard models.

Text-only models omit the visual prefix and visual refresh term, so $h^{(0)} = E(x)$. With $K = 1$, $a = 0$, and $N_{\mathrm{inj}} = 0$, the computation reduces to a standard Transformer. The final state $h^{(K)}$ passes through RMSNorm and the output projection to predict the next token.

Parameter and compute accounting. Repeating the shared core increases computation without adding parameters (Table 4, Appendix A). At

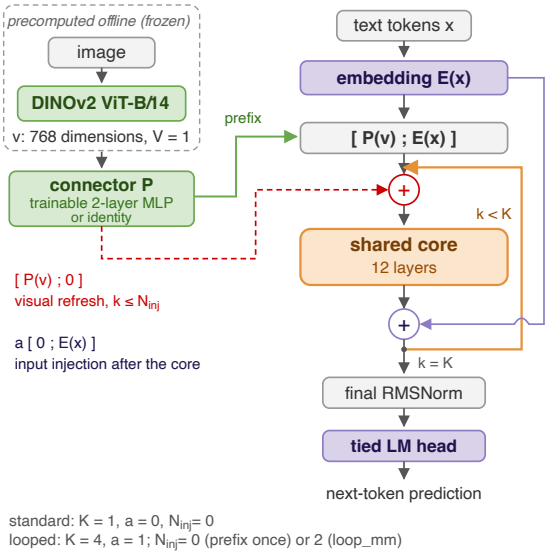


Figure 2: Model architecture with a single visual prefix token from frozen DINOv2 features. Looped models add text embeddings after every application of the shared core. The dashed path adds the visual vector before the first N_{inj} applications in `loop_mm`.

$K = 4$, the 12-layer core performs 48 layer applications. We compare this model with a 12-layer standard Transformer at matched parameter count and a 48-layer standard Transformer at matched inference compute.

3.2 Visual grounding by offline, frozen features

Each image is represented by a single 768-dimensional feature vector from a frozen DINOv2 ViT-B/14 encoder (Oquab et al., 2024). We use the precomputed vectors supplied with the BabyLM multimodal data. The encoder was pretrained on images only; raw images are not processed during language-model training. Figure 2 shows the architecture.

A connector P maps the visual vector into the language model's embedding space. For image–caption examples, we prepend the resulting vector to the caption embeddings as a single visual token. The model is trained to predict text and caption tokens; text-only examples have no visual prefix.

By default, P is a two-layer MLP with a GELU activation between two 768×768 linear transformations (Liu et al., 2023, 2024). We also test an identity connector that passes the visual vector through unchanged, since its dimension already matches the language model's embedding size.

With $N_{\mathrm{inj}} = 0$, the visual vector is supplied only as the initial prefix. With $N_{\mathrm{inj}} > 0$, the same

projected vector is also added at the visual-prefix position before each of the first N_{inj} applications of the shared core (Eq. 1).

4 Experimental Setup

4.1 Conditions and seeds

The four core conditions combine standard or looped models with text-only or multimodal training (Figure 1). The looped models use $K = 4$. In multimodal training, `std_mm` supplies the visual vector only as an initial prefix, whereas `loop_mm` also adds it at the prefix position before the first two iterations. We fixed $N_{\text{inj}} = 2$ a priori; it was not tuned.

Two additional conditions test the visual injection schedule and connector. In `loop_mm_prefix`, the visual vector is supplied only as an initial prefix, matching `std_mm`. In `std_mm_id`,[1] an identity connector replaces the MLP, passing the visual vector through unchanged. Each condition is one Hydra configuration, reproduced with a single argument.

We train each of these six conditions with three seeds (42, 43, and 44), stopping when cumulative exposure reaches 1B words. The peak learning rate is selected on `std_text` and reused across these six conditions.

4.2 Data and word budget

The text-only conditions use 100M words from the BabyLM 2026 Strict corpus (Choshen et al., 2026). The multimodal conditions use 50M words from the same corpus and 50M caption words, keeping the total language budget at 100M words.[2] Each model is trained until cumulative exposure reaches 1B words, counting repeated presentations of the training data. For multimodal models, this total includes words from both text and caption batches.

The captions and precomputed visual features come from the 2024 BabyLM multimodal release (OSF project `ad7qg`). Our caption set contains 3,037,190 examples from Localized Narratives (Pont-Tuset et al., 2020) and Conceptual Captions 3M (Sharma et al., 2018).

For the multimodal conditions, we take the first 50M words of the text corpus in its stored order. This changes the proportions of language sources in the retained text. Together with replacing the remaining text with captions, this produces what we call the *composition shift*.

CHILDES (MacWhinney, 2000) accounts for 56.8% of the retained text, compared with 28.4% of the full text corpus. The remaining words come from BNC spoken dialogue (15.2%) and Gutenberg (27.9%). OpenSubtitles, Simple Wikipedia, and Switchboard are absent from the retained text. These proportions are measured by word count against the official domain files (Choshen et al., 2026).

We train a byte-level BPE tokenizer (Sennrich et al., 2016; Radford et al., 2019) with a vocabulary of 16,000 tokens on the text corpus. The same tokenizer is used across all conditions without further training. We release the assembled corpus as a Hugging Face dataset with a datasheet (Gebru et al., 2021) recording provenance, domain composition, and per-file manifests (Appendix A).

4.3 Optimization: a words-seen–driven schedule

We use AdamW (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay of 0.1. Training uses bfloat16 precision, and the gradient norm is clipped at 1.0. For the six main conditions, we use micro-batches of 64 sequences, with text sequences of up to 512 tokens. Gradients are accumulated over eight micro-batches before each optimizer update.

The peak learning rate is 1.5×10^{-3}, selected from five candidates evaluated in 100M-word training runs on `std_text`.[3] We schedule the learning rate by cumulative words seen. It increases linearly over the first 15M words, then follows a cosine decay (Loshchilov and Hutter, 2017) to 10% of its peak at 1B words. For the decay phase,

$$\eta(w) = \eta_{\max}\left[\rho + \tfrac{1-\rho}{2}\big(1 + \cos \pi\, p(w)\big)\right], \quad (2)$$

where w is cumulative words seen, $p(w) = \frac{w - w_{\text{warm}}}{w_{\max} - w_{\text{warm}}}$, $w_{\text{warm}} = 15\text{M}$, $w_{\max} = 1\text{B}$, and $\rho = 0.1$.

Captions are kept as separate sequences, whereas text-only training packs text into fixed-length blocks. Multimodal training therefore requires more optimizer updates to reach the same word exposure.[4] Scheduling by words seen aligns the learning rate across conditions at the same cumulative exposure.

[1] Named `std_mm_naive` in the released configurations.

[2] Exact whitespace-delimited counts are 100,000,000 words for text-only training, and 49,999,988 text words plus 49,999,996 caption words for multimodal training.

[3] The candidates span 3×10^{-4} to 2×10^{-3}.

[4] At seed 42, `std_mm` takes 13,895 updates and `std_text` takes 6,827.

4.4 Checkpoints and evaluation

We save checkpoints at 28 cumulative-word milestones between 1M and 1B words. Formats and loading details are in Appendix A. We evaluate the models using a fixed revision of the official BabyLM pipeline with causal language-model scoring.

We report BLiMP (Warstadt et al., 2020) accuracy on the full filtered evaluation set, with a breakdown by linguistic category (`linguistics_term`). For the six main conditions, we report the mean and standard deviation across three seeds.[5] Our aggregation reproduces the official average (75.23 for `std_text`, seed 42) as a correctness check.

4.5 Analysis: the interaction as a difference-in-differences

For RQ2, we compare the change in BLiMP accuracy from text-only to multimodal training between standard and looped models. Our primary comparison uses `loop_mm_prefix`, so that both multimodal models receive visual input only as an initial prefix. We compute this contrast as a difference-in-differences, with condition names denoting BLiMP accuracy:

$$\Delta_{\text{DiD}} = (\texttt{loop\_mm\_prefix} - \texttt{loop\_text}) - (\texttt{std\_mm} - \texttt{std\_text}). \quad (3)$$

Positive values indicate a smaller text-to-multimodal drop for the looped model; negative values indicate a larger drop.

We also compute the contrast with `loop_mm` in place of `loop_mm_prefix` to assess sensitivity to visual refresh. A second set of contrasts uses `std_mm_id` as the standard multimodal baseline.

Switching from text-only to multimodal training changes both the visual input and the language data. A corpus effect that is equal in both architectures cancels in the difference-in-differences. Differences in how the architectures respond to the changed corpus remain part of the contrast. The architectural contrast includes both recurrence and text-input injection, whose individual contributions are not separated in this design.

	text	multimodal
standard (K=1)	75.21±0.68	71.37±0.29
looped (K=4, re-inj.)	76.16±0.45	72.38±0.83
looped, prefix-once	—	72.31±0.63
identity connector (K=1)	—	72.48±0.69
identity connector (K=4)	—	72.14±0.15
std2x (24 layers)	73.84±0.25	70.96±0.88
std4x (48 layers)	70.80±0.60	68.64±0.29

Table 1: BLiMP accuracy (%) on the full evaluation set, reported as mean ± standard deviation across three seeds. All multimodal conditions use an initial visual prefix; "re-inj." also adds the visual vector before the first two core applications.

	vs. `std_mm`	vs. `std_mm_id`
penalty (standard)	−3.84	−2.74
penalty (looped, re-inj.)	−3.79	−3.79
penalty (looped, prefix-once)	−3.86	−3.86
Δ_{DiD} (re-injected)	+0.06	−1.05
Δ_{DiD} (**matched**)	−0.01	−1.12

Table 2: Changes in BLiMP accuracy from text-only to multimodal training and their interactions, in percentage points (three-seed means). Positive Δ_{DiD} indicates a smaller drop for the looped model. The primary contrast is the "matched" row under `std_mm`, where both multimodal models use only an initial visual prefix (Eq. 3).

5 Results

5.1 RQ1: iteration beats added parameter depth

A parameter-matched, modality-independent depth gain. At matched parameter count, the looped text-only model scores 76.16 on BLiMP, compared with 75.21 for the 12-layer standard Transformer. The 0.95-point difference is not statistically significant with three seeds per condition (two-sided Welch's test, $p \approx 0.12$).[6] In multimodal training, `loop_mm_prefix` exceeds `std_mm` by 0.94 BLiMP points on average, similar to the text-only difference. The difference is positive in all three seed pairings: 0.72, 0.61, and 1.48 points.

The visual-refresh variant, `loop_mm`, exceeds `std_mm` by 1.01 BLiMP points, although this comparison also changes the visual injection schedule. Against the standard model with an identity connector, `std_mm_id`, `loop_mm` scores 0.10 points lower.

[5] The public leaderboard hosts single-seed submissions of the four core cells (seed 42, except seed 43 for `loop_mm`); the extension cells are analysis-only and not submitted. This paper reports three-seed means; the board's own aggregation can differ in the last digit (76.62 vs. our 76.61 for `loop_text`).

[6] Welch's $t \approx 2.0$; Appendix E reports per-seed values.

Iso-FLOPs baselines: looping beats added depth. We compare the looped model with standard Transformers containing 24 and 48 independent layers. The 48-layer model matches the looped model's inference compute. We registered the comparison and its statistical decision rule on the Open Science Framework before observing the final results from these deeper standard models.[7]

The 24- and 48-layer models have approximately 180M and 350M parameters, respectively. We train each depth in both text-only and multimodal conditions with three seeds. A separate peak learning rate is selected for each depth through 100M-word training probes.[8]

At matched inference compute, the looped model outperforms the 48-layer standard model by 5.36 BLiMP points in text-only training and 3.67 points in multimodal training. Across the standard text-only models, BLiMP falls monotonically as depth is added ($75.21 \rightarrow 73.84 \rightarrow 70.80$; Table 1), while the looped model reaches 76.16 by iterating 97M parameters.

Deeper standard models have lower training loss but also lower BLiMP accuracy. For seed 42, average training loss is 1.88 with 12 layers, 1.63 with 24 layers, and 1.26 with 48 layers.[9]

Our preregistered test asks whether multimodal training increases the looped model's advantage over the 48-layer standard model. We subtract the text-only advantage from the multimodal advantage:

$$\begin{aligned}\mathrm{DiD}_{\mathrm{iso}} = &(\texttt{loop\_mm\_prefix} - \texttt{std4x\_mm}) \\ &- (\texttt{loop\_text} - \texttt{std4x\_text}).\end{aligned}$$

We pair runs with the same seed. The mean contrast is -1.69 BLiMP points, with a 90% confidence interval of $[-3.22, -0.16]$.[10] This meets the preregistered *Reversed* criterion: the looped model's advantage is smaller under multimodal training. The 95% confidence interval is $[-3.95, 0.57]$ and includes zero.[11]

[7] Study-plan registration, accessible through the view-only link: https://osf.io/438b9/?view_only=44cd0d22cb19498a9282ed806de06c02.

[8] For the 48-layer models, we use micro-batches of 32 sequences and accumulate gradients over 16 micro-batches, preserving the nominal batch size of 512 sequences per update.

[9] Averaged over the final eight logged values.

[10] The interval uses a t distribution with df$= 2$, following the registered procedure; the registered decision margin is $\delta = 1.8$, based on the paired 95% confidence interval of the main-grid interaction. The per-seed contrasts are -1.91, -2.47, and -0.69.

[11] On the log-odds scale, the 90% confidence interval is also negative (-0.17 to -0.02).

Deeper standard models have lower text-only BLiMP scores and smaller drops under multimodal training (Table 1). Their smaller drops may partly reflect their lower text-only performance. This is a post-hoc explanation that our design cannot distinguish from a cost specific to the looped architecture.

The corresponding contrast against the 24-layer standard model is -0.97 BLiMP points. Appendix J reports additional experiments varying K, with one seed each for the $K = 2$ and $K = 8$ runs.

5.2 RQ2: recurrence does not clearly reduce the multimodal drop

Similar drops across architectures. Compared with text-only training, BLiMP accuracy drops by 3.84 points for `std_mm` and 3.86 points for `loop_mm_prefix`. The drop for `loop_mm`, which refreshes the visual prefix, is 3.79 points (Tables 1 and 2).

The primary difference-in-differences estimate is -0.01 BLiMP points, with a 95% confidence interval of $[-1.44, 1.41]$.[12] We find no clear evidence that the looped model reduces the drop associated with multimodal training.

Using `loop_mm` instead of `loop_mm_prefix` gives a difference-in-differences estimate of $+0.06$ points. When `std_mm_id` is used as the standard multimodal baseline, the estimates are -1.05 points with `loop_mm` and -1.12 points with `loop_mm_prefix`.

Across the 13 linguistic categories, the mean primary contrasts range from -1.66 to 1.48 BLiMP points. The sign changes across seeds in 10 of the 13 categories (Figure 3, Appendix D).

Connector comparison. In the standard model, the identity connector scores 72.48 on BLiMP, compared with 71.37 for the learned MLP. The 1.11-point difference is not statistically significant with three seeds per condition (two-sided Welch's test, $p \approx 0.09$).

We repeat this comparison in the looped model with three seeds, replacing only the MLP in `loop_mm_prefix` with an identity connector. This condition, `loop_mm_naive`, scores 72.14 on BLiMP, compared with 72.31 for `loop_mm_prefix`. These comparisons provide no

[12] This interval pools the error across twelve runs. The seed-paired interval is $[-1.83, 1.80]$; see Limitations for both estimates.

clear evidence that the learned MLP improves BLiMP accuracy in either architecture.

Visual-refresh comparison. With $K = 4$ and the training data held fixed, visual refresh changes BLiMP accuracy by $+0.07$ points on average. The per-seed differences are -0.78, $+1.55$, and -0.56 points, providing no clear evidence of a benefit from visual refresh.

5.3 RQ3: what the penalty is made of

A matched control for the looped model. We train a no-image control for the looped model, keeping the language data, batches, word schedule, and initialization of `loop_mm_prefix` and omitting only the visual prefix. Its mean BLiMP accuracy is 70.46 across three seeds.[13] Adding the visual prefix raises BLiMP by 1.85 points on average, with positive differences in all three seeds (2.23, 0.28, and 3.04) and a 95% confidence interval of $[-1.67, 5.37]$.

The original looped text-only model scores 5.71 points above the no-image control, with a 95% confidence interval of $[2.73, 8.68]$. This difference includes changes in language data, batch construction, and the number of optimizer updates. The means suggest that the visual prefix partly offsets the cost of this changed training recipe in the looped model, but its contribution remains uncertain with three seeds. Removing the prefix also removes its sequence position, so the contrast does not isolate image content from prefix mechanics.

Other controls and their scope. We also train no-image controls on the multimodal conditions' language data using the text-only procedure of packing tokens into fixed-length blocks. The standard model scores 70.56, compared with 75.21 on its original text corpus; the looped model scores 71.31, compared with 76.16 (Table 1; Appendix L). These results support a contribution from the changed language data even without visual input. Packing uses the same procedure, but the number of updates can change under the fixed word budget. Comparisons with multimodal training also change the batching procedure and therefore do not isolate visual input.

Together, the controls suggest that changes in language data and its presentation are a major source of the drop. The matched loop-side control provides the direct decomposition; the packed controls provide supporting evidence in both architectures.

For the standard model, a training-time control pairs each caption with another image's features. It scores 71.55 on BLiMP across three seeds, compared with 71.37 for `std_mm`.[14] These results provide no clear evidence of a BLiMP benefit from preserving the correct image–caption pairing.

The batch-matched no-image decomposition is limited to the looped model. We therefore cannot compare the visual-prefix and data-and-batching contributions between architectures.

The penalty emerges over training. The multimodal models briefly lead early in training, but the text-only models overtake them as training continues. These fast-eval curves cover 28 checkpoints and use one submitted seed per condition, so we treat them as descriptive (Appendix F).

The grounded models do use vision: a shuffled-features diagnostic. To test whether the trained models use visual information, we score the same 50,000 captions with correct, shuffled, and zeroed image features. This changes the visual input at evaluation time while keeping the model and captions fixed (Appendix G).

Shuffling the image features increases caption prediction loss (negative log-likelihood, NLL) by 1.45 nats per token for `std_mm`. The increases are 1.52 for `loop_mm` and 1.49 for `loop_mm_prefix`.[15] For every evaluated model and seed, shuffling increases the loss for at least 98.6% of captions. This shows that the models use visual information when predicting captions.

We also evaluate visual question answering (VQA) with correct, shuffled, and zeroed image features. We follow the multimodal track's evaluation protocol but extract the image features ourselves, so these results are diagnostic rather than leaderboard scores (Appendix H). All four multimodal conditions score above the 12.5% chance level. Correct image features yield higher accuracy than shuffled features in all twelve model–seed combinations, with gains of 2.4 to 5.6 points.[16] No looped condition exceeds `std_mm`'s mean VQA ac-

[13] The per-seed scores are 70.12, 71.38, and 69.88. All three conditions in this decomposition were evaluated at the same pipeline revision (Appendix L).

[14] The per-seed scores are 71.34, 71.78, and 71.52.

[15] Cohen's $d \approx 1.9$ for `std_mm`; the standard deviation of the NLL increase across seeds is at most 0.01 for the looped variants.

[16] McNemar $p < 10^{-25}$ in each comparison.

Task	pen. (std)	pen. (loop)	DiD	DiD_{m}
BLiMP	−3.84	−3.79	+0.06	−0.01
BLiMP Suppl.	−3.66	−4.95	−1.29	−0.08
EWoK	+0.04	−0.93	−0.96	−0.64
COMPS	−0.62	−0.01	+0.61	−0.10
Entity tracking†	−0.42	+1.85	+2.28	+1.88

Table 3: Changes in zero-shot accuracy from text-only to multimodal training, in percentage points (three-seed means). The pen. (loop) and DiD columns use `loop_mm`, whereas DiD_{m} uses `loop_mm_prefix` with the same visual-input schedule as `std_mm`. Both interactions use `std_mm` as the multimodal standard baseline, and positive values indicate a more favorable change for the looped model. †All conditions score below the 20% chance level on Entity tracking.

curacy of 47.4%. With standard deviations across seeds reaching 4.8 points, these results provide no clear evidence of a VQA benefit from looping.

Beyond BLiMP: different effects across tasks. Multimodal training lowers accuracy on BLiMP Supplement (Warstadt et al., 2023) as it does on BLiMP (Table 3). The changes are smaller on EWoK, which tests world knowledge (Ivanova et al., 2025), and COMPS, which tests conceptual knowledge (Misra et al., 2023). These small changes do not establish that these abilities are unaffected by multimodal training.

On BLiMP Supplement and EWoK, the mean interactions are negative with both matched visual input and visual refresh (Table 3). With visual refresh, the EWoK interaction is negative in all three seeds, whereas BLiMP Supplement shows greater variation across seeds.[17]

Entity tracking (Kim and Schuster, 2023) yields positive mean interactions, but all conditions score below the 20% chance level. We therefore do not interpret these interactions as evidence that looping mitigates the multimodal performance drop.

Reading, age of acquisition (AoA), and GLUE were evaluated only for the submitted checkpoints and are not analyzed here. Appendix B provides comparisons with the official leaderboard baselines for context.

6 Discussion

Recurrence and the multimodal gap. Compared with the 12-layer standard model, the looped model shows a similar BLiMP gain in text-only and multimodal training. It does not clearly narrow the gap between the two training conditions.

Multimodal training replaces half of the original text with captions, changes the source distribution of the retained text, and changes how batches are constructed (Section 4.2). These changes may reduce exposure to some of the linguistic patterns tested by BLiMP. The loop-side matched control quantifies their combined contribution: the no-image condition scores 5.71 points below the original text-only model, and the visual prefix recovers 1.85 points on average. The latter contrast remains uncertain. Together with the packed no-image controls, these results suggest that changes in language data and its presentation are a major source of the drop. The missing standard-side control prevents an attribution of the overall interaction to differences in these components (Section 5.3).

The models use visual information to predict captions and answer questions about images. This does not by itself establish an improvement in grammatical generalization. In the looped model, the improvement over its matched no-image control is positive in every seed but remains uncertain.

The scope of the depth gain. The gain over the 12-layer standard model is modest. In our main multimodal comparisons, `loop_mm` scores 72.38 and remains below the identity-connector standard model, `std_mm_id`, at 72.48.

The comparison with the 48-layer standard model suggests that how computation is allocated matters under this data budget (Section 5.1). Weight sharing may help limit overfitting by allowing additional computation without adding parameters. On 20,000 held-out captions (Appendix F), the looped text-only model has an NLL of 5.83 nats per token. The 24- and 48-layer text-only standard models have NLLs of 6.35 and 7.24, respectively.

The connector. Removing the learned MLP does not restore text-only BLiMP performance in either architecture. This comparison changes the visual connector and does not isolate the contribution of visual input. The experiments with richer visual inputs in Appendix I show no clear BLiMP benefit. Each condition uses one seed, so these results are preliminary.

From compute to mechanism. Gradually shifting from caption-heavy to text-heavy training yields 72.70 on BLiMP, compared with 72.31 for a constant mixture. Reversing this schedule yields

[17] Across seeds, the visual-refresh contrasts have standard deviations of 2.09 points for BLiMP Supplement and 0.67 for EWoK.

66.62. Each direction is evaluated with three seeds (Appendix K). This asymmetry suggests that the language distribution encountered late in training matters. Ending with caption-heavy training lowers BLiMP accuracy, while ending with text-heavy training offers no clear improvement over a constant mixture.

Our visual input is a single global image vector paired with a caption. It does not explicitly represent how an utterance relates to objects and events over time. Pinker (1984) proposes that semantic cues can help learners identify grammatical categories. Gleitman (1990) argues that sentence structure can help learners infer verb meanings.

We hypothesize that grounding helps grammatical generalization only when the model can connect parts of an utterance to the objects and events they describe. Testing this hypothesis requires explicit mechanisms for learning these connections.

7 Conclusion

We studied recurrent Transformers under a 100M-word training-data budget. At matched inference compute, the looped model outperformed the 48-layer standard Transformer on BLiMP in both text-only and multimodal training. With a single global image vector as visual input, we found no clear evidence that looping reduces the text-to-multimodal BLiMP drop. The primary difference-in-differences estimate was -0.01 points, with a 95% confidence interval of $[-1.44, 1.41]$.

Exploratory controls suggest that changes in language data and its presentation are a major source of the BLiMP drop. For the looped model, a matched no-image control separates the change in language data and batching from the visual-prefix contribution. The no-image control scores 5.71 points below the original text-only baseline, while adding the visual prefix recovers 1.85 points on average, with substantial uncertainty across three seeds. A matched standard-model control is needed to compare these components between architectures. Future work should test whether explicitly linking linguistic structure to objects and events in visual input improves grammatical generalization.

Limitations

Statistical power. Each main condition was trained with three random seeds. This limits our ability to detect small differences between architectures. In particular, a primary interaction estimate near zero does not establish that looping has no effect on the multimodal BLiMP drop. The pooled 95% confidence interval for the primary interaction is $[-1.44, 1.41]$ BLiMP points.[18] This interval allows both a reduction and an increase in the multimodal drop (Lakens, 2017). Its upper end corresponds to a reduction of about 37% in the 3.84-point drop observed in the standard model. We report several secondary comparisons without adjusting for multiple testing, including connector variants and breakdowns by linguistic phenomenon and task. These analyses are exploratory, and differences close to the significance threshold should not be treated as confirmed effects.

Language data and visual input. Multimodal training replaces half of the text-only corpus with captions and changes the source distribution of the retained text (Section 4.2). The resulting BLiMP drop therefore combines changes in language data with the addition of visual input. Our primary interaction measures how the two architectures respond differently to this combined change. The effect of the language-data change cancels in this contrast only if it changes BLiMP accuracy by the same amount in both architectures. The batch-matched no-image control for the looped model (Section 5.3) separates the visual prefix from the rest of the multimodal recipe within that model. The remaining term combines language-data and batching changes, because text-only training packs text into blocks and uses fewer optimizer updates. This matched comparison is limited to the looped model, so we cannot determine whether either component differs between architectures. Appendix L reports the loop-side per-seed values, together with packed controls that use a different batching procedure. The looped model also adds the original text embeddings after each application of the shared layers. Our comparison therefore does not separate the contribution of repeated layer applications from that of text-input injection.

Interpreting BLiMP scores. We assess grammatical generalization using BLiMP and its Supplement. Lower scores may reflect weaker grammatical generalization, a mismatch between the language encountered during training and the benchmark sentences, or both. Our experiments do not

[18] Pooling the error across the twelve runs gives the interval reported above. Computing the interaction separately for each of the three seeds and using a paired analysis gives a wider 95% confidence interval of $[-1.83, 1.80]$.

fully distinguish these explanations. Future evaluations should test grammatical contrasts in a wider range of language, including conversation and captions. This would help assess whether the observed drop extends beyond the kinds of sentences used in BLiMP.

Visual evaluation. Our visual question answering evaluation is diagnostic (Appendix H). It covers one task with three seeds per condition and substantial variation across seeds. We also use features extracted ourselves rather than the officially distributed features, so these scores are not directly comparable to leaderboard results.

Visual representation. Our main experiments represent each image with a single global feature vector. The results may differ with visual inputs that preserve more spatial detail, such as patch-level features (Ganescu et al., 2025; Charpentier et al., 2025). Our exploratory experiments with richer visual inputs use only one seed per condition and do not resolve this question (Appendix I).

The identity connector requires matched widths. The identity connector is possible because the visual features and the language model's hidden states both have 768 dimensions. When these dimensions differ, a mapping between them is required, so our findings do not imply that a learned projection can generally be omitted.

Model and training settings. We use a single decoder-only model family, tokenizer, and training objective. The main looped models use four iterations, and experiments with two or eight iterations use only one seed per condition (Appendix J). The findings may therefore change with other model designs, loop counts, or data budgets. The main looped models use the learning rate selected for the 12-layer standard model. We selected learning rates separately for the deeper standard models, but kept weight decay, dropout, and warmup fixed across depths. Further tuning for each architecture could change the comparison. Our learning-rate selection uses training loss from short probe runs. A setting that minimizes this loss need not give the best BLiMP accuracy at the end of training. At 100M words, both the 48-layer standard model and the looped model score higher on BLiMP at 10^{-3} than at their main-run learning rates.[19] All models were trained without dropout. Stronger regularization might improve the deeper standard models, but we did not test this possibility.

Training-time feature controls. We attempted a training-time control with the visual features set to zero while retaining the prefix token. All three seeds produced NaN losses early in training, leaving no usable comparison. We therefore used shuffled visual features as an alternative control (Section 5.3). Shuffling preserves the visual feature distribution but breaks the correspondence between images and captions. It therefore tests the contribution of correct image–caption pairing, rather than the effect of removing visual input altogether.

Proxy multimodal data. We use text from the 2026 BabyLM Strict corpus, but image–caption data and visual features from the official 2024 multimodal release. A different caption corpus could change both the multimodal BLiMP drop and the interaction between architecture and training modality.

Ethics Statement

We use the 2026 BabyLM Strict text corpus and the officially distributed image–caption data from the 2024 multimodal track. The captions originate from Conceptual Captions and Localized Narratives. The 2026 Strict text corpus was filtered to reduce toxic content (Choshen et al., 2026). Nevertheless, our training data may still contain harmful content or social biases, which the models may reproduce. Our models are intended for research on language learning from limited data. They are not intended for deployment in user-facing applications.

For the exploratory experiments, we retrieved images associated with the original caption datasets and extracted richer visual features (Appendix I). Some images were no longer available, so these experiments used a reduced subset. Our released training corpus contains text, captions, and provenance records, but no raw images. We direct users to the official BabyLM release for the precomputed visual features used in the main experiments.

[19] For seed 42, the 48-layer model scores 66.02 at 10^{-3} and 64.03 at its main-run rate of 6×10^{-4}. The looped model scores 63.65 at 10^{-3} and 61.20 at its shared main-run rate of 1.5×10^{-3}. At this probe scale, the gains are similar in size for both models. These probes do not establish the effect of further learning-rate tuning at 1B words.

This study uses existing datasets and does not involve experiments with human participants. Personal information may still be present in the source corpora.

Acknowledgments

We thank the anonymous reviewers for their helpful comments.

References

Theodor Amariucai and Alex Warstadt. 2023. Acquiring linguistic knowledge from multimodal input. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning (CoNLL)*.

Lucas Georges Gabriel Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations (ICLR)*.

Bianca-Mihaela Ganescu, Suchir Salhan, Andrew Caines, and Paula Buttery. 2025. Looking to learn: Token-wise dynamic gating for low-resource vision-language modelling. In *Proceedings of the First BabyLM Workshop*. Association for Computational Linguistics.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pages 41340–41391. Curran Associates, Inc.

Lila Gleitman. 1990. The structural sources of verb meanings. *Language Acquisition*, 1(1):3–55.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6325–6334.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 1–21. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Alina Klerings, Christian Bartelt, and Aaron Mueller. 2024. Developmentally plausible multimodal language models are highly modular. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 118–139, Miami, FL, USA. Association for Computational Linguistics.

Daniël Lakens. 2017. Equivalence tests: A practical primer for t tests, correlations, and meta-analyses. *Social Psychological and Personality Science*, 8(4):355–362.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO:

Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, 3rd edition. Lawrence Erlbaum Associates.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, and 7 others. 2024. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*.

Steven Pinker. 1984. *Language Learnability and Language Development*. Harvard University Press, Cambridge, MA.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. 2020. Connecting vision and language with localized narratives. In *European Conference on Computer Vision (ECCV)*, pages 647–664. Springer.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI technical report.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations (ICLR)*.

Kristian Schwethelm, Daniel Rueckert, and Georgios Kaissis. 2026. How much is one recurrence worth? iso-depth scaling laws for looped language models. *arXiv preprint arXiv:2604.21106*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568.

Ece Takmaz, Lisa Bylinina, and Jakub Dotlačil. 2025. Model merging to maintain language-only performance in developmentally plausible multimodal models. In *Proceedings of the First BabyLM Workshop*. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 5998–6008.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics (TACL)*, 8:377–392.

Ruihan Xu, Yuting Gao, Lan Wang, Jianing Li, Weihao Chen, Qingpei Guo, Ming Yang, and Shiliang Zhang. 2026. Looping back to move forward: Recursive transformers for efficient and flexible large multimodal models. *arXiv preprint arXiv:2602.09080*.

Yiming Zhang, Qiangyu Yan, Borui Jiang, and Kai Han. 2026. Multimodal latent reasoning via hierarchical visual cues injection. *arXiv preprint arXiv:2602.05359*.

A Reproducibility

Cell	K	Params	Core FLOPs
`std_text`	1	97,241,856	1×
`std_mm`	1	98,423,040	1×
`loop_text`	4	97,241,856	4×
`loop_mm`	4	98,423,040	4×

Table 4: Parameter counts and relative core-stack FLOPs per token (d_{model}=768, 12 layers, 12 heads, vocab 16,000). Multimodal cells differ from text cells only by the MLP connector (+1,181,184 parameters); `std_mm_id` removes it. The frozen vision encoder runs offline and is excluded.

We save checkpoints at 28 milestones defined by cumulative training words: every 1M words from 1M to 9M, every 10M words from 10M to 90M, and every 100M words from 100M to 1B. Each checkpoint uses a Hugging Face–compatible format and can be loaded with `AutoModelForCausalLM` using `trust_remote_code=True`. For each of the four submitted conditions, we publish the submitted seed's final model and all 28 milestone checkpoints in a single Hugging Face repository. The two additional main conditions, `std_mm_id` and `loop_mm_prefix`, are evaluated from local checkpoints using the same pipeline (Section 4.4).

We use the official evaluation pipeline with the `causal` backend. The original BLiMP records of the six main conditions (Section 4.1) were produced earlier at revision `68cdd16`; the deeper standard models were evaluated at `3d57ddc`. For Appendix L, we use re-evaluated `loop_text` and `loop_mm_prefix` scores at `3d57ddc`, matching the revision used for the loop-side batch-matched no-image control. Tables 1 and 6 retain the original main-condition scores. For text-only inputs, unit tests check numerical agreement with a single-pass standard Transformer when $K = 1$ and text-input injection is disabled. They compare output scores and input-embedding gradients. Separate tests check the word-based learning-rate schedule in Eq. 2.

We train a byte-level BPE tokenizer with a vocabulary of 16,000 tokens on the 100M-word Strict text corpus. The tokenizer is then kept fixed across all conditions. Using the same tokenizer keeps the segmentation rules fixed across conditions. It does not guarantee that tokenization affects both architectures in the same way.

The assembled multimodal training corpus is available on Hugging Face, together with a datasheet (Gebru et al., 2021) and provenance records.[20] The precomputed DINOv2 features used in the main experiments remain at the official BabyLM distribution site.[21]

B Leaderboard context

Our submitted text-only looped model scores 76.62 on BLiMP under causal scoring on the public leaderboard. This exceeds the listed GPT-2 (Radford et al., 2019) Strict baseline (74.73) and Interaction baseline (72.89). The three official GPT-BERT baseline variants score between 74.56 and 79.29 under the same causal scoring method. Our submitted model falls within this range, below the strongest variant. GPT-BERT (Charpentier and Samuel, 2024) combines masked and causal language modeling during training. These comparisons use individual submitted checkpoints and do not control for differences in training setup. Our conclusions rely on the controlled comparisons within our own implementation.

C Per-task results

Table 3 summarizes the changes in accuracy from text-only to multimodal training and the corresponding differences between architectures. Table 5 reports the accuracy of each condition as the mean and standard deviation across seeds 42, 43, and 44.

Entity tracking uses the updated official evaluation, which excludes items containing a "nothing" answer option.

D Per-phenomenon decomposition

Figure 3 reports the difference-in-differences for each of the 13 linguistic categories in BLiMP. It shows both the primary comparison with matched visual input and the variant with visual refresh during looping.

The categories follow the standard BLiMP order. Mean primary interactions range from −1.66 to 1.48 BLiMP points, and their signs change across seeds in 10 of the 13 categories. This analysis is exploratory, and we do not adjust for multiple comparisons.

[20] https://huggingface.co/datasets/daichi812/babyloop-mm-corpus

[21] https://osf.io/ad7qg/

Condition	BLiMP	Suppl.	EWoK
std_text	75.21±0.68	63.37±0.45	53.92±0.85
loop_text	76.16±0.45	63.56±0.68	54.19±0.50
std_mm	71.37±0.29	59.71±0.29	53.96±0.46
loop_mm	72.38±0.83	58.61±1.53	53.26±0.30
std_mm_id	72.48±0.69	58.78±2.31	53.56±0.19
loop_mm_prefix	72.31±0.63	59.82±1.35	53.58±0.76

Condition	COMPS	Entity
std_text	56.14±0.47	18.49±0.88
loop_text	55.87±0.17	17.00±0.53
std_mm	55.51±0.24	18.07±0.75
loop_mm	55.86±0.44	18.85±0.57
std_mm_id	55.51±0.27	18.47±0.36
loop_mm_prefix	55.14±0.24	18.45±1.74

Table 5: Zero-shot accuracy (%) on the final checkpoints, reported as the mean and standard deviation across three seeds. All scores use the official evaluation pipeline, and the BLiMP results match Table 1.

E Per-seed results

Table 6 reports the BLiMP score for each seed in the six main conditions. These scores underlie the means and standard deviations reported in Table 1.

Condition	s42	s43	s44	mean±sd
std_text	75.23	75.88	74.53	75.21±0.68
loop_text	76.61	75.71	76.17	76.16±0.45
std_mm	71.62	71.05	71.44	71.37±0.29
std_mm_id	71.68	72.93	72.82	72.48±0.69
loop_mm	71.56	73.21	72.36	72.38±0.83
loop_mm_prefix	72.34	71.66	72.92	72.31±0.63

Table 6: BLiMP accuracy (%) on the final checkpoints for seeds 42, 43, and 44. The looped model scores below the standard model for text-only seed 43 and for multimodal seed 42 with visual refresh. With matched visual input, the looped model scores higher for all three seeds.

F Learning trajectories

Figure 4 tracks BLiMP accuracy across 28 checkpoints for the four submitted models. The curves use the fast-evaluation subset and one seed per condition, so they describe individual training runs. The horizontal axis shows cumulative training words. Both multimodal models briefly outperform their text-only counterparts early in training. The text-only models later overtake them, and the gap fluctuates rather than narrowing steadily. By the end of training, multimodal accuracy is about 3.6 BLiMP points lower in both architectures. The early advantage could reflect differences in the language data, the visual input, or both. The trajectories alone do not separate these contributions.

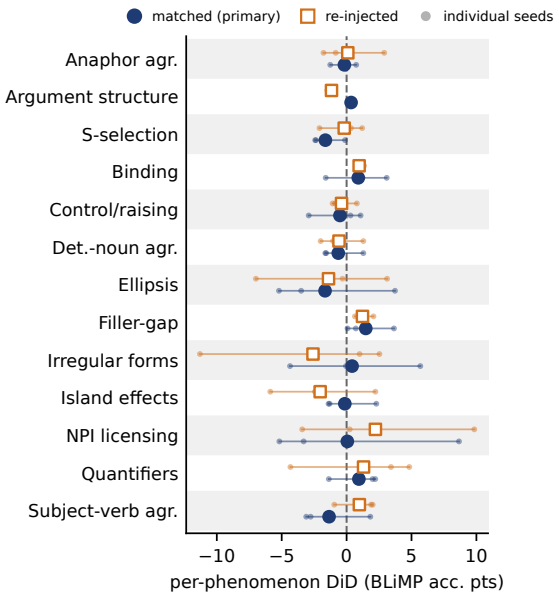


Figure 3: Interactions across the 13 BLiMP linguistic categories. Filled circles show three-seed means for the primary comparison with matched visual input, and open squares show means for the visual-refresh variant. Small dots show individual seed estimates; whiskers span their minimum and maximum values.

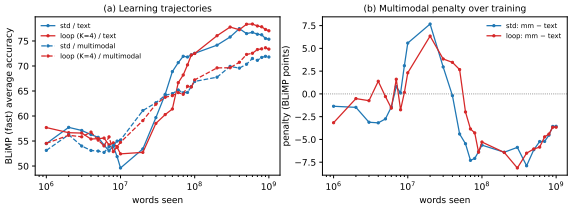


Figure 4: (a) BLiMP accuracy on the fast-evaluation subset at 28 checkpoints, using one submitted seed per condition. (b) Multimodal minus text-only accuracy within each architecture. Both panels plot cumulative training words on a logarithmic horizontal axis.

Figure 5 tracks held-out caption loss for the standard and looped text-only models, using seed 42. We evaluate all 28 checkpoints on the same 20,000 captions selected after the training-caption cutoff. Loss is measured as negative log-likelihood in nats per token. Both models reach their lowest held-out loss at 400M cumulative training words. By the end of training, loss has increased by about 0.37 nats per token in both models. This pattern is consistent with overfitting during later training. At the final checkpoint, held-out loss is 5.84 for the standard model and 5.83 for the looped model. Thus, the looped model's higher BLiMP accuracy is accompanied by little difference in held-out caption loss.

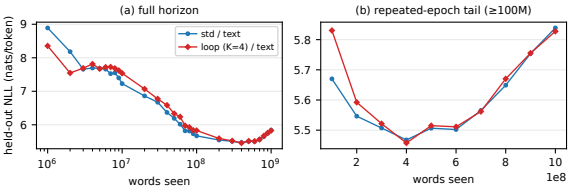


Figure 5: Held-out caption loss (nats per token) for the standard and looped text-only models, using 20,000 captions and seed 42. Cumulative training words are plotted on a log scale for all 28 checkpoints in (a), and on a linear scale from 100M onward in (b).

G Shuffled-features diagnostic

We evaluate the final checkpoints of the four main multimodal conditions, using three seeds per condition. All models are evaluated on the same fixed sample of 50,000 training captions. We change the visual features only at evaluation time, without retraining the models. Each caption is evaluated with its correct image features, features shuffled across the sampled captions, and features replaced by zeros. The shuffled result is averaged over three fixed permutations. Caption text, batching, and numerical precision are held constant across these conditions.

For each caption, we calculate the average token-level negative log-likelihood using the correct preceding tokens. Visual prefix positions are excluded from the loss. The shuffled-minus-correct difference is calculated separately for each caption. Because the captions come from the training data, this diagnostic measures visual reliance on familiar examples. It does not establish generalization to unseen image–caption pairs.

Cell	correct	shuf.	zero	Δ_{shuf}
`std_mm`	1.87	3.33	3.78	1.453±0.005
`std_mm_id`	1.87	3.32	3.31	1.451±0.006
`loop_mm`	1.84	3.35	3.74	1.516±0.002
`loop_mm_prefix`	1.85	3.34	3.78	1.485±0.009

Table 7: Caption negative log-likelihood (nats per token) under correct, shuffled, and zeroed image features, averaged equally across captions and then across three training seeds. Δ_{shuf} reports the shuffled-minus-correct difference as the mean and standard deviation across training seeds.

For every model and seed, at least 98.6% of captions have higher loss with shuffled features, averaged over the three permutations. This shows that correct image–caption pairing helps prediction on these training examples. With MLP connectors, zeroed features produce higher loss than shuffled features. With the identity connector, the losses are similar. Zeroing replaces the usual range of image features with a constant input, so the loss change need not reflect only the removal of visual information.

H Visual question answering

We use the 2024 BabyLM `vqa_filtered` task, based on VQAv2 questions (Goyal et al., 2017) and MS-COCO images (Lin et al., 2014). Each of the 25,230 questions has eight candidate answers. Random choice therefore gives 12.5% accuracy. Given an image and a question, the model scores each candidate answer by summing its token log-probabilities. We select the candidate with the highest score, without length normalization.

For the main conditions, we extract one global feature vector per VQA image using frozen DINOv2 ViT-B/14. These features have the same input format as the training features, but are extracted by us rather than supplied by the organizers. We report the resulting scores as diagnostics, not official leaderboard results. Each model is evaluated with correct, shuffled, and zeroed image features. The shuffled condition uses one fixed permutation of the feature rows. Questions, candidate answers, and item order are kept fixed across conditions and models. We use an exact McNemar test to compare correct and shuffled features within each model and seed. The test uses questions answered correctly under one feature condition but incorrectly under the other.

Cell	correct	shuf.	zero	$\Delta_{\text{cor}-\text{shuf}}$
`std_mm`	47.4±1.1	42.9±0.6	44.2±2.3	+4.6±1.0
`std_mm_id`	45.1±3.6	41.8±3.1	42.6±3.6	+3.3±1.2
`loop_mm`	45.1±4.8	41.0±5.3	35.1±14.1	+4.1±1.0
`loop_mm_prefix`	41.2±4.8	37.3±4.8	40.2±3.5	+3.8±0.6
`std2x_mm`	38.6±13.3	34.9±12.4	33.0±12.9	+3.7±1.0
`std4x_mm`	39.1±6.7	35.7±5.4	30.6±6.5	+3.4±1.4
`subset`‡	44.9	41.8	40.4	+3.1
`rich`‡	45.1	39.0	42.2	+6.1
`hive`‡	41.6	35.4	37.6	+6.2
`hive_k8`‡	27.1	20.4	24.7	+6.7

Table 8: VQA accuracy (%) with correct, shuffled, and zeroed image features. $\Delta_{\text{cor}-\text{shuf}}$ is the correct-minus-shuffled difference in percentage points. Values are mean ± standard deviation across three training seeds, except for the single-seed exploratory models (‡) trained on a reduced image subset.

Correct image features yield higher VQA accuracy than shuffled features in all twelve model–seed combinations. The gains range from 2.4 to 5.6 percentage points, with $p < 10^{-25}$ in every McNemar

test. With zeroed features, loop_mm scores 18.9% for seed 43, compared with 42.7% and 43.8% for seeds 42 and 44. This large variation shows that responses to zeroed features can differ substantially across training seeds. We therefore interpret this condition cautiously when assessing visual reliance.

Deeper standard models. The deeper standard models are trained on the same official visual features and evaluated with the same VQA protocol as the main models. With correct image features, the 24-layer model scores 38.6% and the 48-layer model scores 39.1%. Both means are below the 12-layer standard model's 47.4%. Variation across seeds is substantial, with standard deviations of 13.3 percentage points for the 24-layer model and 6.7 for the 48-layer model. These runs provide no clear evidence of a VQA benefit from increasing standard-model depth.

Exploratory visual inputs. In Table 8, the common loop_mm_ prefix is omitted from the names of the four exploratory models marked ‡. Each exploratory model is evaluated with the visual feature format used during its training: a single global vector, patch-level vectors, or vectors from multiple encoder layers. These models were trained on a reduced image subset using self-extracted features (Appendix I). Comparisons with the main models therefore combine changes in training data and visual representation. The single-vector reference model scores 44.9%, with a 3.1-point advantage for correct over shuffled features. The patch-feature model scores 45.1%, with a 6.1-point advantage. Thus, the larger effect of shuffling is accompanied by similar accuracy with correct images. Hierarchical injection yields 41.6% accuracy and a 6.2-point correct-minus-shuffled difference. The $K = 8$ training recipe yields 27.1% accuracy and a 6.7-point difference. Although the latter still benefits from correct image pairing, its overall VQA accuracy is substantially lower. These exploratory results use one training seed per condition. Additional runs are needed to determine whether the observed differences persist across seeds.

I Exploratory: richer, staged visual signals

We explore whether richer visual features and staged visual input improve BLiMP performance in looped models. We compare four conditions, moving from a single global image vector to training setups inspired by HIVE (Zhang et al., 2026).

Retrieving images for richer features. We match the released captions to their entries in Localized Narratives and Conceptual Captions, then retrieve the corresponding images. We use the frozen DINOv2 ViT-B/14 encoder to extract visual features from these images. As a spot check, we compare self-extracted and official features for 30 retrieved Localized Narratives images. All 30 pairs have cosine similarity of at least 0.9995. The highest similarity to an unrelated official feature vector is 0.42.

Not all captions could be matched to retrievable images. The resulting subset contains 761,334 Localized Narratives pairs and 1,191,243 Conceptual Captions pairs. This gives 38.2M caption words, compared with 50M in the main experiments. Localized Narratives accounts for about 71% of the caption words in this subset, compared with about 55% in the main experiments. All four exploratory conditions use the same reduced subset, with identical caption-to-feature row mappings. We train each condition until cumulative exposure to text and caption words reaches 1B words. Text and caption batches are sampled with equal probability, which does not imply equal word exposure because caption batches contain fewer words. These experiments are analysis-only and were not submitted to the competition.

Experimental conditions. All four conditions use MLP connectors and a peak learning rate of 6×10^{-4}. They use the same word-based learning-rate schedule and batch settings as the main experiments, with one training seed per condition. The peak learning rate differs from the 1.5×10^{-3} used for the main looped models.

- loop_mm_subset uses a single global feature vector from the final encoder layer, supplied once as a visual prefix, with four loop iterations. It provides a reference for the other exploratory conditions on the same reduced dataset.
- loop_mm_rich replaces the single global vector with 16 patch-level feature vectors from the final encoder layer. These are supplied once as visual prefix tokens, while the model still uses four loop iterations.
- loop_mm_hive uses 16 feature vectors from each of encoder blocks 3, 6, 9, and 12. Each block has its own MLP connector. Block-3 features

form the initial visual prefix, while features from blocks 6, 9, and 12 are added before the second, third, and fourth loop iterations. The separate connectors add approximately 3.5M parameters relative to `loop_mm_rich`. This comparison therefore changes the source of visual features, their injection schedule, and the number of connector parameters together.

- `loop_mm_hive_k8` uses eight loop iterations and repeats the sequence of visual features from encoder blocks 3, 6, 9, and 12. Gradients are propagated only through the final four iterations (Zhang et al., 2026). This comparison combines changes in loop count, feature scheduling, and gradient propagation.

BLiMP results. The single-vector reference scores 70.60 on BLiMP. Replacing it with patch-level features gives 70.47, while hierarchical injection gives 70.87. These single-seed results do not establish a benefit from richer or staged visual input. The $K = 8$ training recipe scores 55.36 on BLiMP, substantially below the other three conditions. Because this recipe changes loop count, feature cycling, and gradient propagation together, the decrease cannot be attributed to any one of these changes. The $K = 8$ condition also inherits the learning rate used for the $K = 4$ exploratory models. Further tuning would be needed to determine whether its low BLiMP score persists under other learning rates.

Cell	(adds)	BLiMP
`loop_mm_subset`	anchor	70.60
`loop_mm_rich`	+richness	70.47
`loop_mm_hive`	+staging	70.87
`loop_mm_hive_k8`	+recipe	55.36

Table 9: BLiMP accuracy (%) on the full evaluation set for the four exploratory conditions, each trained with seed 42. All conditions use the same reduced image subset and a peak learning rate of 6×10^{-4}. VQA results for these models are reported in Appendix H, Table 8.

Zeroing features from individual encoder blocks. To probe dependence on features from each encoder block, we zero one block's features at evaluation time while leaving the others unchanged. We apply this diagnostic to `loop_mm_hive` and `loop_mm_hive_k8`, both trained with seed 42. Each comparison uses the same fixed subset of 6,000 VQA questions.

For `loop_mm_hive`, zeroing block-12 features lowers accuracy from 40.6% to 35.5%. The correct-minus-shuffled difference also falls from 6.2 to 1.5 points. Zeroing block 3 instead raises accuracy by 3.7 points, while zeroing blocks 6 or 9 changes it by about 0.1 point. For `loop_mm_hive_k8`, zeroing block-12 features lowers accuracy from 27.1% to 26.1%. The correct-minus-shuffled difference falls from 6.9 to 0.6 points. Zeroing block 9 instead raises accuracy by 3.0 points.

In both models, zeroing block-12 features produces the largest reduction in the correct-minus-shuffled difference. This suggests that these features play an important role in the observed benefit from correct image pairing. These single-seed, evaluation-time probes do not establish that lower-layer features are unnecessary during training.

J K sweep

We compare text-only models at $K = 1, 2, 4$, and 8. The $K = 1$ standard model and $K = 4$ looped model are the main experimental conditions, each trained with three seeds. The additional $K = 2$ and $K = 8$ conditions use one seed each. Text-input injection is enabled for $K = 2, 4$, and 8, and disabled for the $K = 1$ standard model. Comparisons with the standard model therefore change both the number of iterations and text-input injection.

At the shared peak learning rate of 1.5×10^{-3}, the $K = 8$ run uses micro-batches of 32 sequences and 16 gradient accumulation steps. This preserves the 512 sequences per optimizer update used by the other conditions, which use micro-batches of 64 and eight accumulation steps. All looped models in this text-only comparison propagate gradients through every iteration. They use no visual input or gradient truncation, unlike the $K = 8$ multimodal recipe in Appendix I.

Learning-rate probes for $K = 8$. Before starting the 1B-word runs at $K = 8$, we compare four peak learning rates in probes trained for 100M cumulative words. The candidates are 3×10^{-4}, 6×10^{-4}, 1×10^{-3}, and 1.5×10^{-3}. We select the rate with the lowest final training loss among numerically stable runs. Final training loss decreases from 3.03 to 2.86 as the learning rate increases across the tested range. The lowest loss is obtained at 1.5×10^{-3}, which is also the rate used for the main looped model. We train the 1B-word runs at both this rate and the next-best candidate, 1×10^{-3}.

1B-word runs				K=8 probes	
K	peak LR	seeds	BLiMP	LR	loss
1	1.5×10^{-3}	3	75.21±0.68	3×10^{-4}	3.03
2	1.5×10^{-3}	1	75.28	6×10^{-4}	2.96
4	1.5×10^{-3}	3	76.16±0.45	1×10^{-3}	2.93
8	1.5×10^{-3}	1	76.35	1.5×10^{-3}	2.86
8	1×10^{-3}	1	75.42		

Table 10: Left: BLiMP accuracy on the full evaluation set after 1B cumulative training words. The $K = 1$ and $K = 4$ rows report means and standard deviations across three seeds; the other rows report individual runs with seed 42. Right: final training loss from the four $K = 8$ learning-rate probes, each trained for 100M cumulative words.

Results and scope. At the shared learning rate, $K = 2$ scores 75.28 on BLiMP and $K = 8$ scores 76.35. The main $K = 4$ condition has a three-seed mean of 76.16 with a standard deviation of 0.45. At the alternative learning rate of 1×10^{-3}, $K = 8$ scores 75.42. These results provide no clear evidence that eight iterations improve on four under the tested settings. However, with only one seed for each additional condition, they do not establish an optimal loop count. The lowest probe loss occurs at the highest learning rate tested. This does not establish that the optimal learning rate is independent of loop count.

K Modality-order curriculum

We explore training schedules that gradually shift between image–caption batches and text-only batches. The probability of selecting a text-only batch, $p_{\text{text}}(w)$, changes with cumulative training words w. In `v2l`, p_{text} increases linearly from 0 at the start of training to 1 at 1B cumulative words. In `l2v`, it decreases linearly from 1 to 0 over the same horizon. The flat control keeps $p_{\text{text}} = 0.5$ throughout training.

Text-only and caption batches contain different numbers of words, so a symmetric schedule of batch-selection probabilities does not ensure equal word exposure to the two sources. These schedules should not be interpreted as holding cumulative word exposure to each source fixed. All curriculum conditions use the `loop_mm_prefix` architecture: four iterations, an MLP connector, and one visual prefix token supplied at the start. They use the same 50M-word text corpus, 50M-word caption corpus, and official visual features as the flat control. We train `v2l` and `l2v` with three seeds each, and `v2l_short` with one seed.

Cell	p_{text}	BLiMP	VQA	$\Delta_{\text{cor}-\text{shuf}}$
flat	0.5	72.31±0.63	41.0±4.9	+3.7±0.6
v2l	$0 \to 1$	72.70±0.47	46.4±5.4	+6.0±0.8
l2v	$1 \to 0$	66.62±1.13	35.5±5.6	+2.8±1.9
v2l_short	$0 \to 1$, len.	70.66	23.7§	+3.2§

Table 11: BLiMP accuracy on the full evaluation set and VQA accuracy with correct image features (%). Scores with ± are means and standard deviations across three seeds; scores without ± use seed 42. p_{text} is the probability of selecting a text-only batch, and "len." denotes the caption-length schedule. The VQA columns for the three-seed conditions come from a re-evaluation of the final checkpoints in fp32; Table 8 reports the original bf16 evaluation of the flat model. §Single-seed value from the original bf16 evaluation. $\Delta_{\text{cor}-\text{shuf}}$ is the correct-minus-shuffled VQA difference in percentage points.

Reverse schedule. The reverse schedule lets us compare training that ends with text-only batches against training that ends with image–caption batches. This comparison alone does not separate effects of early training from effects of the data seen near the end.

Caption-length schedule. `v2l_short` adds a caption-length schedule to `v2l`. Training begins with the shortest 10% of captions, ranked by word count. The available fraction increases linearly with cumulative training words, reaching the full caption set at 1B words. The schedule changes which captions are available, without shortening the captions themselves. Shorter captions are not necessarily syntactically simpler, so this is a length-based sampling scheme rather than a direct test of syntactic complexity.

Predictions. We expected `v2l` to improve BLiMP accuracy at the cost of lower VQA accuracy. We also expected prioritizing shorter captions in `v2l_short` to yield an additional BLiMP gain.

Results and scope. The mean BLiMP score is 72.70 for `v2l`, compared with 72.31 for the flat control. The mean for `l2v` is 66.62, about 5.7 points below the flat control. The gap between the two curriculum directions is about 6.1 points. All three `l2v` runs score below the corresponding flat-control runs. The `v2l` difference from the flat control is positive for two seeds and negative for one, so its small positive mean should be interpreted cautiously. `v2l_short` scores 70.66 on BLiMP, compared with 73.20 for `v2l` using the same seed, 42. This single-seed result does not support the predicted benefit of prioritizing shorter captions.

Across three seeds, VQA accuracy with correct image features is 46.4% for `v2l`, 41.0% for the flat control, and 35.5% for `l2v`. The `v2l` difference from the flat control is positive for two seeds and negative for one. Its mean paired difference is $+5.4$ percentage points, with a 95% confidence interval of $[-19.4, 30.1]$. The predicted drop in VQA accuracy under `v2l` is not observed in the three-seed mean, but the higher mean does not establish a reliable improvement across seeds. These scores come from a re-evaluation of the final checkpoints in fp32 with the same protocol for all nine models; the single-seed `v2l_short` model scores 23.7% under the original evaluation. The results show an asymmetry between the two training schedules, but do not identify its cause. In particular, they do not rule out effects of recent training data or differences in cumulative exposure to each source.

L No-image controls for the looped model

Batch-matched control. We train the looped model on the language data of `loop_mm_prefix` with the same batches, word schedule, and initialization, omitting only the visual prefix (`loop_mm_no_image_matched`; Section 5.3). Table 12 reports the per-seed BLiMP scores and the paired contrasts, with unadjusted 95% confidence intervals from a t distribution with df$=2$. All three loop-side conditions used in these contrasts were evaluated at pipeline revision `3d57ddc`. We re-evaluate the text-only and multimodal references at that revision and use the existing matched-control records from the same revision (Appendix A).[22]

	s42	s43	s44	mean±sd	95% CI
control (looped)	70.12	71.38	69.88	70.46±0.81	
Visual-prefix effect (multimodal − matched control)					
looped	+2.23	+0.28	+3.04	+1.85±1.42	[−1.67, 5.37]
Language-data and batching change (text-only − matched control)					
looped	6.50	4.33	6.29	5.71±1.20	[2.73, 8.68]

Table 12: Loop-side batch-matched no-image control, BLiMP accuracy (%) and paired contrasts in percentage points. The comparisons use `loop_mm_prefix`, `loop_mm_no_image_matched`, and `loop_text`, with three seeds per condition.

Packed controls. We also train a looped text-only model on the multimodal conditions' language data, packing text and captions into fixed-length blocks. This control, `loop_text_mmdata`, scores 71.31±0.20 on BLiMP across three seeds.[23] The corresponding standard-model packed control scores 70.56 ± 0.19 (Section 5.3).[24] Both controls score below their counterparts trained on the original text-only corpus (Table 1). These results are consistent with changes in language data contributing to the BLiMP drop in both architectures. Under the fixed word budget, the standard packed control uses 6,221 optimizer updates, compared with 6,827 for the original text-only model. Thus the data change is not isolated from its consequences for the number of updates.

The packed controls use a different batching procedure from multimodal training. Comparisons with the multimodal models therefore combine changes in batching and visual input, rather than isolating the effect of vision. The looped packed control does not replace the batch-matched control described in Limitations.

[22] All contrasts in Table 12 use the common-revision per-seed run records. The re-evaluated reference scores differ from the historical scores in Table 6 by at most 0.01 percentage points.

[23] The per-seed scores are 71.54, 71.20, and 71.18 for seeds 42, 43, and 44.

[24] The standard-model scores are 70.34, 70.67, and 70.68 for seeds 42, 43, and 44. The first two records lack an evaluation-revision identifier; seed 44 and all three looped packed controls record `3d57ddc`. We treat the packed comparisons as supporting evidence, separate from the common-revision decomposition.

URCHIN: A Horizontal Spiking Language Model for Data-Constrained Pretraining

Po-Han Chiang
National Yang Ming Chiao Tung University
phc@nycu.edu.tw

Abstract

The BabyLM challenge measures how much language a model can learn from developmentally-plausible, child-scale data rather than internet-scale corpora, yet prior language models forgo the biological constraints of the neural circuitry that acquires human language: spiking neurons separated into excitatory and inhibitory populations wired by a recurrent lateral connectome. This paper presents URCHIN (Unified Recurrent Connectome with Horizontal Integrate-and-fire Neurons), which applies the Parallelized Hierarchical Connectome Spiking State-space Model (PHCSSM) to language modeling: leaky integrate-and-fire neurons coupled by a Dale's-law lateral connectome resolve each token through a multi-transmission loop that recirculates activity to a fixed point. The instantiation is deliberately minimal: a single horizontal layer of 128 neurons, no attention, and 4.23M parameters. Two implementations share one set of weights and produce identical benchmark scores, so URCHIN is trained once and deployed either way with no conversion step: a parallel state-space model (SSM) scan that is GPU-efficient for training, or an event-driven recurrent spiking neural network (RSNN) with constant-cost inference for CPU or neuromorphic edge deployment. Across all three BabyLM tracks (Strict-100M, Strict-Small, and Multilingual), URCHIN offers a biologically plausible, efficient, and directly deployable reference point.

1 Introduction

Scaling to internet-scale corpora has driven most recent progress in language modeling, but it has also moved these models far from the conditions under which humans acquire language. A child reaches broad linguistic competence from on the order of ten to a hundred million words, and the BabyLM challenge takes this as its benchmark, capping training data at a developmentally-plausible budget instead of allowing internet-scale text. The gap this exposes is not only about the amount of data. The dominant recipe, which the official BabyLM baselines (Radford et al., 2019; Charpentier and Samuel, 2024) and most entries (Charpentier et al., 2025) follow, is a feedforward transformer (Vaswani et al., 2017): a stack of roughly a dozen attention layers that processes each token in a single forward pass. In humans, by contrast, language is learned and run by the brain, whose circuits differ from these networks in ways usually treated as biological detail rather than as design guidance: neurons communicate through discrete spikes, each obeys Dale's law and is purely excitatory or purely inhibitory, connectivity within a region is recurrent and lateral rather than feedforward, and the system as a whole operates at a tiny energy budget. Spiking neural networks that adopt these constraints are a rich and active research area, yet they have largely stayed out of language modeling for a practical reason: spikes are event-driven and inherently sequential, so such networks are hard to parallelize and slow to train on the token volumes that language modeling requires. This paper asks whether that barrier can be crossed cheaply: can a minimal, biologically-constrained spiking network learn language under the BabyLM budget, and what does it cost to train and to deploy?

The model introduced here is URCHIN (Unified Recurrent Connectome with Horizontal Integrate-and-fire Neurons).[1] It adapts a recent architecture, the Parallelized Hierarchical Connectome Spiking State-space Model (PHCSSM) of Chiang (2026), which folds a set of biological priors into a trainable spiking network and can be evaluated either as a state-space model (SSM) with parallel scans or as an event-driven recurrent spiking neural network (RSNN), giving the same result both ways.

[1] The name echoes the spiny sea urchin, a fitting image for a spiking network of only a few million parameters.

Proceedings of the Second BabyLM Workshop, pages 563–574
October 29, 2026 ©2026 Association for Computational Linguistics

The parallel form is what removes the training bottleneck, so a biologically-constrained spiking network can be trained on language-model-scale data, while the serial form preserves the event-driven, energy-frugal behavior for deployment. URCHIN carries PHCSSM over to autoregressive language modeling: leaky integrate-and-fire (LIF) neurons coupled only by lateral connections, where a token is resolved not in one forward step but by a multi-transmission loop that recirculates activity through those connections, reusing one shared synapse and neuron nonlinearity at every step until it settles to a fixed point. The name records the two properties that define the method: a unified recurrent connectome, a single weight set shared between training and deployment, and horizontal, lateral-only connectivity. Nothing in this design fixes the number of layers, and the entry to the challenge is deliberately the smallest useful instance, a single horizontal layer of 128 neurons. The minimalism is deliberate: earlier spiking language models (LMs) such as SpikeGPT, SpikeLM, SpikeLLM, and SpikingBERT are surrogate-gradient spiking transformers or RWKV backbones, trained at scale, without biological connectivity constraints, and outside the BabyLM setting; URCHIN instead keeps the biological constraints and asks how far a single tiny recurrent layer can go.

URCHIN is positioned as a contribution to efficiency and biological plausibility. The contribution is threefold.

First, this paper presents a minimal spiking language model with biological constraints in the BabyLM setting. URCHIN is the first spiking network with a Dale's-law excitatory/inhibitory (E/I) connectome and an event-driven deployment mode that is trained and evaluated under the BabyLM data constraints across all three 2026 tracks, and it does so here in a minimal single-layer form in which the same neurons carry the input and the output and are joined only by lateral connections.

Second, this paper shows train and deploy score equivalence for a spiking language model. Parallel URCHIN (the parallel-scan SSM training backbone) and Serial URCHIN (the event-driven RSNN deployment backbone) share one set of weights and give the same benchmark scores across the full evaluation suite, agreeing to the floating-point floor of the two integration orders. This property was established for PHCSSM in the time-series setting, and is confirmed here to carry over to language modeling. It is not the same as ANN-to-SNN (artificial neural network to spiking neural network) conversion equivalence, because URCHIN is natively spiking and needs no conversion step.

Third, URCHIN is shown to be efficient in both parameters and compute. The spiking core is a 34K-parameter single-layer recurrent circuit, which is 0.8 percent of the 4.23M total; the rest is the token embedding and the output projection. Pretraining costs about 15 to 50 times less compute than the GPT-2 and GPT-BERT baselines.

2 Related Work

Spiking language models. SpikeGPT (Zhu et al., 2024) introduces a generative RWKV-style spiking language model with binary activations. SpikeLM (Xing et al., 2024) uses bidirectional ternary firing. SpikeLLM (Xing et al., 2025) scales spiking computation toward large language models. SpikingBERT (Bal and Sengupta, 2024) targets encoder-style text classification, and SpikingBrain 2.0 (Pan et al., 2026) studies brain-inspired foundation models. These systems show that spiking computation is compatible with language modeling. They use spiking transformers or RWKV backbones trained with surrogate gradients (a smooth approximation that lets the non-differentiable spike be backpropagated), without lateral recurrent connectivity between neurons or biological connectivity constraints, they are trained without a developmental data budget, and they are not BabyLM entries.

Spiking state-space models. SpikingSSMs (Shen et al., 2025) apply sparse spiking state-space models to long sequences, with a diagonal state-space recurrence in which neurons carry no lateral connections to one another. URCHIN differs on several points at once: its neurons are joined by a lateral recurrent connectome, it is biologically constrained through Dale's law on that connectome, and it is trained under the BabyLM data budget.

ANN-to-SNN conversion. A separate line of work trains a conventional rate-coded ANN and transfers its weights to a spiking network that emulates the ANN's continuous activations through firing rates over a deployment-time window (Cao et al., 2015; Rueckauer et al., 2017; Sengupta et al., 2019; Bu et al., 2022). This inherits the source ANN's parallel-training scalability but at the cost of a conversion-stage accuracy gap and an emu-

lation window of tens to hundreds of timesteps. URCHIN is instead natively spiking and trained directly in the spiking regime; its parallel and serial modes are two views of a single weight set, so its train-and-deploy equivalence needs no conversion step and incurs no conversion gap.

BabyLM. The challenge (Choshen et al., 2026) and its GPT-2 (Radford et al., 2019) and GPT-BERT (Charpentier and Samuel, 2024) baselines define the data budget and the evaluation suite used here. Most submissions are transformers, but a line of work explores non-transformer and subquadratic architectures for sample-efficient pre-training, such as the recurrent BabyHGRN (Haller et al., 2024); URCHIN extends this line to a biologically-constrained spiking model.

PHCSSM. The model introduced in Chiang (2026) is the first spiking SSM to combine five biological constraints: lateral (horizontal) connectivity, Dale's law, adaptive leaky integrate-and-fire dynamics (ALIF), short-term plasticity (STP), and spike-timing-dependent plasticity (STDP). It is also the first SSM whose single set of weights runs in both a parallel-scan mode for training and a serial event-driven RSNN mode for deployment, with no ANN-to-SNN conversion. It was evaluated only on the UEA MTSCA (the multivariate time-series classification archive), over signal domains such as EEG, ECG, spectral, and worm-motion data, and had not been used for language modeling. URCHIN is its first application to language, following the common BabyLM practice of adapting an established architecture to the challenge; Section 3 describes what carries over from PHCSSM and what changes.

To the best of the author's knowledge, no biologically-constrained spiking language model of this form, in which the neurons communicate by discrete spikes, are each exclusively excitatory or inhibitory, and are wired by a recurrent lateral connectome, has been trained and evaluated under the BabyLM data budget before URCHIN.

3 The URCHIN System

URCHIN is the language-modeling instance of PHCSSM, reduced to a single horizontal layer. This section describes the components that matter for the submission; the full neuron and synapse equations, with their decay and reset terms and the parallel-scan derivation, are given by PHCSSM

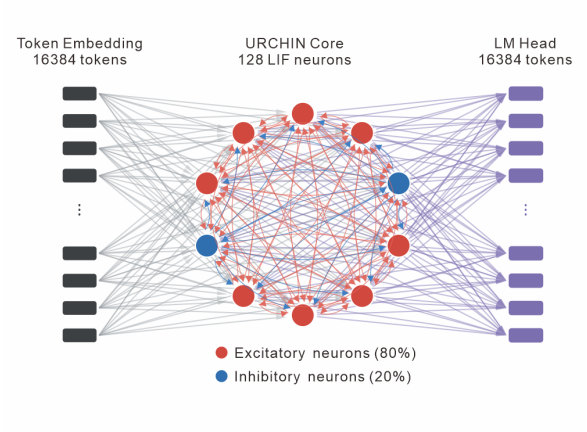


Figure 1: URCHIN architecture. The embedding of the current token is injected as input current into a single layer of excitatory (red) and inhibitory (blue) LIF neurons joined only by lateral connections; a linear LM head reads the membrane voltage of the same neurons out to the 16,384-entry vocabulary. Each token is processed by a multi-transmission loop, a fixed number of lateral transmission steps that settles to a fixed point. Parallel URCHIN evaluates this with parallel scans for training; Serial URCHIN runs the identical dynamics event-driven for deployment.

(Chiang, 2026). The submitted model is a single configuration whose architecture is unchanged across all three tracks.

Relation to PHCSSM. URCHIN uses the same architecture as PHCSSM. The LIF neurons, the Dale's-law E/I synapse, the lateral connectome, and the within-token multi-transmission recurrence (with its parallel-scan and event-driven implementations) are shared by both models. What changes is the input and output: in place of the signal encoder and classification head that PHCSSM used for time-series classification, URCHIN feeds the token embedding as input current and reads the membrane voltage through a linear LM head, trained with a causal next-token objective. Within the core, the submitted model uses only plain LIF neurons; the ALIF, STP, and STDP functions are not used here.

A single horizontal layer. The model is one recurrent region of 128 LIF neurons (Bellec et al., 2018), with no depth and no region hierarchy (Figure 1). A LIF neuron sums its input into a membrane voltage that leaks toward a resting value over time; when the voltage reaches a threshold the neuron emits a spike and the voltage resets. The token embedding is injected as input current into these neurons, and the LM head reads the membrane

voltage of the same neurons, so a single population carries both the input and the output. The neurons are connected only by horizontal (lateral) connections; there are no feedforward layers between an input population and an output population, because there is only one population.

Dale's law and E/I structure. The 128 neurons are split into excitatory and inhibitory populations, and synaptic weights obey Dale's law (Cornford et al., 2021; Balwani et al., 2025): a sign constraint holds each neuron's outgoing projections purely excitatory or purely inhibitory throughout training and inference. The lateral connectivity is full over the four E and I blocks, `[E->E, I->E, E->I, I->I] = [1,1,1,1]`. All three tracks use the same homogeneous population with an excitatory-to-inhibitory ratio of 80 to 20 and a single membrane time constant.

Multi-transmission loop and readout. For each input token the layer runs a fixed number of lateral transmission steps over its horizontal connections, a within-token loop that repeats until the state stops changing. URCHIN uses the plain-LIF forward pass of PHCSSM (Chiang, 2026). At step k and token position t, the token embedding e_t enters as input drive with gain α_{drive}, and each neuron sends its previous spikes back through the Dale-masked lateral weight $W_{\text{struct}} = W_{\text{syn}} \odot M_{\text{topo}}$ (W_{syn} sign-clamped so each neuron's outgoing weights keep one sign, M_{topo} the fixed lateral-topology mask, d the synaptic delay). Each neuron carries an excitatory-membrane state V_{exc} and a refractory-reset state V_{res} with learnable leak factors $\alpha_{\text{exc}}, \alpha_{\text{ref}} \in (0,1)$, applies the softplus nonlinearity $\zeta(x) = \log(1+e^x)$, and fires through the Heaviside H at threshold V_{th}, with $s_t^{\text{pre}} = H(V_{\text{exc},t} - V_{\text{th}})$ the preliminary crossing. For $k = 1, \ldots, K$, with the neuron states reset at the start of each step and $s^{(0)} = 0$,

$$I_t^{(k)} = W_{\text{struct}}\, s_{t-d}^{(k-1)} + \alpha_{\text{drive}}\, e_t, \tag{1}$$

$$V_{\text{exc},t} = \alpha_{\text{exc}} V_{\text{exc},t-1} + \zeta(I_t^{(k)}), \tag{2}$$

$$V_{\text{res},t} = \alpha_{\text{ref}} V_{\text{res},t-1} + \zeta(w_{\text{reset}}\, s_t^{\text{pre}}), \tag{3}$$

$$V_{\text{mem},t}^{(k)} = V_{\text{exc},t} - V_{\text{res},t}, \tag{4}$$

$$s_t^{(k)} = H(V_{\text{mem},t}^{(k)} - V_{\text{th}}). \tag{5}$$

The membrane voltage $V_{\text{mem}} = V_{\text{exc}} - V_{\text{res}}$ is the excitatory-membrane state minus the refractory reset, and the spikes $s^{(k)}$ of one step become the lateral input to the next. The loop settles to a within-token fixed point $s^{(k)} = s^{(k-1)}$; training uses 13 transmission steps (a target of 12 plus one buffer step) and evaluation uses 24, since additional steps at inference move the state closer to that fixed point. State is reset for each token, so this loop is a within-token recurrence, separate from the sequence axis, which is evaluated as a log-domain parallel prefix scan over the token positions. A linear LM head then reads the final membrane voltage of the same neurons out to the 16,384-entry vocabulary (the byte-pair-encoding (BPE) tokenizer provided by the BabyLM challenge),

$$\hat{y}_t = W_{\text{dec}}\, V_{\text{mem},t}^{(K)} + b_{\text{dec}}, \tag{6}$$

with the head weights $W_{\text{dec}}, b_{\text{dec}}$ untied from the token embedding. Training uses a causal next-token objective and backpropagation only.

Parallel URCHIN (SSM) and Serial URCHIN (RSNN). URCHIN has two implementations that share a single set of weights and are two mathematically equivalent views of the same dynamics. Parallel URCHIN is the SSM form: it casts the multi-transmission loop and the sequence recurrence as linear state-space recurrences and evaluates them in JAX (Bradbury et al., 2018) and PyTorch with parallel prefix scans, so it trains with backpropagation at transformer-like throughput; it is used for all training. Serial URCHIN is the RSNN form: it runs the identical dynamics event-driven, one timestep at a time, and is the deployment target for neuromorphic and edge hardware. The two forms are the same model up to floating-point summation order, and Section 5 reports their agreement.

Parameter budget. The submitted model has about 4.23M stored parameters, of which about 4.21M are trainable; the difference is the fixed Dale's-law connectivity mask, which is stored but not learned (Table 1). The token embedding and the untied LM head account for 99.2 percent of the total (two blocks of 16384 by 128). The spiking core is about 34K parameters (about 17K trainable; the remainder is the fixed Dale mask, Table 1), so URCHIN performs language modeling with a single 34K-parameter recurrent core.

4 Experimental Setup

Tracks and data. The same architecture is submitted to all three tracks of the BabyLM challenge: Strict-Small (10M words), Strict-100M

Component	Trainable	Total (stored)
Token embedding	2.10M	2.10M
LM head	2.10M	2.10M
Spiking core	17K	17K
Fixed Dale + input masks	0	16.6K
Total	**4.21M**	**4.23M**

Table 1: Parameter budget of the submitted model. Trainable parameters against total stored tensors. The fixed Dale's-law connectivity mask is stored in the checkpoint but is not trained.

(100M words), and Multilingual (100M words in English, Dutch, and Chinese) drawn from BabyBabelLM (Jumelet et al., 2025). The strict tracks use the official BabyLM corpus, drawn from sources including CHILDES (MacWhinney, 2000), Project Gutenberg (Gerlach and Font-Clos, 2018), OpenSubtitles (Lison and Tiedemann, 2016), and the Switchboard Dialog Act Corpus (Stolcke et al., 2000). Each track uses the tokenizer that the BabyLM challenge provides for it: a BPE tokenizer with a 16,384 vocabulary in all three cases, but a different tokenizer per track. The vocabulary size is therefore shared, while the token inventories differ; the Multilingual tokenizer covers English, Dutch, and Chinese and shares only about a quarter of its tokens with the English strict-track tokenizers. The full pretraining and fine-tuning hyperparameters for all three tracks are given in Appendix A (Table 5); the architecture is identical across tracks, and only the corpus, its tokenizer, and the schedule differ. Training uses a causal objective and backpropagation only, and stays within the BabyLM training budget (a fixed cap on epochs and tokens set by the challenge). The three submitted checkpoints (`URCHIN_BabyLM2026_StrictSmall`, `URCHIN_BabyLM2026_Strict`, and `URCHIN_BabyLM2026_Multilingual`), each packaging both the parallel and serial implementations under one set of weights, are publicly available on the Hugging Face Hub (`https://huggingface.co/phclab`).

Evaluation. Evaluation uses the official evaluation pipeline released by the BabyLM challenge (`babylm-eval`, commit 3d57ddc). It covers BLiMP (Warstadt et al., 2020), BLiMP-Supplement, EWoK (Ivanova et al., 2025), Entity-Tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), GlobalPIQA, reading-behavior fits, and the GLUE (Wang et al., 2019) fine-tuning suite, together with the multilingual suite for the ML track.

Metric	URCHIN	Baseline-GPT2	Baseline-Interaction
Overall Average	37.01	37.38	38.71
BLiMP	61.29	65.23	63.07
BLiMP-Supplement	56.05	57.25	58.13
EWoK	50.29	50.63	51.45
Entity-Tracking (deb.)	18.16	19.10	19.62
COMPS	50.70	51.81	51.29
GlobalPIQA	29.70	35.09	36.14
(Super)GLUE	59.47	63.80	63.62
Reading (combined)	**7.43**	5.63	5.10
Eye-Tracking	10.00	9.63	8.50
Self-paced Reading	4.85	1.64	1.69
AoA (official)	**0.0**	−12.15	0.0
NLP Average	46.52	48.99	49.04
Human-like Average	**3.71**	−3.26	2.55

Table 2: Strict-Small official results: URCHIN against the two official BabyLM baselines, the GPT-2 baseline (`BabyLM-2026-Baseline-GPT2-Strict-Small`) and the interaction baseline (`BabyLM-2026-Baseline-Strict-Small-Interaction`). Bold marks the metrics on which URCHIN matches or exceeds both baselines. AoA is decoupled to zero in URCHIN by design.

Fine-tuning. Fine-tuning uses the challenge pipeline with its optional `--lr` flag set to 3e-4 for the strict-track (Super)GLUE tasks and 1e-3 for the multilingual suite (Table 5), staying within the epoch budget under best-per-epoch early stopping.

5 Results

The same homogeneous single-layer model is submitted to all three tracks; they share one architecture and differ only in the training corpus and the schedule.

Comparison with the official baselines. The official per-track results are given in Tables 2, 3, and 4. URCHIN's advantage is concentrated in the human-like metrics: it matches or exceeds both official baselines on the Human-like Average on both strict tracks (3.71 and 3.59; Tables 2 and 3). The gap is driven by the Reading fits, on which URCHIN's combined score exceeds both baselines on each track, together with a non-negative AoA that URCHIN decouples to zero by design whereas both baselines score negative. On the language-understanding benchmarks URCHIN trails both baselines on most tasks, as expected at 4.23M parameters, the exception being Entity-Tracking on Strict-100M, where it exceeds both (Table 3). On the Multilingual track (Table 4) its 43.56 language

Metric	URCHIN	Baseline-GPT2	Baseline-Interaction
Overall Average	37.54	40.73	39.89
BLiMP	63.08	74.73	72.89
BLiMP-Supplement	52.57	65.00	65.09
EWoK	49.97	54.37	54.23
Entity-Tracking (deb.)	**18.39**	16.91	15.89
COMPS	52.73	55.85	55.37
GlobalPIQA	32.74	36.62	36.18
(Super)GLUE	61.19	67.75	67.28
Reading (combined)	**7.18**	6.93	3.43
Eye-Tracking	10.28	10.54	4.74
Self-paced Reading	4.08	3.32	2.13
AoA (official)	**0.0**	−11.58	−11.36
NLP Average	47.24	53.03	52.42
Human-like Average	**3.59**	−2.32	−3.96

Table 3: Strict-100M official results: URCHIN against the two official BabyLM baselines, the GPT-2 baseline (`BabyLM-2026-Baseline-GPT2-Strict`) and the interaction baseline (`BabyLM-2026-Baseline-Strict-Interaction`), formatted as in Table 2. Bold marks the metrics on which URCHIN matches or exceeds both baselines.

	En	Nl	Zh	Overall
URCHIN				
Zero-shot avg.	48.60	48.72	46.70	48.01
Fine-tune avg.	39.28	39.55	**41.47**	40.10
Language avg.	43.27	43.83	**43.57**	43.56
Baseline-GPT2 (en/nl/zh)				
Zero-shot avg.	53.63	55.05	47.25	51.98
Fine-tune avg.	43.09	39.82	40.69	41.20
Language avg.	47.60	46.92	43.31	45.94
Baseline-GPT2 (multilingual)				
Zero-shot avg.	35.42	49.04	41.71	42.06
Fine-tune avg.	31.51	29.76	31.68	30.98
Language avg.	33.18	38.76	35.69	35.88

Table 4: Multilingual official results: per-language zero-shot, fine-tune, and language averages for URCHIN and the two official GPT-2 baselines, `BabyLM-2026-Baseline-GPT2-en_nld_zho_equal` (en/nl/zh) and `babylm-multilingual-gpt2-baseline` (multilingual). All figures are from the official leaderboard. The Overall column is the mean of the three language averages; the zero-shot average includes the GlobalPIQA surprise task and the fine-tune average includes the POS surprise task. Bold marks cells where URCHIN matches or exceeds both baselines.

average is above the multilingual GPT-2 baseline (35.88) and, on Chinese, above the stronger trilingual (en/nl/zh) GPT-2 baseline (43.57 against 43.31), while remaining below that baseline overall (45.94). All values of offical baselines are from the BabyLM 2026 leaderboard (11 July 2026).

Comparison between Parallel and Serial modes. Parallel URCHIN and Serial URCHIN load the same weights, and the event-driven serial forward pass reproduces the parallel forward pass up to a small floating-point difference from the two integration orders (a log-domain prefix scan against a direct step recurrence). To make an item-level comparison of the difference between the two modes, both backbones are run locally on the released weights, since the official leaderboard reports only the submitted model's aggregate scores. Across the zero-shot suite (BLiMP, BLiMP-Supplement, EWoK, Entity-Tracking, COMPS, GlobalPIQA) and the fine-tuning suite, for the submitted models on all three tracks, the two implementations give the same benchmark scores: across the 66 benchmarks and tasks, together covering 445,447 individual predictions, only 90 predictions flip between the two modes, or 0.02 percent; accuracy is stable to the fourth decimal, the largest single-task difference is 0.7 points (on RTE, with 139 examples), and the Multilingual zero-shot suite is item-identical across all 170,759 of these predictions. A prediction can change only when its two candidate scores are separated by less than this floating-point difference, so it tips an argmax only on items already at the decision boundary: the two integration orders round such a near-tie in opposite directions, and only these 90 items sit that close, flipping either way and leaving the aggregate scores unchanged (Table 6, Appendix B). Fine-tuning is performed in the parallel mode and deployed serial for inference, so the fine-tuned scores are carried over by the same forward pass. Training and deployment are therefore the same model, not an ANN-to-SNN conversion.

Compute and parameter efficiency. Training compute is counted under a convention that counts matrix-multiply operations only, treats the token embedding as a lookup at 0 FLOP, counts the LM head as a full projection, and multiplies the forward cost by three for the backward pass. Under this convention a full pretraining run costs 2.29 PFLOP on Strict-Small, 23.6 PFLOP on Strict-100M, and 18.2 PFLOP on Multilingual. The official GPT-BERT and GPT-2 baselines, counted the same way with the transformer attention term included, cost 34.6, 1009, and about 915 PFLOP on the same three tracks. URCHIN therefore reaches comparable scores using about 15, 43, and 50 times less training compute. Wall-clock training is correspondingly

small, under an hour on a single H200 for every track (Table 5). Together with the 34K-parameter single-layer core, it is efficient in both compute and effective (non-embedding) capacity.

Deployment throughput. Both implementations were benchmarked on an NVIDIA H200 GPU and on a single CPU core (Intel Xeon 8480C, one thread, to emulate an edge or microcontroller compute budget), at batch size one over context lengths from 128 to 2048; the three tracks agree within about 2 percent. Because the two modes execute different amounts of work, the better mode depends on the hardware (Figure 2). On the GPU, Parallel URCHIN runs in near-constant time per forward pass, so its throughput grows with context length and reaches about 177,000 tokens per second at a 2048-token context, which is why it is used for training; Serial URCHIN holds a constant cost of about 13,000 tokens per second. On a single CPU core the ordering reverses: the event-driven Serial URCHIN reaches about 13,000 to 18,000 tokens per second while Parallel URCHIN falls to about 2,000, so serial is about seven times faster and is the intended edge and neuromorphic deployment path. The speedup comes from the event-driven core making a single pass instead of the 24-step transmission loop, which leaves the fixed vocabulary-projection head, rather than the spiking core, as the dominant per-token cost on a single core.

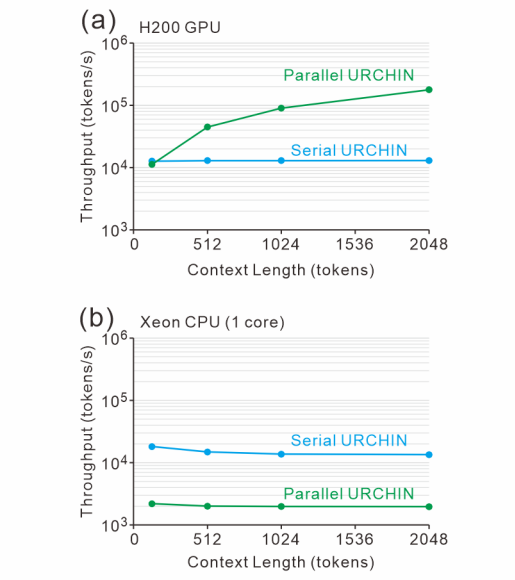


Figure 2: Single-sequence inference throughput of Parallel URCHIN (24 transmission steps, evaluation setting) and Serial URCHIN (event-driven), the mean of the three homogeneous tracks. On the H200 GPU (a) parallel throughput scales with context length while serial is flat; on a single CPU core (b) the event-driven serial path is about seven times faster. Batch size one.

6 Discussion

The results bear most directly on the biological motivation for the design. The GPT-2 and GPT-BERT baselines are 12-layer transformers with 12 attention heads and 768-dimensional hidden states (official baseline model cards and leaderboard, retrieved 11 July 2026); URCHIN is a single horizontal layer of 128 spiking neurons with no attention, yet it reaches a comparable overall level and, on the human-like metrics, predicts human reading behavior better than they do (Section 5). That the smallest and hardest-constrained model is the strongest on the human-like axis is the central observation: the biological constraints act as a useful inductive bias for human-like processing rather than as a handicap, even as they leave the model behind on grammatical and world-knowledge benchmarks. The effect is clearest on the less-saturated multilingual track, a direction worth pursuing.

The two execution modes matter less at the BabyLM scale than as a deployment path that larger spiking language models could inherit. Because Parallel and Serial URCHIN are the same weights under two integration orders, such a model can be trained once at transformer-like throughput on a GPU and then run unchanged as an event-driven spiking network, with no conversion step and no drop in benchmark scores. The absence of attention keeps serial inference at a constant per-token state, with no key-value cache that grows with the sequence, so its memory does not scale with context and it runs on a single CPU core or other edge device. Its event-driven spiking form is moreover a natural match for neuromorphic hardware, which is built for sparse, per-timestep spike computation rather than dense attention. Here that path runs through a 34K-parameter recurrent core (Section 5), and the same train-parallel, deploy-serial route would carry over to larger networks.

The one human-like axis on which URCHIN does not register a signal is age of acquisition (Chang and Bergen, 2022): the official AoA correlation for the submitted model is not statistically significant. This is expected rather than surprising. URCHIN is trained with a plain causal objective and no curriculum, and in particular its data is not ordered by age of acquisition, so a strong AoA

correlation is not something the training procedure is built to produce. The reading-time alignment that does appear therefore comes from the model's learned surprisals rather than from any developmental scheduling of the data; ordering the curriculum by acquisition is a separate design choice, left open here.

URCHIN is deliberately the smallest useful instance of PHCSSM: a single 128-neuron horizontal layer with no depth and no region hierarchy. This minimal choice isolates whether biological constraints can support language modeling at all under a developmental data budget, without confounding that question with capacity. Whether a wider or deeper network helps under the same data budget is left to future work. The parallel-scan formulation that makes training tractable applies equally to multi-layer and richer-cell variants of the framework. The result establishes that even a single tiny layer, under Dale's-law excitation and inhibition and lateral-only connectivity, already reaches competitive scores, which suggests these biological constraints are compatible with sample-efficient language modeling rather than an obstacle to it.

7 Limitations

The single small network bounds capacity, and URCHIN trails the transformer baselines on the grammatical and world-knowledge benchmarks such as BLiMP and EWoK. Learning is not part of the biological story: both pretraining and fine-tuning use standard backpropagation, so only inference runs as an event-driven spiking network, and local plasticity rules for on-device learning are left to future work. URCHIN's developmental plausibility is architectural rather than procedural: it trains on a plain causal objective, so the curriculum learning and other cognitively-motivated training strategies that much of the BabyLM community explores (Warstadt et al., 2023; Charpentier et al., 2025) are complementary to it and remain to be combined with the spiking network. Several results come from a single training seed and are reported without a variance estimate.

8 Conclusion

This paper presented URCHIN, a biologically-constrained spiking language model with a Dale's-law E/I population joined only by horizontal connections, and an event-driven deployment mode, here instantiated as a single horizontal layer and applied to data-constrained language modeling across all three BabyLM 2026 tracks (Strict-Small, Strict-100M, and Multilingual). Its spiking core is a 34K-parameter single recurrent layer, pretraining costs about 15 to 50 times less than the baselines, and Parallel URCHIN and Serial URCHIN give the same benchmark scores, agreeing to the floating-point floor of the two integration orders. URCHIN provides a biologically plausible and deployable reference point in the BabyLM design space, and a base for deeper, richer-cell, and local-learning variants in later work.

Acknowledgements

This work was supported by the National Science and Technology Council (NSTC), Taiwan (NSTC 114-2320-B-A49-027-; NSTC 114-2634-F-A49-006-; NSTC 114-2321-B-A49-003-; NSTC 114-2321-B-A49-014-). During the preparation of this work, the author used Claude (Anthropic) and Gemini (Google) for manuscript editing and code development. The author reviewed and verified all outputs and takes full responsibility for the content of this work.

Declaration of competing interest

The author declares the following competing interests: a patent application related to the work described in this paper has been filed.

References

Malyaban Bal and Abhronil Sengupta. 2024. SpikingBERT: Distilling BERT to train spiking language models using implicit differentiation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38.

Aishwarya H. Balwani, Alex Q. Wang, Farzaneh Najafi, and Hannah Choi. 2025. Constructing biologically constrained RNNs via dale's backpropagation and topologically informed pruning. *Science Advances*, 11(50):eadw4970.

Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. 2018. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 795–805.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. JAX: Composable transformations of Python+NumPy programs.

Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. 2022. Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks. In *International Conference on Learning Representations*.

Yongqiang Cao, Yang Chen, and Deepak Khosla. 2015. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Georges Gabriel Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan G. Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *Proceedings of the BabyLM Challenge (CoNLL)*.

Po-Han Chiang. 2026. Parallelized hierarchical connectome: A spatiotemporal recurrent framework for spiking state-space models. *arXiv preprint arXiv:2604.01295*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan G. Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Jonathan Cornford, Damjan Kalajdzievski, Marco Leite, Amélie Lamarquette, Dimitri Michael Kullmann, and Blake Aaron Richards. 2021. Learning to live with dale's principle: ANNs with separate excitatory and inhibitory units. In *International Conference on Learning Representations (ICLR)*.

Martin Gerlach and Francesc Font-Clos. 2018. A standardized Project Gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Computing Research Repository*, arXiv:1812.08092.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *Proceedings of the BabyLM Challenge*.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galván-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2025. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. *CoRR*, abs/2510.10159.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of ACL*, pages 3835–3855.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929.

Brian MacWhinney. 2000. *The CHILDES Project: The Database*, volume 2. Psychology Press.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of EACL*.

Yuqi Pan, Jinghao Zhuang, Yupeng Feng, Fangzhi Zhong, Siyu Ding, Xuerui Qiu, Shaowei Gu, Bohan Sun, Zhiyong Qin, Yibo Zhong, Lingtao Ouyang, Kun Yang, Zehao Liu, Yuhong Chou, Shurong Wang, Anjie Hu, Han Xu, Bo Xu, and Guoqi Li. 2026. SpikingBrain2.0: Brain-inspired foundation models for efficient long-context and cross-platform inference. *arXiv preprint arXiv:2604.22575*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. 2017. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682.

Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. 2019. Going deeper in spiking neural networks: VGG and residual architectures. *Frontiers in Neuroscience*, 13:95.

Shuaijie Shen, Chao Wang, Renzhuo Huang, Yan Zhong, Qinghai Guo, Zhichao Lu, Jianguo Zhang, and Luziwei Leng. 2025. SpikingSSMs: Learning long sequences with sparse and parallel spiking state space models. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Marie Meteer, and Carol Van Ess-Dykema. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–371.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Xingrun Xing, Boyan Gao, Zheng Zhang, David A. Clifton, Shitao Xiao, Li Du, Guoqi Li, and Jiajun Zhang. 2025. SpikeLLM: Scaling up spiking neural network to large language models via saliency-based spiking. In *International Conference on Learning Representations (ICLR)*.

Xingrun Xing, Zheng Zhang, Ziyi Ni, Shitao Xiao, Yiming Ju, Siqi Fan, Yequan Wang, Jiajun Zhang, and Guoqi Li. 2024. SpikeLM: Towards general spike-driven language modeling via elastic bi-spiking mechanisms. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K. Eshraghian. 2024. SpikeGPT: Generative pre-trained language model with spiking neural networks. *Transactions on Machine Learning Research*.

A Hyperparameters

Hyperparameter	Strict-Small	Strict-100M	Multilingual
Pretraining			
Corpus	`babylm_10m`	`babylm_100m`	`babylm_ml_100m`
Tokenizer vocabulary size	16,384	16,384	16,384
Context length	2,048	2,048	2,048
Synaptic delay	1	1	1
Sensory drive scale (`drive_alpha`)	0.3	0.3	0.3
Transmission steps (train / eval)	12 (+buffer 1) / 24	12 (+buffer 1) / 24	12 (+buffer 1) / 24
Convergence-loss weight (buffer-ramp)	800	800	800
Optimizer	AdamW (β_1 0.9, β_2 0.95)	AdamW (β_1 0.9, β_2 0.95)	AdamW (β_1 0.9, β_2 0.95)
Weight decay	0.1	0.1	0.1
Peak / min learning rate	1.5e-3 / 1e-5	1.5e-3 / 1e-5	1.5e-3 / 1e-5
LR schedule	warmup + cosine	warmup + cosine	warmup + cosine
Warmup steps	500	2,600	2,000
Batch size	16	16	16
Total training steps	5,050	51,900	40,000
Epochs	9.78	10.00	9.93
Training compute (PFLOPs)	2.29	23.57	18.17
Training wall-clock (1×H200, h)	0.08	0.64	0.48
Precision	float32	float32	float32
Random seed	42	42	42
Fine-tuning			
Tasks	GLUE (7 tasks)	GLUE (7 tasks)	9 tasks × (en, nl, zh)
Optimizer	AdamW (HF defaults)	AdamW (HF defaults)	AdamW (HF defaults)
Learning rate	3e-4	3e-4	1e-3 (1e-4 for POS)
Max epochs	20 (WSC 30)	20 (WSC 30)	50
Early stopping	best-per-epoch	best-per-epoch	patience 5
Batch size	32 (16 for long-seq)	32 (16 for long-seq)	64 (16 for POS)
Eval transmission steps	24	24	24
Random seed	42	42	42

Table 5: Pretraining and fine-tuning hyperparameters for the three tracks. The architecture (a single 128-neuron layer, 80:20 E/I, full `[1,1,1,1]` lateral connectivity, LIF neurons, no STDP) is identical across tracks. Fine-tuning tasks: the strict tracks use the seven (Super)GLUE tasks (boolq, mnli, mrpc, multirc, qqp, rte, wsc); the Multilingual track uses nine tasks (arc, belebele, bmlama, include, mnli, sib200, truthfulqa, xnli, pos), each in English, Dutch, and Chinese. The Multilingual POS fine-tuning matches the Multilingual classification fine-tuning except for the learning rate and batch size marked in POS, and additionally uses a maximum sequence length of 128 and a warmup ratio and weight decay of 0.1 and 0.01.

B Item-level parallel-serial agreement

Track	Task	N	Flips	Δ
Strict-Small	BLiMP (ZS)	59,875	4	$+0.000033$
Strict-Small	EWoK (ZS)	7,618	1	-0.000131
Strict-Small	MNLI (FT)	4,908	15	$+0.000204$
Strict-Small	QQP (FT)	20,215	9	-0.000049
Strict-Small	BoolQ (FT)	1,635	4	-0.001223
Strict-Small	RTE (FT)	139	1	$+0.007194$
Strict-100M	BLiMP (ZS)	59,875	4	0.000000
Strict-100M	EWoK (ZS)	7,618	4	0.000000
Strict-100M	Entity-Tr. (ZS)	6,780	5	$+0.000147$
Strict-100M	MNLI (FT)	4,908	9	0.000000
Strict-100M	QQP (FT)	20,215	8	$+0.000099$
Strict-100M	BoolQ (FT)	1,635	6	-0.002446
Multiling.	en/SIB-200 (FT)	201	1	-0.004975
Multiling.	en/XNLI (FT)	2,000	4	0.000000
Multiling.	nl/BMLaMA (FT)	1,213	3	-0.000824
Multiling.	nl/SIB-200 (FT)	201	2	0.000000
Multiling.	zh/MNLI (FT)	1,777	2	-0.000563
Multiling.	zh/XNLI (FT)	2,000	8	$+0.001000$

Table 6: Item-level parallel-serial disagreements. Across the three tracks the two modes are compared on 66 benchmarks and tasks, together covering 445,447 individual predictions. **48 of the 66 are perfectly item-identical between the parallel and serial modes; only the 18 listed below disagree at all, and only on 90 of the 445,447 predictions (0.02 percent).** Each row gives the item count N, the number of flipped items, and the change in the reported score Δ (serial minus parallel); every benchmark or task not listed has zero flips and $\Delta = 0.000000$. The entire Multilingual zero-shot suite falls in the identical set: all 16 of its benchmarks, covering 170,759 of these predictions over English, Dutch, and Chinese, agree item for item. Reading is excluded, as it is scored by surprisal correlation rather than accuracy. ZS = zero-shot, FT = fine-tune.

The Impact of Preference Optimization Data Selection: Teaching Practical Life Skills to BabyLM

Barbora Štěpánková[12] **and Vojtěch Lanz**[1] **and Jakub Szymanik**[2]
[1]Institute of Formal and Applied Linguistics, Charles University
[2]Center for Mind/Brain Sciences, University of Trento
barbora.stepankova320@student.cuni.cz,
lanz@ufal.mff.cuni.cz, jakub.szymanik@unitn.it

Abstract

This work investigates how the choice of preference optimization data affects small language models in the BabyLM setting. We apply SimPO to a GPT-2 model pretrained on 9M words, using 21 different 1M-word preference datasets, including UltraFeedback, which we cluster by topic, grammatical, and Reddit-based data. We show that preference datasets of the same size lead to different downstream performance. A combination of Business Content Creation, Food and Cooking, and Math and Calculations, which we refer to as the Practical Life Skills dataset, consistently improves Entity Tracking, MultiRC, and WSC tasks. The Entity Tracking gains are largely related to the model's ability to handle cases where "nothing." is a valid option, raising the question of whether models' ability to abstain from selecting an answer in such cases or explicitly express uncertainty should receive more attention. Overall, we show that under strict data constraints, allocating the final 1M words to a different training objective can provide larger benefits than using them for additional pretraining.

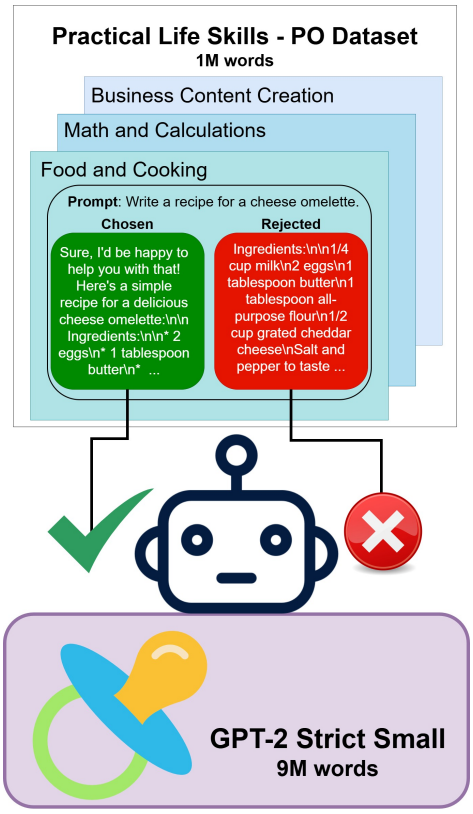


Figure 1: Training pipeline of our Practical Life Skills model. The 1M-word PLS dataset is used for preference optimization of a GPT-2 model pretrained on 9M words.

1 Introduction

Approaches to training language models are increasingly inspired by principles of human learning and cognition (Apidianaki et al., 2024). The BabyLM initiative is one example of this trend, investigating the training process of language models under developmentally motivated constraints inspired by human language acquisition (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025b). Language models are oftentimes pretrained by predicting the next token from large collections of text. While this objective has proven highly effective, it captures only one aspect of human language acquisition. Children do not learn language only by passive exposure, but also through communicative and social interactions (Tomasello, 2005; Kuhl, 2007). These interactions may accelerate the child's learning by presenting known structures in surprising context (Fazekas et al., 2020), providing them with adult reformulations in a form of recasts (Chouinard and Clark, 2003), or other social signals that indicate which behaviors are more appropriate in a given context (Nikolaus and Fourtassi, 2023). The Competition Model (Macwhinney, 1987) argues that language understanding emerges through competition between alternative linguistic cues, whose relative importance is gradually shaped by experience

Proceedings of the Second BabyLM Workshop, pages 575–584
October 29, 2026 ©2026 Association for Computational Linguistics

and communicative feedback. Young et al. (2012) show that prompting children to consider alternative explanations can lead the children to drawing stronger, more accurate inferences from the same evidence. More generally, developmental theories of learning suggest that children actively generate and evaluate alternative explanations, gradually revising their beliefs as new evidence becomes available (Gopnik, 2012; Bonawitz and Lombrozo, 2012; Walker et al., 2017).

Preference optimization introduces an analogous comparison-based learning signal for language models (Rafailov et al., 2023; Meng et al., 2024). During training, the model compares alternative responses: rather than learning from a single target output, the model is trained to assign higher probability to the preferred response and lower probability to the less preferred.

While preference optimization has shown promising results, it remains unclear which types of preference data are effective and for which specific tasks. Different preference datasets emphasize different aspects of response quality, such as factuality, helpfulness, reasoning, or creativity (Cui et al., 2024), and each preference dataset defines a different notions of what a "good answer" is. We cluster existing preference data into coherent categories and further extend them with an additional synthetic grammatical category created by us and a separate category derived from Reddit's *Explain Like I'm Five* discussions (Ethayarajh et al., 2022). We then investigate how different preference data categories affect language learning in the BabyLM-strict-small Challenge setting, where models are trained with a limited 10M-word budget.

We show that preference optimization on 1M words, despite being applied after only a small 9M-word pretraining phase, can improve model performance on some tasks. However, the choice of preference data category can have a substantial impact on the resulting capabilities. We find that consistent improvements are achieved by the *Business Content Creation*, *Food and Cooking*, and *Math and Calculations* categories, primarily on the Entity Tracking task (Kim and Schuster, 2023), and coreference resolution (WSC) and MultiRC subtasks of the SuperGLUE benchmark (Wang et al., 2019). Based on these findings, we propose a new Practical Life Skills (PLS) Preference Optimization dataset by combining these three categories and randomly sampling preference pairs to satisfy the remaining 1M-word constraint of the BabyLM Strict-Small setting (see Figure 1).

Compared to full pretraining on 10M words without preference optimization, PLS doubles the Entity Tracking score and increases WSC and MultiRC performance, while not influencing the other tasks much. Interestingly, when removing all Entity Tracking examples where the model can answer "nothing." from the available options, the improvement from PLS disappears. This suggests that a major contribution of PLS is improving the model's ability to recognize when none of the provided options applies. We argue that this capability is an important aspect of language understanding that should receive more attention in future BabyLM evaluations: models should not only select among predefined options, but also recognize when the correct response is, for instance, "there is nothing", "I do not want any of these", or "I do not know".

2 Related Work

Preference optimization has become a common approach for adapting language models after pretraining, with its usage rapidly expanding into diverse alternative directions (Park et al., 2023; Lee et al., 2024; Lanz et al., 2026). While there has been work carried out comparing the various preference optimization algorithms (Xiao et al., 2026; Meng et al., 2024), relatively little attention has been paid to the role of the preference data itself. Recent work has shown that preference tuning is sensitive to domain shift, suggesting that the domain of the preference data can affect downstream performance (Karouzos et al., 2026). Our work complements these findings by systematically studying how different preference datasets influence linguistic and reasoning abilities in data-constrained BabyLM models while keeping the optimization algorithm fixed.

Several recent BabyLM papers have explored model alignment methods. Zhao et al. (2023) applied reinforcement learning from human feedback to improve story generation, while Padovani et al. (2025) investigated dialogue-based learning together with preference optimization and reinforcement learning to improve communicative abilities. Recently, Charpentier et al. (2025a) proposed an interactive training framework combining teacher corrections with SimPO (Meng et al., 2024), and Mayer Martins et al. (2025) explored interactive feedback for storytelling.

Dataset	All Examples	Sampled Examples	Avg Prompt	Avg Response	Jaccard
Grammatical	23403	23403	0.00	21.36	0.81
r/explainlikeimfive	2484	2484	38.95	159.51	0.10
Algorithms and Programming	2152	2040	90.70	154.31	0.17
Business and Marketing	2956	1322	113.80	264.26	0.20
Business Content Creation	2683	1320	84.24	294.32	0.18
Creative Storytelling	2529	1280	99.83	290.51	0.14
Creative Writing and Arts	1895	1812	81.97	193.93	0.16
Environment Sustainability	1852	1274	85.81	306.65	0.19
Food and Cooking	1881	1571	104.02	214.23	0.20
General Knowledge Questions	3252	2654	39.85	148.54	0.18
Health and Wellbeing	2008	1396	107.29	250.68	0.19
Language Understanding Tasks	4050	2188	171.86	56.65	0.14
Math and Calculations	2413	1716	84.46	206.86	0.19
Multilingual Translation Tasks	5461	3216	93.92	61.52	0.11
Natural Language Inference	4262	1631	98.30	208.20	0.19
QA and Reasoning Tasks	2806	2163	143.84	87.15	0.16
Reading Comprehension Tasks	3388	1281	273.03	117.02	0.19
Software Engineering	7815	1609	80.72	229.83	0.19
Technology and AI Concepts	3054	1408	73.15	281.94	0.19
Text Classification Tasks	4587	2238	113.48	109.83	0.15
Writing and Essays	2091	1172	90.56	336.04	0.18

Table 1: Preference dataset statistics. *All Examples* refers to the total number of preference pairs before sampling, while *Sampled Examples* denotes the number of pairs used for training after applying the 1M-word budget. *Avg Prompt* and *Avg Response* report the average number of words in the prompt and in each response, respectively. *Jaccard* measures the word-level similarity between the chosen and rejected responses (0 = no overlap, 1 = identical).

3 Data

For pretraining, we use a randomly sampled subset of the provided `BabyLM-Strict-Small` dataset (Choshen et al., 2026). For preference optimization, we construct several preference datasets that differ in the type of preference signal they provide, and the resulting statistics are reported in Table 1.

Ultrafeedback Subsets UltraFeedback[1] is a large-scale preference dataset containing instruction-response pairs covering a broad range of topics and tasks (Cui et al., 2024). We use this dataset to investigate whether the topic of preference data influences downstream model abilities.

To create preference subsets covering individual task types and topics, we cluster examples based on their prompts. Each prompt is encoded using the `all-MiniLM-L6-v2` sentence transformer (Reimers and Gurevych, 2019) and clustered using the mini-batch k-means algorithm (Sculley, 2010). We set $k = 25$ to obtain diverse clusters while maintaining sufficient data size. After inspecting the resulting clusters, we merge underrepresented and semantically overlapping clusters with their closest counterparts. This results in 19 final preference subsets. Cluster names were assigned by GPT-5.5-mini (OpenAI, 2026) based on sampled prompts from each cluster and are used only for readability and do not affect training.

The resulting subsets differ in topic, as well as in average prompt and response length, where the prompt length ranges from 39 words on average in *General Knowledge Questions* to 273 words per prompt for *Reading Comprehension Tasks*.

Synthetic Grammatical Preference Dataset Existing preference datasets primarily focus on instruction following and helpfulness, while providing relatively little supervision targeted specifically at grammatical correctness. We therefore construct a synthetic preference dataset designed to provide an explicit grammatical preference signal.

We sample passages from BookSum (Kryściński et al., 2021) and create corrupted versions using one of three randomly selected operations: insertion, deletion, or swapping of 1–5 words. The original passage is used as the preferred response and the corrupted version as the dispreferred response. The prompt is left empty.

This corruption-based construction follows previous work showing that synthetic perturbations

[1] https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized

can improve performance on grammaticality-related benchmarks (Wang et al., 2026). While the generated errors are artificial, they provide a controlled comparison between linguistically plausible and corrupted alternatives.

The dataset contains over 23 thousand short preference pairs. Compared to other datasets, the chosen and rejected responses are highly similar, with an average Jaccard similarity of 0.81.

r/ExplainLikeImFive We additionally use a subset of the Stanford Human Preferences Dataset (Ethayarajh et al., 2022), which contains naturally occurring Reddit questions paired with human preference judgments over responses.

We select examples from the *Explain Like I'm Five* subreddit because they provide preference data for explanations intended to be simple and understandable. Unlike synthetic preference datasets, these examples reflect naturally occurring human judgments about response quality.

We keep only examples where the preferred response received at least twenty times more upvotes than the rejected response. We also remove duplicate responses appearing in multiple preference pairs. Both responses remain plausible answers to the same question, requiring the model to learn which type of explanation is preferred rather than simply distinguishing correct from incorrect answers.

4 Experimental Setup

In our experiments, we first train a small language model on a limited BabyLM corpus (9M words) and then apply preference optimization using different preference datasets (1M words, counting the chosen response, rejected response, and the prompt twice for each preference pair) described in Section 3.[2] To adhere to the BabyLM `Strict-Small` challenge rules, the total amount of training data seen by the model is limited to 10M words.

To evaluate the impact of different preference optimization data, we account for training variability by using five random seeds. Each preference optimization dataset is applied to all five independently trained base models, and the reported results are averaged across the resulting runs.

Base Model and Pretraining Following Tapaninaho (2025), we use a compact GPT-2-style decoder-only architecture (Radford et al., 2019) designed for efficient small-scale language modeling under limited data and compute budgets. Rather than adopting the larger GPT-2 configuration from the BabyLM shared-task baseline, we use this reduced configuration to enable efficient experimentation while maintaining the core GPT-2 architecture and similar performance (Tapaninaho, 2025). The model is initialized from scratch and consists of 12 Transformer layers with a hidden size of 384 and 6 attention heads.

The model is trained on a randomly sampled subset of the `BabyLM-Strict-Small` dataset containing 9M words. We train a Byte-Level BPE tokenizer from the training corpus with a vocabulary size of 16,384 tokens. Training uses the AdamW optimizer with a learning rate of 3×10^{-4}, weight decay of 0.1, cosine learning-rate decay, and 800 warmup steps. The batch size is 16, and training is performed for 10 epochs.

Preference Optimization After pretraining on the 9M words, we apply SimPO (Meng et al., 2024) as the preference optimization method. SimPO directly optimizes preference pairs without requiring a separately trained reward model, making it suitable for controlled comparisons across multiple preference datasets.

For each preference dataset, we randomly sample preference pairs without replacement until reaching 1M words. Both prompts and responses are included in this word count, although the model weights are updated only based on the response. All datasets are trained with identical SimPO hyperparameters: 10 epochs, effective batch size of 16 (batch size 4 with gradient accumulation over 4 steps), learning rate of 5×10^{-6}, and maximum sequence length of 512 tokens. The maximum prompt length is limited to 256 tokens. We use the SimPO loss with $\beta = 2.0$ and target margin $\gamma = 0.5$.

5 Results

We compare the 9M-word pretrained GPT-2 base model with the same model after SimPO training on different 1M-word preference datasets. Results are averaged across five random seeds, with standard deviations reported in the tables. Results on zero-shot benchmarks are shown in Table 2, while results on fine-tuning benchmarks are reported in Table 3.

[2] https://github.com/stepankovab/BabyLM_2026

	BLiMP	BLiMP Supplement	COMPS	ET	ET w/o nothing.	EWOK
Base (9M words)	67.57 ± 0.31	54.28 ± 1.10	51.51 ± 0.18	21.84 ± 5.21	$\mathbf{19.26 \pm 0.15}$	$\mathbf{50.79 \pm 0.87}$
Grammatical	63.12 ± 0.72	56.28 ± 1.26	50.38 ± 0.18	31.40 ± 10.54	17.22 ± 0.40	50.25 ± 0.15
r/explainlikeimfive	67.12 ± 0.48	52.11 ± 1.92	50.94 ± 0.27	29.40 ± 3.94	18.43 ± 0.69	50.00 ± 0.81
Algorithms and Programming	66.11 ± 0.39	55.15 ± 1.49	51.52 ± 0.33	34.05 ± 9.94	18.48 ± 0.37	50.36 ± 0.52
Business and Marketing	67.79 ± 0.29	54.41 ± 2.01	51.12 ± 0.22	34.14 ± 4.92	19.18 ± 0.23	50.43 ± 0.77
Business Content Creation	67.45 ± 0.41	55.38 ± 2.00	50.99 ± 0.33	$\mathbf{40.68 \pm 1.28}$	18.65 ± 0.42	50.19 ± 0.54
Creative Storytelling	66.83 ± 1.08	54.59 ± 1.12	51.01 ± 0.40	34.42 ± 8.03	18.58 ± 0.73	50.57 ± 0.32
Creative Writing and Arts	65.94 ± 0.97	53.89 ± 0.62	51.00 ± 0.50	35.62 ± 4.89	17.75 ± 0.56	50.59 ± 0.34
Environment Sustainability	65.74 ± 0.40	55.81 ± 1.70	51.47 ± 0.30	33.95 ± 7.89	18.32 ± 0.50	50.28 ± 0.73
Food and Cooking	67.07 ± 0.45	54.77 ± 1.75	51.36 ± 0.29	40.06 ± 2.88	18.28 ± 0.79	50.00 ± 0.77
General Knowledge Questions	65.51 ± 1.11	54.29 ± 1.56	51.13 ± 0.22	37.47 ± 3.28	17.76 ± 0.48	50.12 ± 0.41
Health and Wellbeing	66.96 ± 0.31	55.02 ± 1.44	51.23 ± 0.41	34.46 ± 6.54	18.57 ± 0.73	50.31 ± 0.63
Language Understanding Tasks	66.55 ± 0.35	54.82 ± 0.68	51.55 ± 0.25	36.70 ± 8.22	18.36 ± 0.26	50.16 ± 0.69
Math and Calculations	67.26 ± 0.50	54.84 ± 1.21	51.14 ± 0.30	40.45 ± 2.60	18.63 ± 0.56	50.05 ± 0.22
Multilingual Translation Tasks	66.73 ± 0.42	55.89 ± 0.76	51.36 ± 0.23	37.89 ± 4.75	19.05 ± 0.49	50.01 ± 0.95
Natural Language Inference	$\mathbf{67.88 \pm 0.51}$	55.63 ± 1.74	51.39 ± 0.30	37.95 ± 5.64	18.78 ± 0.67	50.60 ± 0.21
QA and Reasoning Tasks	66.44 ± 0.49	53.72 ± 1.00	$\mathbf{51.76 \pm 0.14}$	37.87 ± 6.20	18.12 ± 0.42	50.20 ± 0.56
Reading Comprehension Tasks	66.37 ± 0.53	54.51 ± 1.14	51.42 ± 0.25	34.19 ± 10.64	18.79 ± 0.82	50.54 ± 0.43
Software Engineering	66.83 ± 0.40	54.19 ± 1.36	51.27 ± 0.42	39.62 ± 2.75	18.58 ± 0.80	50.30 ± 0.65
Technology and AI Concepts	67.03 ± 0.29	$\mathbf{56.47 \pm 1.70}$	51.00 ± 0.40	33.99 ± 6.29	18.71 ± 0.48	50.46 ± 0.51
Text Classification Tasks	65.26 ± 0.95	53.47 ± 0.81	51.56 ± 0.34	27.92 ± 11.95	18.29 ± 0.71	50.44 ± 0.46
Writing and Essays	66.13 ± 1.29	53.92 ± 1.21	51.34 ± 0.21	28.05 ± 4.07	17.75 ± 0.60	50.20 ± 0.67

Table 2: Zero-shot benchmark performance after preference optimization with different datasets. Values represent mean performance across five runs with standard deviation.

Zero-Shot Benchmarks Most zero-shot benchmarks remain close to the pretrained baseline regardless of the preference optimization dataset.

On BLiMP (Warstadt et al., 2020), most datasets change the overall score by less than one point. The main exception is the *Grammatical* subset, which decreases the score from 67.57 to 63.12. Interestingly, the opposite trend is observed on BLiMP Supplement, where the same subset improves performance from 54.28 to 56.28. COMPS (Misra et al., 2023) and EWOK (Ivanova et al., 2025) remain nearly unchanged across all evaluated datasets.

The largest differences are observed on the Entity Tracking (ET) benchmark. The pretrained baseline achieves 21.84 points, while almost every preference optimization dataset improves this score by at least six points. The largest improvements are obtained using the *Business Content Creation*, *Food and Cooking*, and *Math and Calculations* subsets, all exceeding 40 points. We also observe larger standard deviations for lower-performing datasets, whereas the highest-scoring datasets remain consistently strong across all five runs.

The Entity Tracking benchmark contains a substantial number of instances where the option "nothing." is available among the possible answers (2703 out of 9483 instances). This allows the model to indicate that none of the provided entities applies, reducing potential hallucinations caused by being forced to select an entity when no valid option is available. To investigate whether the observed improvements are primarily due to learning this ability or instead reflect improved entity tracking among valid options, we additionally evaluate models on a filtered version of the benchmark where all instances containing the "nothing." option are removed. This results in the *ET w/o nothing* setting reported in Table 2. On this version of the benchmark, the improvements observed on the original Entity Tracking task disappear, with performance becoming comparable to or slightly below the pretrained baseline.

Fine-Tuning Benchmarks Most of the fine-tuning benchmarks also show relatively small differences across preference optimization datasets.

Performance on BoolQ, MNLI, MRPC and QQP (Wang et al., 2019) remains close to the pretrained baseline for nearly all datasets. MultiRC is similarly stable overall, although the *r/explainlikeimfive*, *Math and Calculations* and *Food and Cooking* datasets improve the average score by more than two points.

RTE performance decreases slightly for most datasets, typically by around one percentage point compared to the pretrained baseline. In contrast, WSC generally improves after preference optimization, with most datasets gaining between two and three points. The largest improvement is achieved by the *General Knowledge Questions* subset, which improves the average score by almost five points.

	BoolQ (acc)	MNLI (acc)	MRPC (f1)	MultiRC (acc)	QQP (f1)	RTE (acc)	WSC (acc)
Base (9M words)	66.92 ± 0.84	50.30 ± 0.93	83.02 ± 0.41	60.29 ± 2.40	62.25 ± 0.93	59.42 ± 1.09	62.69 ± 2.92
Grammatical	67.21 ± 0.66	50.38 ± 0.54	82.73 ± 0.19	59.44 ± 1.92	61.83 ± 0.33	57.84 ± 2.47	66.54 ± 4.21
r/explainlikeimfive	66.96 ± 0.58	$\mathbf{50.74 \pm 0.47}$	82.61 ± 0.20	62.52 ± 3.16	62.08 ± 0.71	58.56 ± 2.25	63.46 ± 1.36
Algorithms and Programming	$\mathbf{67.74 \pm 0.85}$	50.47 ± 0.59	83.06 ± 0.50	61.59 ± 3.21	62.37 ± 0.54	58.13 ± 2.56	65.38 ± 4.51
Business and Marketing	66.66 ± 0.92	50.63 ± 0.73	82.81 ± 0.61	60.49 ± 3.08	62.17 ± 1.08	56.12 ± 1.13	63.08 ± 2.50
Business Content Creation	67.47 ± 0.74	50.44 ± 0.61	82.96 ± 0.30	60.98 ± 2.27	61.96 ± 0.82	58.42 ± 3.43	64.23 ± 2.19
Creative Storytelling	67.08 ± 1.17	50.40 ± 0.37	82.92 ± 0.41	61.69 ± 2.79	62.50 ± 0.39	59.13 ± 1.56	65.00 ± 3.94
Creative Writing and Arts	67.01 ± 0.50	50.24 ± 0.60	82.72 ± 0.38	61.59 ± 2.40	62.27 ± 0.76	$\mathbf{60.29 \pm 2.12}$	65.00 ± 5.16
Environment Sustainability	66.80 ± 0.71	49.99 ± 0.57	82.87 ± 0.60	59.95 ± 2.00	61.90 ± 0.54	57.84 ± 1.30	64.23 ± 1.72
Food and Cooking	66.53 ± 0.93	50.49 ± 0.63	82.78 ± 0.46	62.69 ± 3.39	61.97 ± 0.86	57.98 ± 2.99	64.61 ± 2.58
General Knowledge Questions	66.80 ± 1.04	50.58 ± 0.16	82.59 ± 0.40	61.24 ± 3.57	62.06 ± 0.76	58.13 ± 2.18	$\mathbf{67.31 \pm 4.90}$
Health and Wellbeing	66.39 ± 0.57	50.57 ± 0.39	82.53 ± 0.40	60.85 ± 2.69	62.15 ± 0.51	58.42 ± 2.98	65.00 ± 2.85
Language Understanding Tasks	66.35 ± 1.25	50.22 ± 0.60	82.45 ± 0.22	60.44 ± 2.80	62.11 ± 0.45	59.71 ± 3.94	66.15 ± 3.22
Math and Calculations	66.71 ± 0.87	50.23 ± 0.43	$\mathbf{83.13 \pm 0.65}$	$\mathbf{62.94 \pm 2.97}$	62.20 ± 0.81	58.56 ± 0.97	65.00 ± 3.16
Multilingual Translation Tasks	66.16 ± 0.73	49.89 ± 0.37	82.16 ± 0.44	60.24 ± 3.37	61.64 ± 0.79	57.41 ± 1.71	65.00 ± 3.16
Natural Language Inference	66.79 ± 1.25	50.40 ± 0.68	82.77 ± 0.37	60.77 ± 3.06	62.20 ± 0.24	58.70 ± 1.31	64.23 ± 1.72
QA and Reasoning Tasks	66.80 ± 0.59	50.35 ± 0.23	82.63 ± 0.83	59.57 ± 2.92	62.22 ± 0.45	58.56 ± 1.31	65.77 ± 3.94
Reading Comprehension Tasks	66.57 ± 1.09	50.37 ± 0.47	82.22 ± 0.44	58.63 ± 0.23	$\mathbf{62.56 \pm 0.53}$	57.70 ± 3.15	64.23 ± 2.58
Software Engineering	65.78 ± 0.54	50.32 ± 0.21	82.67 ± 0.87	59.73 ± 2.95	62.33 ± 0.36	57.12 ± 1.49	64.23 ± 2.19
Technology and AI Concepts	67.31 ± 0.56	50.57 ± 0.57	82.94 ± 0.29	59.93 ± 2.85	62.22 ± 0.28	57.98 ± 1.49	63.46 ± 2.35
Text Classification Tasks	66.77 ± 0.81	50.10 ± 0.23	82.60 ± 0.49	60.63 ± 2.16	62.48 ± 0.42	57.84 ± 0.82	63.85 ± 2.85
Writing and Essays	66.55 ± 0.93	50.27 ± 0.65	82.73 ± 0.26	59.77 ± 1.03	62.02 ± 0.49	61.15 ± 2.55	66.15 ± 2.19

Table 3: Fine-tuning benchmark performance after preference optimization with different datasets. Values represent mean performance across five runs with standard deviation.

Benchmark	Baseline	10M	9M	PLS
BLiMP	65.22	**67.98**	67.85	66.86
BLiMP Sup.	**57.25**	55.85	53.93	55.50
COMPS	**51.81**	51.18	51.38	51.57
ET	21.07	20.69	21.87	**41.92**
ET w/o nothing.	19.10	**19.79**	19.20	18.97
EWOK	50.63	50.23	**50.71**	50.39
BoolQ	**68.26**	67.28	67.58	65.87
MNLI	50.08	**50.26**	49.74	50.12
MRPC	81.93	82.53	**82.74**	82.39
MultiRC	65.29	58.50	58.00	**65.35**
QQP	62.60	**63.16**	61.07	62.23
RTE	58.27	56.83	**59.71**	58.27
WSC	61.54	61.54	61.54	**65.38**

Table 4: Performance of GPT-2-based models across zero-shot and fine-tuning benchmarks, including the official `BabyLM-2026-Baseline-GPT2-Strict-Small` baseline, 9M- and 10M-word pretrained variants, and our PLS model.

5.1 Final Submission

Based on the benchmark results above, we selected three datasets that performed consistently well across the evaluated benchmarks: *Business Content Creation*, *Food and Cooking*, and *Math and Calculations*. We refer to this combination as the *Practical Life Skills* dataset. We trained our final submission model[3] using the PLS dataset and the same SimPO procedure.

Table 4 compares the resulting PLS model with our GPT-2 model pretrained on 9M words, the same model pretrained on the full 10M-word `BabyLM-Strict-Small` dataset, and the official `BabyLM-2026-Baseline-GPT2-Strict-Small` baseline (Choshen et al., 2026).

Overall, the three pretrained models achieve comparable performance across most benchmarks. In contrast, the PLS model doubles the Entity Tracking score on the original benchmark, while also achieving the highest scores on MultiRC and WSC, increasing WSC performance by 4 points, and MultiRC performance by 7 points compared to both the 9M and 10M-word pre-trained base model. Performance on the remaining benchmarks is almost unchanged or only slightly decreased.

6 Discussion

Small pretraining improvements The 9M-word and 10M-word pretrained models achieve comparable performance, showing that the additional 1M words used for pretraining provide only limited and inconsistent gains across downstream benchmarks, such as small improvements on some tasks (e.g., BLiMP Supplement, ET without "nothing.", MNLI, and QQP) and decreases on others (e.g., Entity Tracking, EWOK, BoolQ, and RTE). In contrast, using the same 1M-word budget for preference optimization leads to larger improvements, particularly on Entity Tracking, MultiRC, and WSC. This suggests that under strict data constraints, allocating part of the training budget to a different optimiza-

[3] https://huggingface.co/stepanb2/babylm-gpt2-simpo-pls-10m

tion objective can be more effective than using the entire budget for additional pretraining.

Dataset content matters Our results suggest that the topic of the preference optimization data matters, although its effect depends strongly on the downstream task. While most evaluated benchmarks remain stable across different preference datasets, some datasets consistently perform above or below average. In particular, the *Food and Cooking* or *Math and Calculations* datasets repeatedly rank among the strongest performers, while datasets such as *Multilingual Translation Tasks*, *Writing and Essays*, and *Text Classification Tasks* consistently achieve lower scores.

We also investigated whether the observed differences could be explained by simple dataset statistics (Table 1). We found a small number of statistically significant correlations, for example between average prompt length and COMPS performance (Spearman $\rho = 0.62$, $p = 0.003$), and between average response length and MRPC (Spearman $\rho = 0.55$, $p = 0.009$). However, these benchmarks vary by less than one point across almost all preference datasets, suggesting that these correlations do not have a substantial impact on downstream performance. Overall, these correlations are unlikely to explain the main differences observed in our experiments. In particular, we found no meaningful relationship between the examined dataset statistics and performance on Entity Tracking, MultiRC, or WSC, where the largest gains were observed.

Practical Life Skills The three datasets selected for our final submission all involve solving practical procedural tasks. Recipes require keeping track of ingredients throughout a sequence of actions, mathematical problems require tracking variables during a computation, and business communication frequently describes requests, documents, or workflows whose state changes over time. While this observation is only qualitative, it motivated our decision to combine these datasets into the *Practical Life Skills* dataset.

Entity Tracking The most striking result of our experiments is the difference between the original Entity Tracking benchmark and its filtered version without instances containing the "nothing." option. On the original benchmark, the PLS model almost doubles the Entity Tracking score compared to the pretrained baseline. However, this improvement almost completely disappears once examples whose possible option is "nothing." are removed. This suggests that the observed gains are primarily driven by the model's ability to recognize when none of the provided entities applies, reducing hallucinations caused by being forced to select one of the available options, rather than by a general improvement in entity tracking. Since the evaluation is performed in a zero-shot setting, this reflects the model's ability to recognize such cases rather than overfitting to the benchmark.

One possible explanation is that practical procedural domains naturally contain many situations where objects, ingredients, resources, or quantities become unavailable or are no longer present. Preference optimization on such examples may therefore encourage the model to predict the absence of an entity when appropriate instead of selecting one of the remaining alternatives.

More broadly, this observation highlights an aspect of language understanding that receives relatively little attention in current benchmark design: recognizing when none of the candidate answers applies. In addition to selecting the correct answer, language models should also learn to recognize situations where the correct response is "there is nothing", "none of the above", or "I do not know". We believe future BabyLM evaluations could benefit from including more tasks of this kind.

7 Conclusion

In this work, we studied how the choice of preference optimization data affects small language models in the BabyLM setting. Despite the model being pretrained on only a very limited amount of data, applying preference optimization with a 1M-word dataset led to improvements on several downstream tasks. Moreover, these improvements were larger than those obtained by using the same 1M-word budget for additional pretraining.

We found that preference datasets of the same size can lead to different performance, showing that the content of preference data plays an important role. The *Business Content Creation*, *Food and Cooking*, and *Math and Calculations* datasets consistently performed well and improved several benchmarks when used as a 1M-word preference optimization dataset. We combined these three datasets into a single 1M-word *Practical Life Skills* dataset, which improved performance on MultiRC and WSC, as well as the model's ability to handle cases where "nothing." is available as a possible

answer in Entity Tracking. This raises a question for future work: whether the ability of models to abstain from selecting an answer when none of the options applies, or to explicitly state uncertainty (e.g., "I don't know"), should receive more attention in future evaluations.

Limitations

Our study has several limitations. First, we evaluate only SimPO, leaving open whether the same observations hold for other preference optimization methods. Second, all experiments are conducted in the BabyLM Strict-Small setting using a GPT-2 architecture, and the findings may not directly transfer to larger models. An important direction for future work is investigating whether these findings scale to the Strict (100M) setting and whether post-training a larger model on the same 1M-word corpus leads to similar observations. Finally, our interpretation of the Entity Tracking results is based on a single benchmark. Evaluating the models on additional benchmarks that require recognizing the absence of a correct answer, uncertainty, or hallucinations would help determine whether the observed behavior generalizes beyond Entity Tracking.

Acknowledgments

This research was partially supported by the SVV project number 260 821.

References

Marianna Apidianaki, Abdellah Fourtassi, and Sebastian Padó. 2024. Language learning, representation, and processing in humans and machines: Introduction to the special issue. *Computational Linguistics*, 50(4):1201–1210.

Elizabeth Baraff Bonawitz and Tania Lombrozo. 2012. Occam's rattle: Children's use of simplicity and probability to constrain inference. *Developmental Psychology*, 48(4):1156–1164.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Wilcox, and Adina Williams. 2025a. Babylm turns 3: Call for papers for the 2025 babylm workshop. *Preprint*, arXiv:2502.10645.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025b. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Michelle M. Chouinard and Eve V. Clark. 2003. Adult reformulations of child errors as negative evidence. *Journal of Child Language*, 30(3):637–669.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Qiang Yue, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with high-quality feedback.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR.

Judit Fazekas, Andrew Jessop, Julian Pine, and Caroline Rowland. 2020. Do children learn from their prediction mistakes? a registered report evaluating error-based theories of language acquisition. *Royal Society Open Science*, 7(11):180877.

Alison Gopnik. 2012. Scientific thinking in young children: Theoretical advances, empirical research, and policy implications. *Science*, 337(6102):1623–1627.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Constantinos Karouzos, Xingwei Tan, and Nikolaos Aletras. 2026. An empirical study on preference tuning generalization and diversity under domain shift. *Preprint*, arXiv:2601.05882.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *Preprint*, arXiv:2105.08209.

Patricia K Kuhl. 2007. Is speech learning 'gated'by the social brain? *Developmental science*, 10(1):110–120.

Vojtěch Lanz, Yufei Cui, and Prasanna Parthasarathi. 2026. Dimpo: Dimensionality reduction for attention using preference optimization.

Janghwan Lee, Seongmin Park, Sukjin Hong, Minsoo Kim, Du-Seong Chang, and Jungwook Choi. 2024. Improving conversational abilities of quantized large language models via direct preference alignment. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11346–11364, Bangkok, Thailand. Association for Computational Linguistics.

Brian Macwhinney. 1987. *The Competition Model*, pages 249–308. Psychology Press.

Jonas Mayer Martins, Ali Hamza Bashir, Muhammad Rehan Khalid, and Lisa Beinborn. 2025. Once upon a time: Interactive learning for storytelling with small language models. In *Proceedings of the First BabyLM Workshop*, pages 454–468, Suzhou, China. Association for Computational Linguistics.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Mitja Nikolaus and Abdellah Fourtassi. 2023. Communicative feedback in language acquisition. *New Ideas in Psychology*, 68:100985.

OpenAI. 2026. Gpt-5.5-mini. `https://openai.com/`.

Francesca Padovani, Bastian Bunzeck, Manar Ali, Omar Momen, Arianna Bisazza, Hendrik Buschmeier, and Sina Zarrieß. 2025. Dialogue is not enough to make a communicative BabyLM (but neither is developmentally inspired reinforcement learning). In *Proceedings of the First BabyLM Workshop*, pages 421–435, Suzhou, China. Association for Computational Linguistics.

Ryan Park, Ryan Theisen, Navriti Sahni, Marcel Patek, Anna Cichońska, and Rayees Rahman. 2023. Preference optimization for molecular language models. *Preprint*, arXiv:2310.12304.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

D. Sculley. 2010. Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 1177–1178, New York, NY, USA. Association for Computing Machinery.

Joonas Tapaninaho. 2025. MoEP: Modular expert paths for sample-efficient language modeling. In *Proceedings of the First BabyLM Workshop*, pages 540–547, Suzhou, China. Association for Computational Linguistics.

Michael Tomasello. 2005. *Constructing a language: A usage-based theory of language acquisition*. Harvard University Press.

Caren M. Walker, Tania Lombrozo, Joseph J. Williams, Anna N. Rafferty, and Alison Gopnik. 2017. Explaining constrains causal learning in childhood. *Child development*, 88(1):229–246. Publisher Copyright: © 2016 The Authors. Child Development © 2016 Society for Research in Child Development, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Yingshan Susan Wang, Linlu Qiu, Zhaofeng Wu, Roger P. Levy, and Yoon Kim. 2026. Implicit representations of grammaticality in language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15035–15055, San Diego, California, United States. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Wenyi Xiao, Zechuan Wang, Leilei Gan, Shuai Zhao, Zongyue Li, Ruirui Lei, Wanggui He, Luu Anh Tuan, Long Chan, Hao Jiang, Zhou Zhao, and Fei Wu. 2026. A survey of direct preference optimization: Datasets, theories, variants, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20.

Andrew G. Young, Martha W. Alibali, and Charles W. Kalish. 2012. Disagreement and causal learning: Others' hypotheses affect children's evaluations of evidence. *Developmental Psychology*, pages 1242–1253.

Xingmeng Zhao, Tongnian Wang, Sheri Osborn, and Anthony Rios. 2023. BabyStories: Can reinforcement learning teach baby language models to write better stories? In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 186–197, Singapore. Association for Computational Linguistics.

Subword Segmental BabyLMs: Learning to Tokenise for Sample-Efficient Pretraining

Francois Meyer
Department of Computer Science
University of Cape Town
francois.meyer@uct.ac.za

Abstract

In the standard LM training pipeline, subword tokenisation is applied as a preprocessing step. Subword segmental language modelling is an alternative paradigm in which tokenisation is learned during training, allowing the model to discover subword units that optimise its training objective. In this paper, we present our submission to the 2026 BabyLM Challenge, for which we develop two new subword segmental LMs: SubSegGPT and SubSegDeBERTa. SubSegGPT is a decoder-only model that learns tokenisation during autoregressive pretraining. SubSegDeBERTa is an encoder-based model that jointly learns to generate and tokenise masked words. We train both for the STRICT and STRICT-SMALL tracks. Our top submission to STRICT is SubSegDeBERTa, which achieves notable gains in zero-shot evaluation. Our top submission to STRICT-SMALL is SubSegGPT, which outperforms tokenisation-based baselines. Our results show that learnable subword tokenisation can improve sample-efficiency for BabyLM pretraining. We analyse the subword learning dynamics of our models and find that tokenisation gradually converges on subword units that balance morphological alignment and fine-grained segmentation.

1 Introduction

Current language model (LM) training is developmentally implausible in many ways, one of which is its reliance on subword tokenisation. The subwords produced by standard tokenisers do not reliably align with morpheme boundaries (Batsuren et al., 2024). Moreover, the dominant paradigm of applying a fixed tokeniser during preprocessing does not mirror child language learning, as humans are not born pre-equipped with a fixed lexical vocabulary. Instead, children incrementally learn to segment speech into meaningful units during language acquisition (Jusczyk, 1999).

From a more practical, performance-driven perspective, standard NLP tokenisation poses problems for data-efficient modelling. The algorithms behind tokenisers like BPE (Sennrich et al., 2016) and ULM (Kudo, 2018) learn subword boundaries based on frequency-based objectives, with no guarantee that the resulting units are optimal for LM learnability. Given sufficient training data, neural LMs are robust to sub-optimal tokenisation. However, in small data settings, this may further compound the difficulty of learning generalisable linguistic representations.

By restricting tokenisation to preprocessing, LMs are bound to a pre-determined tokenisation scheme. Alternatively, tokenisation can be cast as a learnable component of language modelling, to be continually optimised *during* training. This is the motivation behind subword segmental modelling (Meyer and Buys, 2022), which unifies tokenisation and language modelling in an end-to-end trainable framework. Instead of determining subword boundaries before training, subword segmental modelling marginalises over all possible tokenisations of a sentence and allows the model to learn which subword units optimise its LM training objective.

In this paper, we present our submission[1] to the 2026 BabyLM Challenge, for which we develop[2] two new **Sub**word **Seg**mental LMs: SubSegGPT and SubSegDeBERTa. SubSegGPT is a decoder-only subword segmental architecture with a GPT-2 (Radford et al., 2019) backbone. It combines subword segmental modelling with more modern, GPT-style conventions in decoder LM design and implementation. SubSegDeBERTa is a novel, encoder-based architecture and the first instantiation of subword segmental modelling for masked language modelling. It jointly learns to generate and tokenise masked words during training, conditioned on bidirectional context from a DeBERTa-v2 backbone (He et al., 2021).

[1] Models: francois-meyer/babylm-2026-models
[2] Code: github.com/francois-meyer/subseg-babylms

Proceedings of the Second BabyLM Workshop, pages 585–597
October 29, 2026 ©2026 Association for Computational Linguistics

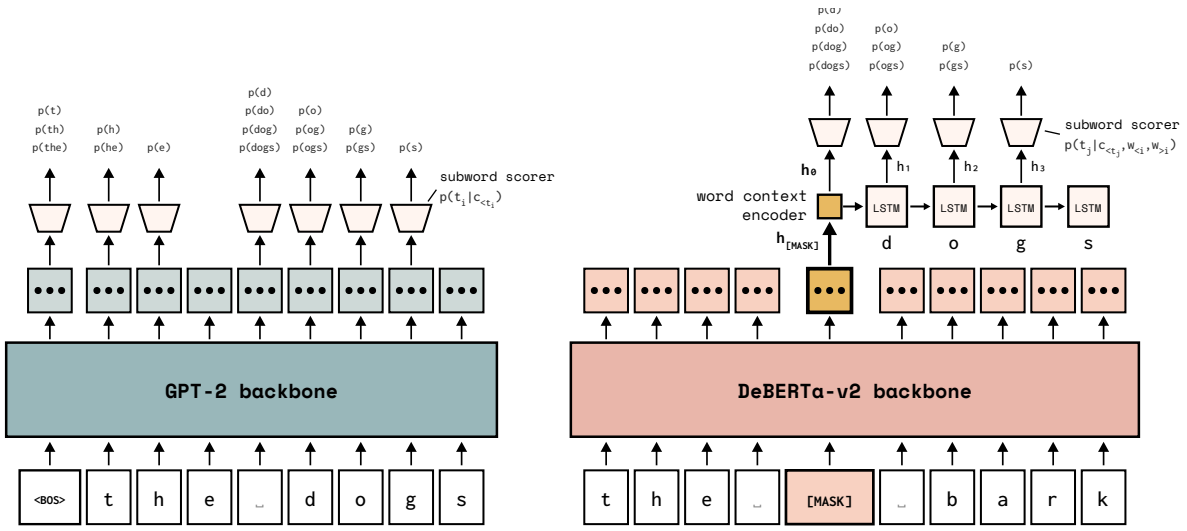


Figure 1: The architectures of SubSegGPT (left) and SubSegDeBERTa (right). Both encode character-level context, which is passed to a subword segment scorer to condition next-subword probabilities. Subword probabilities are passed to a dynamic programming algorithm that marginalises over all possible tokenisations.

We train our models for the STRICT (100M words) and STRICT-SMALL (10M words) tracks and evaluate on the official evaluation pipeline, which includes zero-shot evaluations, finetuning tasks, and human likeness tests. We compare our models to equivalent tokenisation-based BabyLM baselines, based on respectively GPT-2 and DeBERTa-v2, to isolate the effect of learning tokenisation over fixed tokenisation.

In the STRICT track, SubSegDeBERTa and SubSegGPT outperform tokenisation-based baselines on average across NLP tasks. SubSegDeBERTa is our strongest submission to STRICT, improving average zero-shot performance by 3.16 points over GPT-2. In STRICT-SMALL, SubSegGPT outperforms baselines, but SubSegDeBERTa does not. Our two models are complementary: SubSegDeBERTa is highly competitive in the STRICT track, while SubSegGPT is better suited to the more data-constrained STRICT-SMALL setting. On human likeness tasks, our models exhibit little resemblance to human language acquisition and are outperformed by tokenisation-based baselines.

Lastly, we analyse the subword learning dynamics of SubSegGPT and SubSegDeBERTa. We compare learned subword units over the course of training, tracking fertility and morpheme boundary F1. Both models undergo an initial period of rapid tokenisation change, followed by stabilised learning trajectories that converge on higher fertility (shorter subwords) and limited alignment with morpheme boundaries, which we support with a qualitative analysis of tokenised child-directed speech.

2 Background

2.1 Learning Tokenisation During Training

Without being motivated by developmental plausibility, several works have explored learning to segment language inputs during training, for tasks such as handwriting recognition (Kong et al., 2016), speech recognition (Wang et al., 2017), and translation (Kreutzer and Sokolov, 2018). These efforts equip models with the ability to discover optimal segmentation units for a given task, rather than relying on a pre-determined segmentation scheme.

Sun and Deng (2018) propose the segmental language model (SLM): an LSTM LM that computes the likelihood of a sentence by marginalising over all possible segmentations of the character sequence. Their aim is not improved performance, but rather unsupervised word segmentation for Chinese, which lacks explicit word boundaries. Kawakami et al. (2019) augment the SLM with a lexicon of high-frequency segments, improving unsupervised word segmentation. Downey et al. (2022) propose the masked SLM, a bi-directional, transformer-based SLM that outperforms recurrent SLMs on unsupervised word discovery. SLMs are trained on raw character sequences without word boundaries (whitespaces) and, as a by-product of optimising the LM training objective, they partially recover word-level segmentation.

Meyer and Buys (2022) propose the subword segmental language model (SSLM), which adapts the SLM to model subword tokenisation. SSLM assumes access to word boundaries, constrains seg-

ments to subword units (they cannot cross word boundaries), and learns subword tokenisation to optimise the LM training objective. The original SSLM (Meyer and Buys, 2022) is LSTM-based (Hochreiter and Schmidhuber, 1997). Meyer and Buys (2025) propose a transformer (Vaswani et al., 2017) SSLM, primarily as a tool to study subword learning dynamics. For both models, evaluation is limited to the Nguni languages: low-resource, agglutinative languages, for which tokenisation was hypothesised to have an outsized impact. SSLM outperformed tokenisation-based LMs on perplexity and sequence-to-sequence tasks, and performed strongly as an unsupervised morpheme segmenter.

It is unknown whether learnable subword tokenisation can improve pretraining sample-efficiency beyond the narrow linguistic scope of previous work. The BabyLM Challenge provides the ideal setting to test this. We propose two new SSLM variants, SubSegGPT and SubSegDeBERTa, which incorporate learnable subword tokenisation into modern, BabyLM-style architectures. Understanding our models requires familiarity with the subword segmental framework, so in the next subsection we provide a technical overview of SSLM.

2.2 Subword Segmental Language Modelling

SSLM (Meyer and Buys, 2022) establishes a framework for learning tokenisation *during* training. Any LM architecture can be adapted to the framework, removing its reliance on a fixed tokeniser. The key idea is that tokenisation is cast as a latent variable, inferred jointly with model parameters to optimise the LM objective. Adapting a model for subword segmental modelling requires augmenting its architecture with additional subnetworks, which we describe for our models in Section 3, and adopting the training algorithm of Meyer and Buys (2022), which we now summarise.

For a training sequence S, SSLM still minimises the standard LM loss $\mathcal{L}(\theta) = -\log p(S)$. However, whereas a vanilla LM computes $p(S)$ with the chain rule over a single pre-determined subword token sequence, SSLM computes $p(S)$ as

$$p(S) = \sum_{T \in \pi(S)} p(T), \tag{1}$$

where $\pi(S)$ is the set of all candidate tokenisations of S: every possible way that the sequence of words S can be segmented into subword units (word boundaries are enforced, so subword units cannot span across whitespaces). Each $p(T)$ is still computed with the chain rule over the token sequence T, but the overall sequence probability $p(S)$ now incorporates multiple potential tokenisations of the training example S.

Marginalising over $\pi(S)$ is intractable for long sequences, so SSLM introduces two constraints:

1. Subword segments cannot exceed a maximum character length L, which is a hyperparameter.

2. In computing the probability of a candidate tokenisation $T = \{t_1, t_2, ..., t_{|T|}\}$ with the chain rule, each next-token probability is conditioned on the *untokenised* autoregressive (preceding) character-level context, so

$$p(T) = \prod_{i=1}^{|T|} p(t_i|t_{<i}) \approx \prod_{i=1}^{|T|} p(t_i|c_{<t_i}), \tag{2}$$

where $c_{<t_i}$ is the character sequence in S that precedes t_i. This approximation discards tokenisation history, but enables tractable conditioning via a character-level context encoder.

Finally, to compute Equation 1 efficiently, Meyer and Buys (2022) use a dynamic programming algorithm that iteratively computes $p(S_{1:k})$, the marginal probability of the sequence up to each character position k, for $k = 1, ..., |S|$ (we refer the reader to Meyer and Buys (2022) for a detailed presentation of the algorithm).

3 Models

The modelling assumptions and training algorithm outlined above describe the generative model of the subword segmental framework. Parameterising this with a neural architecture requires a model capable of computing $p(t_i|c_{<t_i})$ for *any* subword token t_i and a mechanism for conditioning this probability on the character-level context $c_{<t_i}$. A vanilla LM cannot assign probabilities to arbitrary subwords, so Meyer and Buys (2022) augment the standard autoregressive architecture to enable this. Their methodology can be followed to adapt any architecture for subword segmental modelling. Doing so requires building an architecture with two components: an encoder that computes representations for a character-level context c and a subword segment scorer that computes $p(t|c)$ for any subword t.

In this section, we present SubSegGPT and SubSegDeBERTa, two new SSLMs that adapt respectively GPT-2 (Radford et al., 2019) and DeBERTa-

v2 (He et al., 2021) for learnable subword tokenisation. Their architectures are visualised in Figure 1.

3.1 SubSegGPT

SubSegGPT is a decoder-only SSLM with a GPT-2 backbone. It is a straightforward adaptation of the original, LSTM-based SSLM (Meyer and Buys, 2022) to GPT-style language modelling and makes use of the same training algorithm outlined in Section 2.2. We now describe its architecture, which closely mirrors the transformer-based SSLM of Meyer and Buys (2025), but is parameterised by the GPT-2 architecture to incorporate more recent architectural conventions and match the setup of competitive decoder-based BabyLMs.

3.1.1 Character-level history encoder

To compute $p(t_i|c_{<t_i})$ we need to encode $c_{<t_i}$, the full character sequence preceding the subword segment t_i. We encode $c_{<t_i}$ with a character-level GPT-2 backbone (excluding the language modelling head), using the final-layer output embedding $\mathbf{h}_{<t_i}$ of the last character before t_i to represent the sequence history and to condition next-subword probabilities $p(t_i|\mathbf{h}_{<t_i})$.

3.1.2 Subword segment scorer

We follow previous SSLMs in computing $p(t_i|c_{<t_i})$ as a mixture of two subword probabilities,

$$p(t_i|c_{<t_i}) = \lambda p_{\text{lex}}(t_i|c_{<t_i}) + (1-\lambda) p_{\text{char}}(t_i|c_{<t_i}), \quad (3)$$

where $\lambda \in (0,1)$ is a mixture coefficient, dynamically computed for each subword with a sigmoid-activated linear projection of $\mathbf{h}_{<t_i}$.

$p_{\text{lex}}(t_i|c_{<t_i})$ is a language modelling head that maps $\mathbf{h}_{<t_i}$ to a probability distribution over a fixed subword lexicon containing the V most frequent subwords in the training corpus (the lexicon size V is a pre-specified hyperparameter). $p_{\text{char}}(t_i|c_{<t_i})$ is a small decoder subnetwork, parameterised by a 1-layer character-level LSTM,[3] that generates subword segment t_i one character at a time and computes the subword probability as a chain rule product of individual character probabilities. It is conditioned on the sequence history by initialising the LSTM hidden state as $\mathbf{h}_{<t_i}$.

[3] SubSegGPT and SubSegDeBERTa are transformer-based LMs, parameterised by deep transformer backbones. LSTMs appear only as 1-layer subnetworks in the LM head, conditioning or generating subwords as short character sequences.

The lexicon layer p_{lex} and character decoder p_{char} play complementary roles in next-subword prediction. p_{lex} directly computes probabilities of frequent subwords, such as common morphemes and words, but cannot handle infrequent, out-of-lexicon segments. By contrast, p_{char} can assign probabilities to arbitrary subword segments by composing them character by character, covering rare and previously unseen subwords. The mixture gate λ learns to balance their contributions dynamically, based on the character-level context $\mathbf{h}_{<t_i}$.

The mixture-based subword segment scorer enables SubSegGPT to compute $p(t_i|c_{<t_i})$ for any candidate subword segment t_i at any position in a training sequence. These probabilities (top of Figure 1) are passed to the dynamic programming algorithm of Section 2.2, which efficiently computes the marginal (Equation 1) via the probabilities of all candidate tokenisations (Equation 2). SubSegGPT is trained end-to-end by minimising the negative log-likelihood, jointly optimising autoregressive language modelling and subword tokenisation.

3.2 SubSegDeBERTa

We propose SubSegDeBERTa, an encoder-based, masked SSLM with a DeBERTa-v2 backbone. Meyer and Buys (2022) introduce subword segmental modelling for decoder-only LMs. Their framework is inherently autoregressive, so its extension to masked language modelling is non-trivial. In SubSegDeBERTa, we develop the first masked SSLM, incorporating learnable subword tokenisation into encoder-based pretraining. We encode bi-directional character-level context with a DeBERTa-v2 backbone, randomly mask a subset of input words, and jointly learn to generate and tokenise masked words into subword units.

3.2.1 Character-level context encoder

We randomly mask a fixed proportion of words in each training sequence. We define words as whitespace-delimited character sequences and restrict subword segments to span within word boundaries. If a word is sampled for masking, we replace its entire character sequence with a single `[MASK]` token (as shown in Figure 1 for the masked word "dogs"). Our bi-directional encoder is a character-level DeBERTa-v2 architecture that produces final-layer output embeddings for all characters, including `[MASK]` tokens.

3.2.2 Masked word scorer

In vanilla MLMs, the final-layer representation $\mathbf{h}_{\texttt{[MASK]}}$ is used to predict the masked token. In SubSegDeBERTa, the masked word is not predicted as a single token. Instead, we generate masked words subword segmentally i.e. by marginalising over all possible subword tokenisations of a masked word.

Suppose we mask the ith word in a sentence, denoted by w_i and consisting of characters $c_1, c_2, ..., c_{|w_i|}$. During training, we maximise

$$p(w_i|w_{<i}, w_{>i}) = \sum_{T \in \pi(w_i)} p(T|w_{<i}, w_{>i}), \quad (4)$$

where $\pi(w_i)$ is the set of all candidate tokenisations of w_i, and word probabilities are conditioned on bi-directional context (all words to the left $w_{<i}$ and right $w_{>i}$ of the target word). The probability of each tokenisation $T = \{t_1, t_2, ..., t_{|T|}\}$ is computed with the chain rule as

$$p(T|w_{<i}, w_{>i}) = \prod_{j=1}^{|T|} p(t_j|c_{<t_j}, w_{<i}, w_{>i}), \quad (5)$$

so this component of SubSegDeBERTa is autoregressive: masked word generation is conditioned on bi-directional context beyond the word $(w_{<i}, w_{>i})$, but within the word it is conditioned on left-to-right context ($c_{<t_j}$ is the character sequence preceding subword segment t_j within word w_i).

To compute the marginal of Equation 4 efficiently, we use the same dynamic programming algorithm discussed in Section 2.2 and introduce the same simplifying assumptions: we limit the length of t_j to a pre-specified maximum number of characters and condition on untokenised character-level context, although here the context is bi-directional (Section 3.2.1). To compute the subword probability $p(t_j|c_{<t_j}, w_{<i}, w_{>i})$, we use the same mixture model as SubSegGPT (Section 3.1.2). However, the masked LM setting introduces a complication that requires additional subnetworks.

The GPT-2 backbone of SubSegGPT outputs contextual representations for each character in an input sequence. These representations are passed to the subword segment scorer and are used to condition next-subword probabilities at any position in the character sequence (including subword segments that start mid-word, such as "gs" in "dogs", on the left of Figure 1). The DeBERTa backbone of SubSegDeBERTa outputs contextual representations for all characters in a sequence *except masked words*, whose characters are replaced by a single `[MASK]` token. $\mathbf{h}_{\texttt{[MASK]}}$ encodes the bidirectional context $(w_{<i}, w_{>i})$, but provides only a single representation per masked word. To score subword segments at each position within w_i, we additionally need per-position representations that encode the autoregressive within-word history $c_{<t_j}$.

To address this, we introduce a *word context encoder* to encode the context at every character in word w_i (its role is visualised in Figure 1). This is a 1-layer character-level LSTM that processes the target word's characters $c_1, c_2, ..., c_{|w_i|}$ left-to-right, conditioned on $\mathbf{h}_{\texttt{[MASK]}}$ in two ways: we initialise its hidden state as a learned projection of $\mathbf{h}_{\texttt{[MASK]}}$ and concatenate $\mathbf{h}_{\texttt{[MASK]}}$ to input character embeddings at every step. This produces a sequence of per-position context representations $\mathbf{h}_k = \text{LSTM}(c_k, \mathbf{h}_{\texttt{[MASK]}})$, that encode the bidirectional context $(w_{<i}, w_{>i})$ via $\mathbf{h}_{\texttt{[MASK]}}$ and the within-word history $(c_{<t_j})$ via LSTM recurrence. This provides a contextual representation $\mathbf{h}_k$ for every character position in w_i, which is used to condition $p(t_j|c_{<t_j}, w_{<i}, w_{>i})$ for any subword segment t_j in w_i, as shown for all possible subwords in the word "dogs", on the right of Figure 1.

To compute these probabilities, we use the same mixture model as the SubSegGPT subword segment scorer (Equation 3), with $\mathbf{h}_k$ passed as conditioning context, instead of $\mathbf{h}_{<t_i}$. The dynamic programming algorithm of Section 2.2 computes the masked word marginal (Equation 4) via the probabilities of all candidate tokenisations (Equation 5). SubSegDeBERTa is trained end-to-end by minimising the negative log-likelihood of masked words, jointly optimising masked language modelling and subword tokenisation.

4 Experimental Setup

4.1 Pretraining

We follow the guidelines of the 2026 BabyLM Challenge (Choshen et al., 2026) to pretrain SubSegGPT and SubSegDeBERTa for STRICT and STRICT-SMALL. We use the text-only datasets released by the organisers and pretrain for 10 epochs.

To test the impact of learnable tokenisation, we compare our models to fixed-tokenisation BabyLMs based on the corresponding backbone architectures and trained on the same data. For SubSegGPT, we use the GPT-2 (Radford et al., 2019) baselines released by the BabyLM Challenge organisers. For SubSegDeBERTa, we pretrain our own

Track	Model	BLiMP	BLiMP Sup.	EWoK	Entity	COMPS	PIQA	Avg.
	Random chance	50.00	50.00	50.00	20.00	50.00	37.50	42.92
STRICT	GPT-2	74.73	65.00	54.37	16.91	55.85	36.62	50.58
	SubSegGPT	**78.22**	64.78	51.72	16.12	53.75	**40.15**	50.79
	DeBERTa	73.95	64.20	52.35	17.46	53.13	35.30	49.40
	SubSegDeBERTa	**78.22**	**68.73**	**55.39**	**22.28**	**57.77**	40.02	**53.74**
STRICT-SMALL	GPT-2	65.23	57.25	50.63	19.1	**51.81**	35.09	46.52
	SubSegGPT	**68.80**	**62.77**	49.76	**20.48**	50.51	30.67	**47.17**
	DeBERTa	63.78	59.94	50.28	19.67	50.88	**35.28**	46.64
	SubSegDeBERTa	64.68	60.56	**50.99**	17.89	51.28	33.74	46.52

Table 1: Results for zero-shot tasks in the BabyLM evaluation pipeline. Best result per track is **boldfaced**.

DeBERTa-v2 (He et al., 2021) models as baselines, matching the backbone architectural configurations of SubSegDeBERTa for comparability. Table 4 in the appendix details our baseline setup.

4.2 Hyperparameters

Our backbone architectures match the size of the BabyLM GPT-2 baseline, which corresponds to BASE configurations of GPT-2 and DeBERTa. We tune pretraining hyperparameters as detailed in Appendix A. Our backbone architectures are character-based, so their embedding matrices contribute negligibly to overall parameter count. However, our subnetworks for subword segment scoring introduce additional parameters. The model sizes and hyperparameter settings of our submissions and baselines are reported in Tables 4 and 5 in the appendix.

4.3 Evaluation

We evaluate on the official 2026 BabyLM Challenge evaluation pipeline, which includes three types of tasks (full list in Appendix B). **(1) Zero-shot tasks** test linguistic knowledge directly from model probabilities via minimal pair evaluations. **(2) Finetuning tasks** test natural language understanding on (Super)GLUE (Wang et al., 2018, 2019) with task-specific finetuning. **(3) Human-likeness tasks** evaluate how well model predictions align with human psycholinguistic data (we report these results in Appendix D, as neither our models nor baselines perform well on these tasks).

4.4 Evaluating Subword Segmental LMs

The BabyLM evaluation pipeline is designed for standard LM architectures. To evaluate our models on finetuning tasks, we can apply the pipeline without change (final-layer representations are passed to classification heads). However, for zero-shot tasks, the pipeline expects per-position logits, which our models do not emit: SubSegGPT outputs a per-sentence marginal (Equation 1) and SubSegDeBERTa outputs a per-word marginal (Equation 4). We implement model-specific wrappers that transform our outputs into the quantities required for each task. We leave the official pipeline unchanged, except for one line of code (see Appendix C.1 for details). Appendix C describes the wrappers we implement to evaluate our models.

5 Results

Table 1 reports zero-shot results. In the STRICT track, SubSegDeBERTa achieves the highest scores across most tasks, comfortably outperforming the strongest baseline, GPT-2. SubSegGPT also outperforms both tokenisation-based baselines, suggesting that at this scale (10 epochs over 100M words) learning tokenisation during pretraining reliably improves sample-efficiency of intrinsic linguistic knowledge acquisition.

In the STRICT-SMALL track, the relative performances of our two models are reversed. SubSegGPT outperforms SubSegDeBERTa and, on average, outperforms both baselines. Performance is more mixed across individual tasks – largely because, for most tasks, all models fail to reach above-chance performance. On the only two tasks where models reliably outperform chance, SubSegGPT achieves large performance gains over its tokenisation-based equivalent, GPT-2 (+3.57 for BLiMP and +5.52 for BLiMP Supplement). SubSegDeBERTa does not outperform DeBERTa in the STRICT-SMALL track, but reaches similar performance levels on average.

Overall, these results suggest that the benefits of learnable tokenisation are scale-dependent. In the extremely low-resource setting, where even above-chance zero-shot performance is challenging, SubSegGPT already offers gains. SubSegDeBERTa requires a larger data scale for its sample-efficiency to

Track	Model	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg.
	Majority class	64.04	35.70	68.14	57.55	62.78	53.96	61.54	57.67
STRICT	GPT-2	69.66	60.76	85.34	65.92	71.56	57.55	63.46	67.75
	SubSegGPT	64.59	58.21	84.51	66.38	72.34	60.43	63.46	67.13
	DeBERTa	**72.66**	**62.59**	84.80	**68.15**	**78.64**	**66.19**	63.46	**70.93**
	SubSegDeBERTa	71.44	60.80	**90.54**	67.16	75.16	64.03	63.46	70.37
STRICT-SMALL	GPT-2	67.71	49.84	81.37	65.76	61.67	56.83	63.46	63.81
	SubSegGPT	64.83	**54.65**	**85.71**	65.64	**71.23**	**60.43**	63.46	**66.56**
	DeBERTa	67.52	47.29	70.59	**67.53**	70.17	56.83	61.54	63.07
	SubSegDeBERTa	**68.93**	45.78	81.93	65.80	62.94	56.83	**65.38**	63.94

Table 2: Results for finetuning tasks in the BabyLM evaluation pipeline. Best result per track is **boldfaced**.

take effect, but under those conditions it produces even more reliable gains than SubSegGPT. The relative underperformance of SubSegDeBERTa in the STRICT-SMALL track might be due to its MLM objective, which provides a sparser training signal than autoregressive modelling. SubSegDeBERTa is only trained to generate a proportion of words in each sequence, whereas SubSegGPT is trained to generate every word in the corpus.

Table 2 reports results for text classification and entailment tasks from the (Super)GLUE benchmark. With task-specific finetuning, the benefits of subword segmental pretraining diminish. In the STRICT track, neither of our models consistently outperform their baselines. Among all the models we tested, DeBERTa achieves the highest average performance. At the scale of 100M words, conventional encoder-only pretraining with fixed subword tokenisation, combined with downstream finetuning, is sufficient for natural language understanding tasks. In the STRICT-SMALL track, SubSegGPT again achieves the best performance overall, as it did in zero-shot evaluation. This supports our claim that SubSegGPT provides better sample-efficiency when pretraining data is severely limited, and that this pretraining sample-efficiency transfers to improved downstream finetuning.

6 Analysing Subword Learning

In SubSegGPT and SubSegDeBERTa, we can study subword tokenisation as a learnable component of language modelling: equipped with the ability to learn tokenisation, how do subword boundaries evolve over pretraining and what are the linguistic properties of the final subword units?

6.1 Subword Learning Dynamics

We compare tokenisations of the 2022 SIGMORPHON English test set (Batsuren et al., 2022) across regular interval checkpoints (every 1M words until 10M words, every 10M words until 100M words). To extract the learned tokenisation of a sentence, we use the Viterbi algorithm to extract the highest-probability tokenisation (replacing the sum in Equations 1 and 4 with an argmax).

For each checkpoint, we quantify the properties of its subwords with three metrics. (1) Boundary flip rate measures the rate of change in tokenisation as the fraction of possible subword boundaries that change from one checkpoint to the next. (2) Fertility (Ács, 2019) is the average number of subwords per word, reflecting the granularity of subword tokenisation. (3) Morpheme boundary identification F1 measures the overlap between learned subword boundaries and ground truth morpheme boundaries, as annotated in the SIGMORPHON dataset.

Figure 2 plots the learning dynamics of SubSegGPT and SubSegDeBERTa during STRICT-SMALL pretraining. The two models exhibit similar learning trajectories. After an initial period of rapid changes, subword learning gradually stabilises and converges on a settled tokenisation scheme. This convergence is characterised by a steady increase in fertility: words are tokenised into more subword units. As a comparison, the fertility of the BabyLM baseline tokeniser (16k-vocabulary BPE) on this dataset is 1.23, so our models learn more aggressive segmentation. The subword boundaries of both models shift towards greater alignment with morphological boundaries. This alignment remains weak compared to a dedicated unsupervised morphological segmenter like Morfessor (Smit et al., 2014), which achieves 42.9% F1 on the same evaluation set, but is well above random boundary insertion at the same fertility as our models, which would achieve around 11.6% F1.

6.2 Qualitative Analysis

The subword units learned by SubSegGPT and SubSegDeBERTa reflect the tokenisation demands of

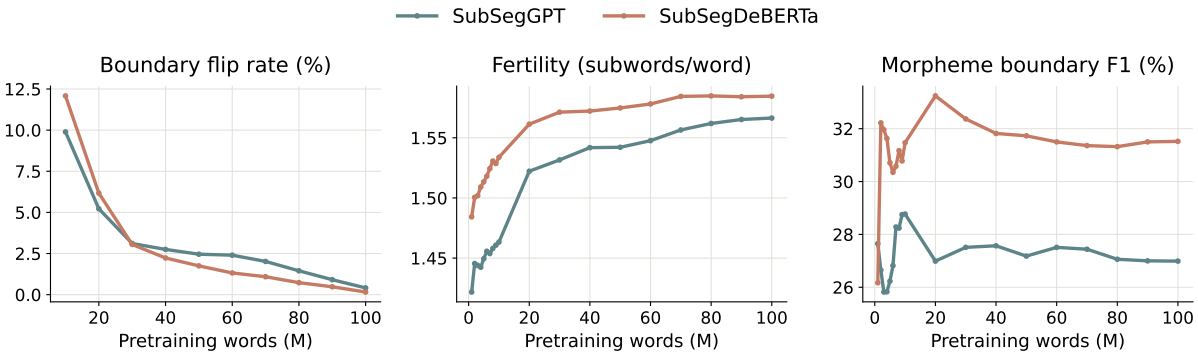


Figure 2: Subword learning dynamics of SubSegGPT and SubSegDeBERTa during STRICT-SMALL pretraining.

	SubSegGPT	SubSegDeBERTa
1M	yes he ' s playi ng in the sand box .	yes he ' s play ing in the s and box .
10M	yes he ' s playi ng in the sand box .	yes he ' s play ing in the sand box .
100M	yes he ' s playi ng in the sand box .	yes he ' s play ing in the sand box .
1M	six littl e ta xis stand ing in a row .	six l ittle ta xis stand ing in a row .
10M	six littl e ta xi s stand ing in a row .	six littl e t ax i s stand ing in a row .
100M	six littl e ta xi s s tandi ng in a row .	six littl e ta xi s stand ing in a row .
1M	you ' re jumpi ng in Mommy ' s kitch en ?	you ' re jump ing in Mommy ' s kitch en ?
10M	you ' re jumpi ng in Mommy ' s ki tchen ?	you ' re jump ing in Mommy ' s kitch en ?
100M	you ' re jumpi ng in Mommy ' s ki tchen ?	you ' re jump ing in Mommy ' s ki tchen ?

Table 3: Learned subword tokenisations of CHILDES utterances across STRICT-SMALL pretraining checkpoints, showing how subword boundaries evolve and highlighting subwords that correspond to morphemes.

sample-efficient language modelling. Our quantitative analysis suggests that this balances linguistic plausibility (partial morphological alignment) against frequency-based criteria (finer-grained segmentation). To study this trade-off qualitatively, Table 3 presents CHILDES (MacWhinney, 2000) utterances tokenised by our models, highlighting subwords corresponding to morphemes.

The examples show several instances of our models discovering morphemes as subword units, such as suffixes ("–ing", "–s") and compound constituents ("sand–box"). They also show instances of morphologically unsound tokenisation, some of which are model-specific: SubSegGPT performs worse as a morphological segmenter (see Figure 2) and this is reflected in examples like "playi–ng" and "jumpi–ng". Other morphological violations can be attributed to biases introduced by our architecture design (e.g. our mixture model for subword generation) and constraints imposed by modelling assumptions. For example, subword segments cannot exceed 5 characters, so words like "kitchen" and "little" cannot be left untokenised. Removing these constraints would allow us to study truly unrestricted subword learning, but marginalising over unbounded segment lengths is computationally infeasible in our current setup.

7 Conclusion

We present two new models that learn subword tokenisation during training to optimise their pretraining objectives: SubSegGPT for autoregressive language modelling and SubSegDeBERTa for masked language modelling. Our models excel at different scales: SubSegDeBERTa is our most competitive submission in the STRICT track and SubSegGPT performs strongly in the STRICT-SMALL track. By combining the SSLM framework with current best practices in sample-efficient pretraining, we show that SSLMs are effective beyond the low-resource, agglutinative languages for which they were originally proposed. More generally, our findings suggest reconsidering tokenisation conventions and identify end-to-end subword modelling as a promising future direction for BabyLM research.

8 Limitations

A disadvantage of subword segmental modelling is the additional computational complexity introduced by its training algorithm. Marginalising over several tokenisations requires more computations than using a single tokenisation, so SubSegGPT and SubSegDeBERTa have much longer training times than fixed-tokenisation BabyLMs (A100 GPU hours are listed in Table 4 in the appendix). For example, training SubSegDeBERTa required approximately 5× the A100 GPU hours of DeBERTa. Our aim is sample-efficiency (better performance with the same pretraining data budget), but this comes at the cost of compute-efficiency (more training FLOPs and longer training times).

References

Khuyagbaatar Batsuren, Gábor Bella, Aryaman Arora, Viktor Martinovic, Kyle Gorman, Zdeněk Žabokrtský, Amarsanaa Ganbold, Šárka Dohnalová, Magda Ševčíková, Kateřina Pelegrinová, Fausto Giunchiglia, Ryan Cotterell, and Ekaterina Vylomova. 2022. The SIGMORPHON 2022 shared task on morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 103–116, Seattle, Washington. Association for Computational Linguistics.

Khuyagbaatar Batsuren, Ekaterina Vylomova, Verna Dankers, Tsetsuukhei Delgerbaatar, Omri Uzan, Yuval Pinter, and Gábor Bella. 2024. Evaluating subword tokenization: Alien subword composition and oov generalization challenge. *Preprint*, arXiv:2404.13292.

Tyler A. Chang, Catherine Arnett, et al. 2026. Global piqa: Evaluating commonsense reasoning across 100+ languages and cultures. *Preprint*, arXiv:2510.24081.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 English sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*, 56:5190–5213.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

C.m. Downey, Fei Xia, Gina-Anne Levow, and Shane Steinert-Threlkeld. 2022. A masked segmental language model for unsupervised natural language segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 39–50, Seattle, Washington. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Peter W. Jusczyk. 1999. How infants begin to extract words from speech. *Trends in Cognitive Sciences*, 3(9):323–328.

Kazuya Kawakami, Chris Dyer, and Phil Blunsom. 2019. Learning to discover, ground and use words with segmental neural language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6441, Florence, Italy. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Lingpeng Kong, Chris Dyer, and Noah A. Smith. 2016. Segmental recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Julia Kreutzer and Artem Sokolov. 2018. Learning to segment inputs for NMT favors character-level processing. In *Proceedings of the 15th International Conference on Spoken Language Translation*, pages 166–172, Brussels. International Conference on Spoken Language Translation.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, 3 edition. Lawrence Erlbaum Associates, Mahwah, NJ.

Francois Meyer and Jan Buys. 2022. Subword segmental language modelling for nguni languages. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6636–6649, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Francois Meyer and Jan Buys. 2025. The learning dynamics of subword segmentation for morphologically diverse languages. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 647–661, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

David Samuel. 2024. Berts are generative in-context learners. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24, Gothenburg, Sweden. Association for Computational Linguistics.

Zhiqing Sun and Zhi-Hong Deng. 2018. Unsupervised neural word segmentation for Chinese via segmental language modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4915–4920, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. *SuperGLUE: a stickier benchmark for general-purpose language understanding systems*. Curran Associates Inc., Red Hook, NY, USA.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018.

GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Chong Wang, Yining Wang, Po-Sen Huang, Abdelrahman Mohamed, Dengyong Zhou, and Li Deng. 2017. Sequence modeling via segmentations. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 3674–3683. JMLR.org.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Judit Ács. 2019. Exploring BERT's vocabulary.

A Hyperparameters

During development, we tuned pretraining hyperparameters by comparing performance on the FAST evaluation pipeline, which samples subsets of the zero-shot BabyLM evaluation datasets. Among standard pretraining hyperparameters, we tuned the learning rate, batch size, warmup ratio, and masking ratio (for SubSegDeBERTa). Among hyperparameters unique to subword segmental modelling, we tuned the maximum subword segment length and the subword lexicon size (the number of top-frequency character n-grams to include in the lexicon subword scorer).

We did not perform a full grid search. Instead, we started with the BabyLM baseline hyperparameters as our default setup and varied one hyperparameter at a time. This revealed that changing certain hyperparameters (batch size, warmup ratio, and maximum segment length beyond 5 characters) had little effect on performance, while others (learning rate, masking ratio, and subword lexicon size) were influential. We subsequently experimented with different learning rates (1e-3, 1e-4, 5e-4), masking ratios (0.15, 0.3, 0.4, 0.5), and subword lexicon sizes (5k, 10k, 20k, 40k), conducting most experiments in the STRICT-SMALL setting due to its faster pretraining iterations, with more limited experimentation in the STRICT setting. The hyperparameters of our final submissions are reported in Tables 4 and 5.

B Evaluation Tasks

The official 2026 BabyLM Challenge evaluation pipeline contains three types of tasks for the STRICT and STRICT-SMALL tracks.

1. **Zero-shot:** BLiMP (Warstadt et al., 2020), BLiMP Supplement, EWoK (Ivanova et al., 2025), COMPS (Misra et al., 2023), Entity Tracking (Kim and Schuster, 2023), and the English subset of Global PIQA (Chang et al., 2026) (a hidden task announced shortly before the deadline).

2. **Finetuning:** BoolQ (Clark et al., 2019), MultiRC (Khashabi et al., 2018), RTE (Giampiccolo et al., 2007), WSC (Levesque et al., 2011), MRPC (Dolan and Brockett, 2005), QQP, and MNLI (Williams et al., 2018).

3. **Human likeness:** Reading correlations (de Varda et al., 2024) are computed based on self-paced reading times and eye tracking data. Age-of-acquisition scores (Chang and Bergen, 2022) are computed by tracking word surprisal across pretraining checkpoints and comparing learning curves to child vocabulary acquisition data.

C Evaluation Wrappers

Because our models do not output per-position logits over a fixed subword vocabulary, they cannot be evaluated directly with the official BabyLM zero-shot and human likeness evaluation pipelines. Here we describe the wrappers we implement to transform the outputs of SubSegGPT and SubSegDeBERTa (marginal probabilities) into the quantities required by each evaluation task.

C.1 SubSegGPT

Zero-shot tasks compare log-probabilities of minimal pair sentences. The evaluation pipeline computes this by summing per-position log-probabilities over the tokens in a sentence. SubSegGPT computes the log-probability of a full sentence, which we transform to per-character log-probability estimates by dividing the sentence

Hyperparameter	GPT-2	SubSegGPT	DeBERTa-v2	SubSegDeBERTa
	STRICT / SMALL	STRICT / SMALL	STRICT / SMALL	STRICT / SMALL
A100 training	–	85h / 16h	17h / 2h	80h / 9h
Parameters	98.4M	95.8 / 103.1M	113.2M	140.4 / 116.9M
Layers	12	12	12	12
Hidden size	768	768	768	768
FF size	3,072	3,072	3,072	3,072
Attention heads	12	12	12	12
Dropout	0.1	0.1	0.1	0.1
Vocabulary size	16,384	–	16,384	–
Sequence length	512	1,024	512	1,024
Learning rate	5e-5	5e-4	5e-5	5e-4
LR scheduler	cosine	cosine	cosine	cosine
Warmup ratio	0.01	0.1	0.01	0.1
Weight decay	0	0.01	0	0.01
Gradient clipping	1.0	1.0	1.0	1.0
Masking ratio	–	–	0.3	0.3 / 0.4
Batch size	16	16	16	16

Table 4: Backbone architecture configurations and training hyperparameters for our submissions and baselines.

	SubSegGPT		SubSegDeBERTa	
	STRICT	SMALL	STRICT	SMALL
Char vocab	742	416	744	418
Lexicon size	10k	20k	40k	10k
Max segment	5	5	5	5
Char decoder LSTM				
– hidden size	256	256	256	256
– layers	1	1	1	1
– embedding	128	128	128	128
Word context encoder LSTM				
– hidden size	–	–	768	768
– layers	–	–	1	1

Table 5: Configurations for subnetworks unique to subword segmental LMs.

log-probability by the number of characters in a sentence. This ensures that the per-position log-probabilities summed by the evaluation pipeline add up to the true sentence log-probability, enabling valid minimal pair comparisons. To force the evaluation pipeline to compare full sentence log-probabilities, rather than only the log-probabilities of the differing spans between minimal pairs, we disable sentence masking in the official pipeline.[4]

Human-likeness tasks require word-level surprisal, $-\log p(w_i \mid w_{<i})$ which vanilla LMs compute by summing subword token log-probabilities. We instead derive this as

$$\log p(w_i \mid w_{<i}) = \log p(w_{\leq i}) - \log p(w_{<i}), \quad (6)$$

where both terms on the right are computed as full sequence marginals (Equation 1) using our dynamic programming algorithm.

C.2 SubSegDeBERTa

For MLMs, the BabyLM pipeline scores sentences with pseudo-log-likelihood (Salazar et al., 2020), masking each subword token in turn and summing the log-probabilities over tokens in a sentence. SubSegDeBERTa lends itself naturally to this type of evaluation, as it computes the probability of a masked word (Equation 4), which can be used for cloze-style scoring. For zero-shot tasks we mask each word in turn and sum the marginal of each word to compute the pseudo-log-likelihood of a sentence. For human-likeness tasks, we mirror the BabyLM pipeline, which estimates word surprisal $-\log p(w_i \mid w_{<i})$ in MLMs by masking the target word at the end of its context, so the model conditions only on preceding words. For reading time evaluation, we match the official evaluation pipeline by applying multi-mask ending (Samuel, 2024): appending three trailing `[MASK]` tokens to obtain a less restricted sentence continuation prediction.

D Human-Likeness Results

Neither of our models exhibits strong correlations with human psycholinguistic data, as shown in Table 7. **Read** scores are positive, so incorporating

[4]To evaluate SubSegGPT on zero-shot tasks, disable phrase masking by changing the following line to `phrase_mask = [0 for _ in range(len(tokens))]`: https://github.com/babylm-org/babylm-eval/blob/68cdd160f34826307e650c484904c274692e82ce/strict/evaluation_pipeline/sentence_zero_shot/dataset.py#L89.

	SubSegGPT	SubSegDeBERTa
1M	there aren ' t any gra ham crack ers sweet ie .	there aren ' t any gra ham crack ers sweet ie .
10M	there aren ' t any g raham crack ers sweet ie .	there aren ' t any gra ham crack ers sweet ie .
50M	there aren ' t any g raham crack ers sweet ie .	there aren ' t any gra ham crack er s sweet ie .
100M	there aren ' t any g raham crack ers sweet ie .	there aren ' t any gra ham crack er s s weet ie .
1M	they ' re runni ng down the stair s .	they ' re runni ng down the stair s .
10M	they ' re runni ng down the stair s .	they ' re runn ing down the stair s .
50M	they ' re runni ng down the stair s .	they ' re runn ing down the s tairs .
100M	they ' re runni ng down the stair s .	they ' re runn ing down the s tairs .
1M	I ' m putti ng icing on them .	I ' m putti ng ic ing on them .
10M	I ' m putti ng icing on them .	I ' m put ting ic ing on them .
50M	I ' m putti ng icing on them .	I ' m put ting i cing on them .
100M	I ' m putti ng icing on them .	I ' m put ting i cing on them .
1M	did you find the littl e red bicyc le ?	did you find the l ittle red bi cycle ?
10M	did you find the littl e red b icycl e ?	did you find the littl e red bi cycle ?
50M	did you find the littl e red b icycl e ?	did you find the littl e red bi cycle ?
100M	did you find the littl e red b icycl e ?	did you find the littl e red bi cycle ?
1M	I ' m dec orati ng the cooki es .	I ' m d ecor ating the cooki es .
10M	I ' m de cora ting the c ookie s .	I ' m dec orat ing the cooki es .
50M	I ' m de cora ting the c ookie s .	I ' m d ecor ating the c ookie s .
100M	I ' m dec orati ng the c ookie s .	I ' m d ecor ating the c ookie s .

Table 6: Learned subword tokenisations of CHILDES utterances across STRICT-SMALL pretraining checkpoints, showing how subword boundaries evolve and highlighting subwords that correspond to morphemes.

Track	Model	Read	AoA
STRICT	GPT-2	**6.93**	-11.58
	SubSegGPT	1.56	0.00
	DeBERTa	4.76	0.00
	SubSegDeBERTa	2.47	-20.47
STRICT-SMALL	GPT-2	**5.63**	-12.15
	SubSegGPT	2.21	0.00
	DeBERTa	4.17	0.00
	SubSegDeBERTa	2.22	0.00

Table 7: Results for human likeness tasks. **Read** reports by how much (%) model word surprisal improves human reading time prediction. **AoA** reports correlation between model and child word acquisition.

word-level surprisal from our models does slightly improve reading time prediction regression, but baseline surprisals lead to greater improvements than SubSegGPT and SubSegDeBERTa. **AoA** scores are zero or negative for all the models we tested, which shows that model word acquisition patterns exhibit no correlation with child acquisition data.

Optimizing Multilingual BabyLM: The Roles of Tokenization, Data Packing, and Curriculum Learning

Fuming Yang[1] **Hsuan Lien**[1] **Kai-Hao Yang**[1]
Lukas Edman[1,2] **Alexander Fraser**[1,2,3]
[1]School of Computation, Information and Technology, TU Munich
[2]Munich Center for Machine Learning
[3]Munich Data Science Institute

go93pub@mytum.de, go46lic@mytum.de, go82qeq@mytum.de, lukas.edman@tum.de

Abstract

Training multilingual language models under strict data constraints remains a challenging problem. This paper presents our submission to the BabyLM 2026 Multilingual Track, focusing on English, Dutch, and Chinese. Three key challenges arise under the BabyLM constraints: efficiently sharing limited model capacity, maximizing sequence utilization, and allocating parameters efficiently under a fixed vocabulary budget. To address these challenges, we propose a three-stage curriculum learning strategy that dynamically adjusts language mixing ratios throughout training. Furthermore, we investigate data packing strategies, comparing naive chunking with document wrapping to maximize the learning signal within a strict 100M-token budget. Finally, we design a custom BPE tokenizer with an asymmetric sampling strategy. By reducing the size of the embedding matrix, the saved parameters are reallocated to the Transformer backbone, enabling highly efficient GPT-2 and BERT models. Experimental results show that combining our curriculum strategy with document packing consistently improves GPT-2 training convergence and achieves highly competitive multilingual performance.

1 Introduction

The BabyLM 2026 Multilingual Track (Choshen et al., 2026b) challenges participants to train multilingual language models under a strict data budget. Unlike conventional large-scale language models trained on billions of tokens, the BabyLM Challenge restricts the training corpus to encourage data-efficient language modeling; model size itself is not formally restricted, but participants typically keep models compact, since prior BabyLM findings suggest that scaling parameters further yields limited benefit under such constrained data budgets. The goal is to investigate how effectively language models can learn under these resource limitations.

1.1 Challenge

When training a multilingual language model under strict data constraints, we consider three main problems.

First, a significant structural bottleneck arises from the tokenizer's vocabulary size. A multilingual tokenizer must support subwords in the Latin alphabet for English and Dutch while simultaneously covering thousands of unique Chinese characters. This necessitates a massive vocabulary, which in turn requires a highly parameterized embedding matrix. If the overall model is kept compact for training efficiency, a larger vocabulary allocates a greater proportion of the model parameters to the embedding layer. Consequently, fewer parameters remain available for the Transformer layers, potentially limiting the model's representation capacity.

Second, efficiently utilizing the limited data budget during sequence formatting is highly challenging. When mixing documents of varying lengths, naive truncation or padding wastes the strict 100M-token budget on useless tokens or splits semantic contexts. Efficient data packing strategies are required to maximize the learning signal extracted from the available context window, ensuring the model shares its limited capacity effectively across languages with diverse linguistic characteristics.

Third, determining the optimal data mixing strategy is highly complex due to inherent language differences. English and Dutch share the Latin alphabet but differ in syntactic structures, while Chinese uses a character-based writing system with fundamentally different linguistic properties. As a result, these languages exhibit different learning behaviors during training. Determining how to dynamically adjust the allocation of the limited training budget across languages over time—rather than using a static mix—is therefore a key challenge.

Proceedings of the Second BabyLM Workshop, pages 598–610
October 29, 2026 ©2026 Association for Computational Linguistics

1.2 Our Contribution

To solve the challenges mentioned above, we built a highly efficient multilingual language model for the BabyLM 2026 Multilingual Track. Beyond the engineering contributions listed below, our experiments also yield a concrete scientific finding: unlike much of the prior curriculum-learning literature, where dynamically reordering or reweighting training data has often failed to yield consistent overall gains, we find that a dynamic, stage-based multilingual curriculum *does* meaningfully improve causal language model performance in this data-constrained multilingual setting—though, notably, the same curriculum does not transfer as effectively to masked language models, as we discuss later in this paper. Our main contributions are summarized below:

- **Curriculum-based Multilingual Data Mixing:** We propose a three-stage curriculum learning strategy for multilingual language modeling and show empirically that it improves causal language model (GPT-2) performance over static and naive mixing baselines. Combined with a multilingual data cleaning and multi-level quality-gating pipeline, our approach dynamically adjusts language mixing ratios while preserving informative mixed-language documents to facilitate cross-lingual transfer under a fixed data budget.
- **Custom Vocabulary:** We design a custom BPE tokenizer that balances the vocabulary size across the three target languages. This approach prevents Chinese characters from disproportionately dominating the vocabulary capacity, which saves a significant number of parameters for the actual transformer layers.
- **Comprehensive ablation study:** We conduct a comprehensive ablation study of vocabulary size, context length, hyperparameters, and language model architectures under the BabyLM multilingual constraints.

2 Background and Related Work

2.1 Multilingual Data Mixing

When training a multilingual model, a critical question is how to mix data from different languages. Simple approaches involve either equal mixing or sampling based strictly on original dataset sizes; however, these static methods are suboptimal. As demonstrated by Lample and Conneau (2019), sampling based strictly on dataset size biases the model heavily towards high-resource languages, causing it to ignore low-resource ones.

To fix this, researchers often use temperature-based sampling to balance high-resource and low-resource languages. But, keeping this mixing ratio static throughout the entire training process ignores how the model's learning needs change over time. Recent studies suggest that dynamic data mixing, inspired by curriculum learning, yields much better results. For instance, Zhang et al. (2021) introduced a curriculum framework that dynamically schedules the introduction of languages based on the model's learning state. To address this in our work, we implement a dynamic, three-stage curriculum learning strategy that adjusts language mixing ratios as training progresses, allowing the model to establish high-resource linguistic foundations before tackling harder cross-lingual alignments.

2.2 Tokenization

Multilingual Tokenizer Variations Tokenization has a massive impact on model performance, especially when applying multilingual language models to monolingual tasks (Rust et al., 2021). Due to the 'curse of multilinguality' (Conneau et al., 2020), a standard shared tokenizer often lacks the vocabulary capacity to represent all languages equitably, underrepresenting certain languages even with pretraining oversampling (Conneau et al., 2020). If a tokenizer is not carefully adapted, it can aggressively over-segment morphologically rich languages—such as Arabic, Finnish, Korean, or Turkish—leading to high subword fertility that severely hurts downstream performance (Rust et al., 2021).

Vocabulary Size Constraints Another major issue is the size of the tokenizer's vocabulary. To support subwords in the Latin alphabet (for English and Dutch) and thousands of characters (for Chinese), the vocabulary must be large. However, in small-scale models typically used in the BabyLM track, a large vocabulary creates a severe parameter bottleneck (Dagan et al., 2024). Every token in the vocabulary requires an embedding vector. If the vocabulary size is excessively large, the embedding matrix accounts for a substantial proportion of the model parameters. Whether it is more advantageous to allocate parameters to the Transformer backbone rather than to the embedding layer is

ultimately an empirical trade-off: an embedding layer that is too small under-represents the script diversity of the target languages, while one that is too large starves the Transformer backbone of capacity. Our own ablation results (Section 5.1) support this view, showing that performance peaks at an intermediate vocabulary size (32k) rather than improving monotonically as the embedding layer shrinks. Selecting an appropriate vocabulary size therefore constitutes an important architectural design trade-off for efficient multilingual language modeling, rather than a simple rule that smaller embeddings are always better.

2.3 Architectures

Under the limited training budgets of the BabyLM Challenge, model architecture plays an important role in determining data efficiency and downstream performance. We consider both encoder-only Masked Language Models (MLMs), such as BERT (Devlin et al., 2019), and decoder-only Causal Language Models (CLMs), such as GPT-2 (Radford et al., 2019). Notably, encoder-only architectures have proven highly competitive in prior data-efficient settings: in the 2023 BabyLM Challenge, MLM-based submissions such as LTG-BERT achieved the highest average performance, outperforming models trained on vastly larger corpora (Warstadt et al., 2023). GPT-2 additionally serves as the official baseline architecture in the 2026 multilingual track. In this work, we investigate both BERT-style MLMs and GPT-2-style CLMs under different vocabulary sizes, packing strategies, and data mixing strategies to better understand their behavior in multilingual low-resource language modeling.

3 Method

3.1 Data Cleaning and Filtering

To maximize the information density of each language's training signal within our fixed 100M-token budget—removing noisy or duplicated text that would otherwise consume tokens without contributing to the model's linguistic competence—we created a step-by-step cleaning process for the English (`eng`), Dutch (`nld`), and Chinese (`zho`) text. Instead of treating all languages the same, we carefully remove low-quality data while keeping the useful parts using three main rules.

First, **Language-Specific Normalization** repairs nonstandard formatting. For example, we clean up Chinese conversations by standardizing speaker names and adding line breaks. We also remove extra spaces and replace difficult math equations with a simple `[formula]` tag so the model would not focus on math details during training.

Second, **Redundancy Elimination** removes duplicate text. We convert all characters to lowercase and strip all whitespace and punctuation before checking for duplicates. This ensures the removal of repeated sentences even in the presence of slight typographical variations. Through this process, we successfully filtered out 12 duplicate instances (0.87%) from the English dataset, 654 instances (7.93%) from the Dutch dataset, and 896 instances (5.09%) from the Chinese dataset.

Third, **Multi-Level Quality Gating** applies strict thresholds to filter out low-quality text. We enforce length filters, requiring Chinese documents to be at least 3 characters and English/Dutch documents to be at least 30 characters. Additionally, we implement a stopword-based purity check specifically for the Dutch corpus to filter out English contamination (the English and Chinese corpora naturally maintain high purity due to source reliability and distinct scripts). Finally, we actively extract mixed-language documents by detecting Latin characters within the Chinese training corpus. We isolate these specific texts to facilitate cross-lingual transfer during Stage 2 of training. The exact rules and formulas for this cleaning process are detailed in Appendix A.

Before training, we randomly take out 5% of the clean data from each language to use as a validation set. The rest of the data is used for training.

3.2 Data Mixing Strategy

The BabyLM competition sets strict limits on training data. Instead of relying on a static data mixing ratio, we propose a dynamic, three-stage curriculum learning strategy strictly regulated by precise Byte-Pair Encoding (BPE) token accounting. All samples in these stages are drawn from the global training pool.

Byte-Premium Sampling Different languages exhibit varying levels of tokenization efficiency, requiring different amounts of token space to convey the same information. This disparity is formally known as the "byte premium" (Arnett et al., 2024). To ensure the actual training tokens for each language strictly adhere to our budget ratios, we implement an exact BPE token calculation mecha-

nism. For a given target budget, our system counts the exact number of processed tokens rather than estimating by character or word count. Furthermore, we introduced premium factors (English is 1.0, Dutch is 1.0516, and Chinese is 0.93) to adjust the allowed sampling volume. We divide our data budget by these numbers. This step ensures we stop training exactly when we hit our data limit.

Baseline Mixing Strategies To strictly evaluate the effectiveness of our proposed curriculum, we implement two comparative baseline strategies:

- **Naive Mixing:** This approach aggregates the raw corpora uniformly without enforcing specific language balancing or strict per-language token budget constraints. It serves as the foundational lower-bound reference for our multilingual modeling.
- **Static Mixing:** This strategy maintains a fixed, predetermined language distribution across the entire training budget. Unlike the dynamic curriculum approach, the sampling ratio (44% English, 23% Dutch, and 33% Chinese) remains completely constant from the initial to the final training step.

Three-Stage Curriculum Mixing We divide the total training budget into three progressive stages, dynamically adjusting the language ratios based on the model's learning requirements:

- **Stage 1: Foundation (30% of total budget)** This stage focuses on establishing fundamental linguistic structures. We allocate 50% of the stage budget to English, since English benefits from the largest share of our asymmetric BPE tokenizer's training corpus and therefore the most efficient subword segmentation, allowing the model to quickly build core syntactic and semantic representations before addressing typologically more distant languages. Chinese and Dutch each receive 25%.
- **Stage 2: Alignment (30% of total budget)** Aiming to strengthen cross-lingual representations, we balance the distribution (33% English, 33% Chinese, 34% Dutch). Crucially, we enable an **Aligned Upsampling** mechanism for the Chinese split: half of the Chinese data is sampled from the mixed-language texts identified during preprocessing (Section 3.1), which act as a bridge between the languages and help the model relate the foundational English representations from Stage 1 to Chinese and Dutch.
- **Stage 3: Hard Boosting (40% of total budget)** Once a foundational cross-lingual mapping is established, we shift the focus toward Chinese and Dutch, whose subword vocabularies are comparatively less optimized due to their smaller share of the tokenizer training corpus, and whose typological distance from English—a character-based script for Chinese, distinct morphosyntax for Dutch—makes cross-lingual transfer more challenging. The distribution is skewed heavily towards Chinese (40%) and Dutch (40%), reducing English to 20%, thereby pushing the model to strengthen its representations of the non-English languages once a shared foundation is in place.

During data mixing at each stage, all samples are globally shuffled. To prevent the model from encountering identical data sequences across different stages, we dynamically generate unique random seeds based on the hash of the stage name. This approach ensures a robust and continuously varied training distribution throughout the curriculum.

3.3 Custom BPE Tokenizer and Asymmetric Sampling

We train our own BPE tokenizer from scratch on the sampled dataset, rather than adopting a massive, ready-made tokenizer. This approach allowed us to use more parameters for the main model by keeping the vocabulary size small. For the 100M budget, we evaluated the vocabulary with 8k, 16k, 32k and 64k. Additionally, to ensure proper processing of Chinese semantics, we explicitly register common Chinese punctuation marks as special tokens.

Asymmetric Sampling Strategy When constructing the tokenizer training corpus, we do not sample the three languages uniformly. Instead, we adopt an asymmetric sampling ratio. For the 100M track, the tokenizer is trained on a 15M-token corpus consisting of 80% English, 17% Dutch, and 3% Chinese.

This distribution is chosen because Chinese characters typically encode more information per character and require substantially larger vocabulary coverage than alphabetic languages. An equal sampling ratio would therefore allocate a disproportion-

ate fraction of the vocabulary to Chinese, reducing the capacity available for English and Dutch subwords. This strategy encourages efficient subword learning for English and Dutch while preserving sufficient coverage of commonly used Chinese characters.

Bootstrapping Workflow The tokenizer training process is integrated into our precise budget-calculation pipeline through a three-step bootstrapping workflow. First, we extract a heuristic dataset from our pre-cleaned global training pool using basic length estimations (character counts for Chinese and whitespace splits for English and Dutch). Second, we train the custom BPE tokenizer on this proxy dataset. Finally, as detailed in Section 3.2, we feed the trained tokenizer back into the main data pipeline to execute exact, BPE-based token counting (adjusted via the Byte Premium). This bootstrapping process guaranteed that our final dataset perfectly adhered to the strict competition budget limits.

3.4 Model Architecture

In this study, we implement and compare two distinct transformer-based architectures.

Autoregressive Baseline (GPT-2): We implement a standard decoder-only GPT-2 architecture. This model relies on causal language modeling (CLM) and we implement two packing strategies: *wrapped-packing and bfd-split*.

Bidirectional Encoder (BERT) with Chunking and Wrapping: We implement a BERT-style Masked Language Model (MLM) to learn bidirectional contextual representations and compare three sequence formatting strategies:

- *Naive Chunking:* Documents are truncated or padded to a fixed sequence length (e.g., 32, 128, 256, or 512 tokens). This approach is simple but inherently inefficient, as it either splits semantic context across chunk boundaries or wastes the strict data budget on useless padding tokens.

- *Document Wrapping (Packing):* Multiple documents are concatenated into fixed-length sequences separated by EOS/SEP tokens. This strategy improves sequence utilization by reducing padding overhead while allowing more training tokens to be processed within the fixed token budget.

- *BFD-Split (Best-Fit Decreasing Split):* As an alternative to naive chunking, documents are first sorted by length and then packed into fixed-length sequences using a best-fit decreasing heuristic without crossing document boundaries.

Controlled Token Exposure per Update To enable a fair comparison across different sequence lengths, we control the effective number of tokens processed per optimizer update:

$$E = B \times L \times G,$$

where B denotes the micro-batch size, L the sequence length, and G the gradient accumulation factor.

For the controlled curriculum-learning experiments, we fix $E = 2048$ across sequence lengths of 32, 64, 128, 256, and 512 by adjusting G accordingly. This reduces confounding effects caused by different optimizer update frequencies and enables a more controlled evaluation of sequence-length effects.

4 Experimental Setup

4.1 Dataset

The training data is sampled from the official BabyLM Challenge dataset (BabyLM Multilingual datasets, 2026), built on the BabyBabelLM multilingual corpus (Jumelet et al., 2026a), specifically targeting English (`eng`), Dutch (`nld`), and Chinese (`zho`). To ensure high-quality inputs, we applied a strict data preprocessing pipeline, as detailed in Section 3.1. Furthermore, to account for the information density differences across languages, we implement a Byte Premium scaling factor to adjust the exact token budgets during sampling. For data mixing, we explored strategies including Naive Baseline (equal ratios), Static Baseline (44% EN, 33% ZH, 23% NL), and a curriculum-based progression.

4.2 Tokenizer Settings

We trained a custom BPE tokenizer from scratch using a heuristically sampled multi-lingual corpus (80% English, 17% Dutch, 3% Chinese). The tokenizer is configured with a vocabulary size of 32,000 to fit the 100M track budget. The special tokens include `<unk>`, `<s>`, `</s>`, `<pad>`, `<mask>`, alongside a set of common Chinese punctuation marks to prevent unnecessary fragmentation. A

post-processor is applied to append <s> and </s> to sequence boundaries.

4.3 Model Configuration

Although we train and evaluate both the MLM and CLM objectives introduced in Section 2.3, we focus our main-text discussion and hyperparameter details on the decoder-only GPT-2 architecture, since, as we show in Section 5.1, GPT-2 consistently outperforms our BERT models across all ablations and official evaluations. Full BERT configurations and results are reported in Appendix C. Table 1 details the GPT-2 configuration used for our 100M track experiments.

Hyperparameter	Value
Model Architecture	GPT-2
Hidden Size	768
Num Layers	12
Attention Heads	12
Inner Dimension	3072
Vocab Size	32,000
Max Seq Length	512

Table 1: Model architecture and training hyperparameters for the GPT-2 baseline.

4.4 Evaluation

We evaluate our models locally using the official BabyLM Evaluation Pipeline (Choshen et al., 2026a), which relies on the Language Model Evaluation Harness (`lm-eval`). Our zero-shot suite covers BLiMP (Warstadt et al., 2020), MultiBLiMP (Jumelet et al., 2026b), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2019), XStoryCloze (Lin et al., 2022b), and XCOMPS (He et al., 2025); our fine-tuning suite covers ARC (Clark et al., 2018), Belebele (Bandarkar et al., 2024), BMLAMA (Qi et al., 2023), INCLUDE (Romanou et al., 2024), MNLI (Williams et al., 2018), SIB-200 (Adelani et al., 2024), TruthfulQA (Lin et al., 2022a), and XNLI (Conneau et al., 2018). Zero-shot and fine-tuned evaluations are conducted on the three primary languages: `eng`, `nld`, and `zho`. The final overall scores are calculated strictly according to the methodology defined by the official 2026 BabyLM Leaderboard (BabyLM-community, 2026).

5 Results and Analysis

5.1 Ablation Studies

To optimize performance under the strict 100M token limit, we conduct systematic ablation studies focusing on tokenizer vocabulary size, data mixing strategies, sequence packing methods, sequence length, and hyperparameters. Exhaustively testing every combination of these factors was computationally infeasible under our budget. Instead, within each ablation group, we vary one primary factor while holding the remaining settings fixed. Due to computational constraints, different ablation groups were conducted at different model scales: the vocabulary-size and tokenizer experiments use the Standard configuration (hidden size 768, 12 layers, and 12 attention heads), whereas the data-mixing and sequence-length experiments use the Small configuration (hidden size 256, 4 layers, and 4 attention heads). Unless otherwise stated, the experiments use our custom asymmetric BPE tokenizer (32k vocabulary, 80C.

A clear and consistent trend across all experiments is that GPT-2 consistently outperforms the official baseline (38.86), whereas BERT consistently underperforms the baseline across all tested variations.

First, we evaluate different BPE vocabulary sizes—specifically 8,000, 16,000, 32,000, and 64,000 tokens—trained via our custom BPE trainer under our asymmetric sampling strategy (Section 3.3), while keeping the curriculum data mixing, Pack-Wrap packing, and sequence length fixed at their default values. As shown in Table 2, the optimal vocabulary size depends on the model architecture. For GPT-2, a vocabulary size of 32,000 yields the highest overall score (44.06) and the best fine-tuned average across languages (37.32%). Detailed results for BERT are provided in Appendix C, Table 8.

Vocab	Zero-shot (%)				Fine-tune (%)				Score
	en	nl	zh	avg	en	nl	zh	avg	
Base.	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
8k	52.69	55.51	49.28	52.49	35.87	33.70	34.61	34.73	42.30
16k	53.41	55.84	49.77	53.01	36.16	**35.84**	34.12	35.37	42.85
32k	**53.51**	55.88	49.87	53.09	**38.81**	35.66	**37.47**	**37.32**	**44.06**
64k	53.41	**56.38**	**50.08**	**53.29**	35.57	33.98	37.43	35.66	43.20

Table 2: Ablation results for the Standard GPT-2 configuration (768/12/12) across different BPE vocabulary sizes. Bold values indicate the best performance among the evaluated configurations in each column.

Second, we investigate three distinct data mixing strategies to efficiently allocate the 100M token budget: a *naive* uniform sampling, a *static* weighted sampling based on task relevance, and a *curriculum* learning approach, all using our custom asymmetric BPE tokenizer with the best-performing 32k vocabulary identified above, and default packing and sequence length settings. As shown in Table 3, for GPT-2, the *curriculum* approach achieves the highest Overall Score (41.59) and the best fine-tuned average performance. Detailed results for BERT are provided in Appendix C, Table 9.

Strategy	Zero-shot (%)				Fine-tune (%)				Score
	en	nl	zh	avg	en	nl	zh	avg	
Base.	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
Naive	52.27	54.19	48.78	**51.74**	33.91	**33.00**	33.75	33.56	41.29
Static	**52.44**	53.95	48.57	51.65	33.87	32.17	34.01	33.35	41.15
Curr.	51.65	**54.22**	**48.85**	51.57	**35.46**	32.02	**34.96**	**34.15**	**41.59**

Table 3: Ablation results for GPT-2 data mixing strategies using the Small configuration (256/4/4). Bold values indicate the best performance among the evaluated strategies in each column.

Furthermore, to validate the effectiveness of our asymmetric sampling strategy, we compared our custom BPE tokenizer against a standard default BPE tokenizer trained on a uniformly sampled multilingual corpus. As demonstrated in Table 4, the custom asymmetric BPE tokenizer yields superior overall performance compared to the standard approach. This confirms that deliberately balancing vocabulary allocation based on the varying information densities of the target scripts is crucial for maximizing tokenization efficiency under strict parameter constraints.

Method	Zero-shot (%)				Fine-tune (%)				Score
	en	nl	zh	avg	en	nl	zh	avg	
Default BPE	53.26	56.68	50.24	53.39	36.05	33.14	34.86	34.68	42.67
Custom BPE	**53.51**	**55.88**	**49.87**	**53.09**	**38.81**	**35.66**	**37.47**	**37.32**	**44.06**

Table 4: Comparison between default uniformly sampled BPE and custom asymmetric BPE using the Standard GPT-2 configuration (768/12/12) with a 32k vocabulary.

Third, we compare sequence packing strategies under the default sequence length of 256 and our best-performing 32k custom BPE tokenizer with curriculum mixing. For GPT-2, we compare Pack-Wrap (document wrapping) against Bfd-Split; as shown in Table 5, Pack-Wrap achieves a marginally higher overall score (44.55) than Bfd-Split (44.35), indicating that simple document-wrapping concatenation is at least as effective as, and slightly better than, sorting-based best-fit packing for causal language modeling. For BERT, we additionally compare against Naive Chunking; as shown in Table 12 (Appendix C), the three strategies perform comparably (35.42–35.45 overall score), with no packing method showing a consistent advantage for the masked language modeling objective—unlike for GPT-2, where document wrapping shows a small but consistent edge.

Lastly, we evaluate the impact of sequence length and hyperparameter scaling on model performance. The detailed discussions and corresponding results are provided in Appendix Tables 10 and 11.

5.2 Training Dynamics

To evaluate learning stability and capability during the curriculum learning process, we analyze the validation loss progression of GPT-2 and BERT across three distinct training stages. As illustrated in Figure 1, the training dynamics of the two models exhibit significant differences.

For GPT-2, the validation loss decreases consistently across all three curriculum stages, dropping from 4.002 in Stage 1 to 3.498 in Stage 3. This indicates that GPT-2 benefits from the progressively more challenging curriculum.

In contrast, BERT maintains a much higher validation loss and shows only limited improvement during training. For the Pack Wrapped setting, the validation loss changes from 7.612 (Stage 1) to 7.533 (Stage 2) and then increases to 7.640 (Stage 3). The Chunk configuration exhibits a similar trend, with the loss increasing from 7.600 in Stage 1 to 7.820 in Stage 3. Overall, unlike GPT-2, BERT does not achieve continuous improvement throughout the curriculum.

We note two caveats when interpreting these results. First, because MLM and CLM losses are computed over different prediction objectives (bidirectional cloze reconstruction versus unidirectional next-token prediction), their absolute magnitudes are not directly comparable; the loss gap between GPT-2 and BERT in Figure 1 should not be read as evidence that one model is learning more effectively than the other overall, only that their loss scales differ. Second, BERT's lack of consistent improvement across curriculum stages may not solely reflect an inherent incompatibility between the MLM objective and curriculum-based data mix-

ing; it could also stem from hyperparameters (e.g., learning rate) that were primarily tuned for GPT-2. We leave a dedicated hyperparameter search for BERT under the curriculum setting to future work.

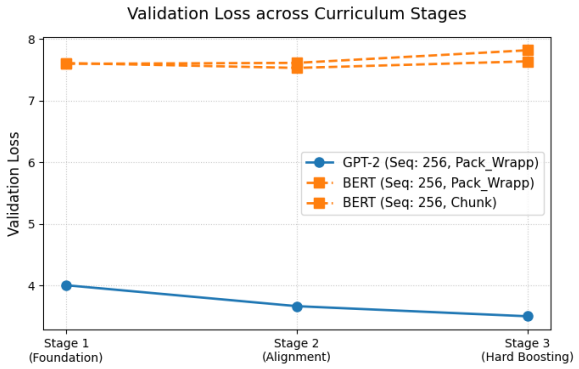


Figure 1: Overview of the validation loss progression of GPT-2 and BERT.

5.3 Official Evaluation Results

We evaluate our models using the official babylm-eval pipeline. Table 5 summarizes the zero-shot and fine-tuned performance of our causal (GPT-2) models, each trained with our best-performing configuration from the ablation studies above (32k custom asymmetric BPE tokenizer, three-stage curriculum mixing, sequence length 256), comparing the two sequence packing variants (Pack-Wrap and Bfd-Split) discussed in Section 5.1. Overall, the GPT-2 models achieve strong results, with the best model reaching an overall score of 44.55, significantly outperforming the official baseline (38.86). While we also evaluated masked (BERT) architectures, the GPT-2 models consistently outperformed their BERT counterparts across our evaluations (detailed BERT results are provided in Appendix 12).

As shown in Tables 6 and 7, our modeling strategies and custom BPE tokenization prove particularly effective for English `sib200` fine-tuning, where our curriculum GPT-2 Pack-Wrapped model scores 78.50, outperforming the official baseline of 21.50 by 57.00 points. However, performance on certain zero-shot grammar evaluations, such as English `multiblimp`, indicates room for improvement; our GPT-2 model scores 79.61, failing to surpass the baseline's 87.14.

Seq	Method	Fine-tune				Zero-shot				Score
		en	zh	nl	avg	en	zh	nl	avg	
-	Baseline	28.34	27.23	24.78	26.78	56.34	51.34	57.81	55.16	38.86
256	**Pack-Wrap**	**40.64**	36.70	**36.79**	**38.04**	**53.94**	**49.78**	**56.19**	**53.30**	**44.55**
256	Bfd-Split	40.05	39.05	34.16	37.75	53.34	49.78	56.12	53.08	44.35

Table 5: Official evaluation results for curriculum GPT-2 using the Standard configuration (768/12/12). Bold values indicate the best score among our two GPT-2 packing variants (Pack-Wrap, Bfd-Split) in each column; note that the official baseline outperforms both variants on every zero-shot metric, while our models substantially outperform it on fine-tuning metrics and overall score.

Lang.	Task	Baseline	GPT-2 Pack-Wrap
ENG	blimp	69.18	65.51
	hellaswag	26.66	26.03
	multiblimp	87.14	79.61
	winogrande	48.80	50.04
	xstorycloze	49.92	48.53
	avg	**56.34**	**53.94**
NLD	blimp_nl	79.27	75.27
	hellaswag	26.37	26.00
	multiblimp	91.85	88.20
	winogrande	49.30	47.40
	xcomps_nl	52.68	51.59
	xstorycloze	47.37	48.68
	avg	**57.81**	**56.19**
ZHO	blimp	75.44	71.86
	hellaswag	27.05	26.75
	winogrande	49.71	49.88
	xcomps_zh	53.90	52.64
	xstorycloze	50.62	47.76
	avg	**51.34**	**49.78**
Overall Average		**55.16**	**53.30**

Table 6: Zero-shot tasks results for GPT-2.

Lang.	Task	Baseline	GPT-2 Pack-Wrap
ENG	arc	24.79	27.29
	belebele	22.16	25.57
	bmlama	9.35	22.27
	mnli	51.63	50.68
	sib200	21.50	78.50
	truthfulqa	22.32	33.93
	xnli	46.65	46.25
	avg	**28.34**	**40.64**
NLD	arc	24.38	26.25
	belebele	30.11	31.82
	bmlama	10.68	23.84
	include	19.64	27.68
	mnli	43.92	49.89
	sib200	21.50	69.50
	truthfulqa	23.21	28.57
	avg	**24.78**	**36.79**
ZHO	arc	25.00	24.38
	belebele	22.73	19.89
	bmlama	9.35	19.78
	include	26.79	27.68
	mnli	45.83	54.22
	sib200	21.50	76.00
	truthfulqa	23.21	23.21
	xnli	43.45	48.45
	avg	**27.23**	**36.70**
Overall Average		**26.78**	**38.04**

Table 7: Fine-tuning task results for GPT-2.

Conclusion

In this work, we present an optimized approach for training small-scale multilingual language models under the strict 100M token of the BabyLM 2026 Multilingual Track. By implementing a custom BPE tokenizer with asymmetric sampling, we effectively balance the vocabulary allocation across languages with distinct information densities, ensuring optimal tokenization efficiency for each script. Furthermore, our dynamic three-stage curriculum learning strategy successfully guides the model from high-resource linguistic foundations to harder, cross-lingual alignments. Our evaluations demonstrate that the GPT-2 architecture adapts exceptionally well to this curriculum, achieving a strong overall score of 44.55 and significantly outperforming the baseline. Coupled with our Pack-Wrapped data packing method, these strategies provide a highly data-efficient framework for multilingual language modeling.

Limitations

Despite the strong overall performance, our approach has several limitations. First, while our asymmetric tokenization strategy aims to balance vocabulary allocation efficiently, heavily skewing the tokenizer corpus towards English and Dutch may introduce slight language-specific biases, potentially limiting the representational depth of Chinese characters. Second, our evaluation reveals that the models struggle to surpass the official baseline in certain zero-shot grammar tasks, such as English `multiblimp`, indicating that the current curriculum may prioritize semantic alignment over strict syntactic rule acquisition. Third, our curriculum stage transitions are highly beneficial for the autoregressive GPT-2 model but fail to yield similar improvements for the masked BERT architecture, suggesting that MLM objectives might require a fundamentally different data mixing schedule. Fourth, strict computational constraints and the 10-epoch limit restrict our capacity for exhaustive hyperparameter tuning and finer-grained stage adjustments. Finally, our ablation studies (Section 5.1) vary one factor at a time while holding the others fixed at a single default configuration, rather than exhaustively testing every combination; the reported optimal settings (e.g., the 32k vocabulary size, the asymmetric sampling ratio) are therefore only shown to be optimal under that particular default configuration, and could interact with other factors in ways we have not measured.

Another limitation concerns checkpoint reproducibility. Due to storage constraints, the exact checkpoint corresponding to the best exploratory evaluation run was not preserved. The leaderboard submission was therefore produced by retraining the same configuration from scratch. As a result, the leaderboard score may differ from the score reported for the original run. We report the original evaluation results to analyze the effectiveness of the configuration, while the submitted checkpoint should be interpreted as a reproduced model trained under the same setting rather than the identical checkpoint.

Code Availability

The complete source code, preprocessing scripts, tokenization pipelines, and training configurations for this project are publicly available at:
`https://anonymous.4open.science/r/BabyLM_Challenge_III-BFCE`

Acknowledgments

We are also deeply grateful to the Technical University of Munich (TUM) for providing the essential computational resources through the Leibniz Supercomputing Centre (LRZ) cluster. Furthermore, we would like to thank the BabyLM challenge and Hugging Face for making their datasets publicly available, which were instrumental to the completion of this research.

The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universit"at Erlangen-N"urnberg (FAU) under the NHR project b279bb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O. Alabi, Yanke Mao, Hao-

nan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, St. Julian's, Malta.

Catherine Arnett, Tyler A. Chang, and Benjamin K. Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. *arXiv preprint arXiv:2403.00686*.

BabyLM-community. 2026. Babylm leaderboard 2026. https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026. Accessed: 2026-07-02.

BabyLM Multi-lingual datasets. 2026. Babylm 2026 dataset collection. https://huggingface.co/collections/BabyLM-community/babylm-2026. Accessed: 2026-06-09.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The Belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026a. Babylm turns 4: Call for papers for the 2026 babylm workshop. *arXiv preprint arXiv:2602.20092*.

Leshem Choshen, Tal Linzen, Raj Sanjay Shah, BabyLM Team, Ryan Cotterell, Mustafa Omer Gul, Aaron Mueller, Alex Warstadt, Jaap Jumelet, Suchir Salhan, and Ethan Gotlieb Wilcox. 2026b. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *arXiv preprint arXiv:2602.20092*, v2.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation. In *arXiv preprint arXiv:2402.01035*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schütze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP (SIGTYP 2025)*, Vienna, Austria.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, Rabat, Morocco.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, and 2 others.

2022b. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052. Introduces the XStoryCloze evaluation set.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2023. Cross-lingual consistency of factual knowledge in multilingual language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10650–10666, Singapore.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical Report 1, OpenAI.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A. Haggag, Snegha A, Alfonso Amayuelas, Azril Hafizi Amirudin, Viraat Aryabumi, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Aditya Kumar Dalmia, Abraham Diress, and 40 others. 2024. INCLUDE: Evaluating multilingual language understanding with regional knowledge. In *arXiv preprint arXiv:2411.19799*.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WinoGrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the 27th Conference on Computational Natural Language Learning: Volume 2: The BabyLM Challenge*, pages 1–34, ETH Zürich. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy.

Mingliang Zhang, Fandong Meng, Yunhai Tong, and Jie Zhou. 2021. Competence-based curriculum learning for multilingual machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2481–2493, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Full Data Cleaning Rules

This appendix explains the exact rules mentioned in Section 3.1.

A.1 Text Normalization and Changes

- **Cleaning Chinese Conversations:** We change speaker names to standard ones (for example, changing `*TEACHER:` to `lao shi:`). We also add line breaks (`\n`) before a new person speaks to help the model understand who is talking.
- **Changing Math Equations:** We categorize math equations using regular expressions. "Simple" formulas, defined strictly as those containing only alphanumeric characters (e.g., single variables like x or y), are kept in plain text by stripping their surrounding LaTeX tags. Conversely, any complex formula containing spaces, operators, or special LaTeX syntax is replaced entirely with a single `[formula]` tag.

A.2 Removing Duplicates and Strict Quality Rules

- **Length Filter:** We throw away text that is too short. Chinese text (`zho`) must be at least **3 characters** long, and English (`eng`) and Dutch (`nld`) must be at least **30 characters** long. We also delete any text that has more than 3 `[formula]` tags.
- **Dutch Purity Check:** To make sure the Dutch text does not have too much English, we count the common words (stopwords). We throw away the text if there are more English common words than Dutch common words. Also, if a Dutch sentence has more than 10 words but no Dutch common words, we delete it.

- **Finding Mixed Languages:** We give special importance to Chinese text that also has some English letters. This shows the text mixes both languages, which helps connect them in Stage 2 of training.

B Training System and Token Exposure Tracking

Models are trained using the Hugging Face Transformers framework and optimized with the AdamW optimizer using a learning rate of 1×10^{-4}.

To ensure compliance with the multilingual BabyLM token budget, we maintain runtime token-exposure tracking during training. Token counts are monitored using the attention mask and language identifiers associated with each training sample. The adjusted multilingual exposure follows the byte-premium weighting scheme described in Section 3.2.

Training is controlled by adjusted token exposure, with a maximum of 10 epochs. Before processing each batch, the trainer estimates the additional adjusted exposure that the batch would introduce. If adding the batch would exceed the current token budget, the batch is skipped and training terminates at the last valid state. This strict pre-check ensures that the final adjusted exposure never exceeds the 100M adjusted-token limit.

Padding tokens are excluded from exposure accounting. For GPT-2 packed training, artificial document-boundary EOS tokens are assigned ignored language identifiers and excluded from language-specific exposure accounting. For curriculum-learning experiments, the same strict exposure tracking is applied cumulatively across stages.

C Detailed Ablation Results and Training Dynamics

Vocab	Zero-shot (%)				Fine-tune (%)				Score
	en	nl	zh	avg	en	nl	zh	avg	
Base.	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
8k	49.68	49.18	47.52	48.80	25.42	25.80	25.59	25.60	35.38
16k	49.76	49.45	47.40	48.87	25.42	25.80	25.59	25.60	35.42
32k	49.49	49.78	47.54	48.94	25.63	25.17	25.78	25.53	35.42
64k	**50.09**	**50.14**	**47.02**	**49.08**	**25.23**	**25.80**	**25.42**	**25.48**	**35.45**

Table 8: Ablation results for the Standard BERT configuration (768/12/12) across different BPE vocabulary sizes.

Strategy	Zero-shot (%)				Fine-tune (%)				Score
	en	nl	zh	avg	en	nl	zh	avg	
Base.	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
Naive	46.82	46.33	45.17	46.11	30.70	27.43	32.61	30.25	37.00
Static	**47.10**	**47.41**	**45.68**	**46.73**	**31.53**	**29.07**	**33.15**	**31.25**	**37.84**
Curr.	47.54	45.23	46.57	46.44	32.04	28.27	32.17	30.83	37.43

Table 9: Ablation results for BERT data mixing strategies.Ablation results for BERT data mixing strategies using the Small configuration (256/4/4).

We analyze the impact of sequence length on model performance and training efficiency, evaluating lengths of 32, 64, 128, 256, and 512 tokens. According to Table 10, a sequence length of 512 with GPT-2 achieves the highest Overall Score (41.54). For BERT, performance peaks at an Overall Score of 37.71 with a sequence length of 128.

Finally, we compare hyperparameters, specifically a standard configuration ($h = 768, L = 12, a = 12$) against a smaller configuration ($h = 256, L = 4, a = 4$). As detailed in Table 11, scaling up the model dimensions yields higher overall scores for GPT-2 (44.06). Interestingly, while the standard configuration consistently improves GPT-2's performance across all metrics, BERT exhibits a trade-off: scaling up shows strong gains in zero-shot capabilities but causes a noticeable decline in its average fine-tuned performance (dropping from 30.25% to 25.53%).

Model	Seq	Zero-shot (%)				Fine-tune (%)				Score
		en	nl	zh	avg	en	nl	zh	avg	
Base.	-	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
GPT-2	32	50.49	54.04	49.67	51.40	26.31	26.54	32.98	28.61	38.34
	64	51.92	54.48	48.81	51.74	30.94	29.21	34.01	31.38	40.09
	128	52.13	54.63	48.50	51.75	31.41	30.04	35.22	32.22	40.59
	256	52.27	54.19	48.78	51.74	33.91	33.00	33.75	33.56	41.29
	512	**51.28**	**53.81**	**48.26**	**51.12**	**34.22**	**33.69**	**35.33**	**34.41**	**41.54**
BERT	32	45.84	44.29	46.35	45.50	29.55	28.61	30.77	29.64	36.32
	64	46.75	45.19	45.78	45.91	31.79	28.44	32.93	31.05	37.35
	128	**46.21**	**46.29**	**45.30**	**45.94**	**33.61**	**28.58**	**32.63**	**31.61**	**37.71**
	256	46.82	46.33	45.17	46.11	30.70	27.43	32.61	30.25	37.00
	512	48.12	46.42	46.19	46.91	30.11	27.92	32.09	30.04	37.19

Table 10: Sequence-length ablation results using the Small GPT-2 and BERT configurations (256/4/4).

Model	Params	Zero-shot (%)				Fine-tune (%)				Score
		en	nl	zh	avg	en	nl	zh	avg	
Base.	-	56.34	57.81	51.34	55.16	28.34	24.78	27.23	26.78	38.86
GPT-2	Small	52.27	54.19	48.78	51.74	33.91	33.00	33.75	33.56	41.29
	Std.	**53.51**	**55.88**	**49.87**	**53.09**	**38.81**	**35.66**	**37.47**	**37.32**	**44.06**
BERT	Small	46.82	46.33	45.17	46.11	**30.70**	**27.43**	**32.61**	**30.25**	**37.00**
	Std.	**49.49**	**49.78**	**47.54**	**48.94**	25.63	25.17	25.78	25.53	35.42

Table 11: Ablation results comparing different hyperparameter configurations.

In addition to the GPT-2 models discussed in the main text, we evaluated masked language models using the BERT architecture. Table 12 summarizes the official evaluation results for BERT trained with static data mixing strategies and chunking or BFD-split methods.

The highest overall score achieved by our BERT models is 35.45, which underperforms compared to both the official baseline (38.86) and our GPT-2 models (44.55). Detailed performance breakdowns for BERT across individual zero-shot and fine-tuning tasks are provided in Tables 13 and 14, respectively. Although BERT struggles to match the baseline overall, the BFD-Split strategy with a sequence length of 128 slightly improves its fine-tuning average (25.48%) and zero-shot average (49.11%) compared to naive chunking. The results suggest that the masked language modeling objective, when constrained by strict vocabulary limits and the 100M token budget, may require substantially different hyperparameter tuning or training durations compared to autoregressive models.

Seq	Method	Fine-tune				Zero-shot				Score
		en	zh	nl	avg	en	zh	nl	avg	
-	Baseline	28.34	27.23	24.78	26.78	56.34	51.34	57.81	55.16	38.86
256	**Chunk**	**25.63**	**25.78**	**25.17**	**25.53**	**49.78**	**47.82**	**49.47**	**49.02**	**35.45**
256	Pack-Wrap	25.23	25.42	25.80	25.48	49.97	47.54	49.63	49.05	35.42
128	**Bfd-Split**	**25.23**	**25.42**	**25.80**	**25.48**	**49.74**	**47.68**	**49.91**	**49.11**	**35.45**

Table 12: Official evaluation results for selected Standard BERT configurations (768/12/12). Bold values indicate the best score among our three BERT packing/formatting variants (Chunk, Pack-Wrap, Bfd-Split) in each column; note that the official baseline outperforms all variants on every zero-shot metric.

Lang.	Task	Baseline	BERT Bfd-split
ENG	blimp	69.18	59.89
	hellaswag	26.66	25.98
	multiblimp	87.14	65.84
	winogrande	48.80	50.29
	xstorycloze	49.92	46.67
NLD	blimp_nl	79.27	64.57
	hellaswag	26.37	25.60
	multiblimp	91.85	62.63
	winogrande	49.30	50.29
	xcomps_nl	52.68	50.45
	xstorycloze	47.37	45.90
ZHO	blimp	75.44	62.27
	hellaswag	27.05	26.12
	winogrande	49.71	49.05
	xcomps_zh	53.90	52.14
	xstorycloze	50.62	48.84

Table 13: Zero-shot tasks results for BERT.

Lang.	Task	Baseline	BERT Bfd-split
ENG	arc	24.79	24.79
	belebele	22.16	26.14
	bmlama	9.35	9.35
	mnli	51.63	35.59
	sib200	21.50	21.50
	truthfulqa	22.32	27.68
	xnli	46.65	31.55
	avg	28.34	25.23
NLD	arc	24.38	24.38
	belebele	30.11	26.14
	bmlama	10.68	9.52
	include	19.64	34.82
	mnli	43.92	36.54
	sib200	21.50	21.50
	truthfulqa	23.21	27.68
	avg	24.78	25.80
ZHO	arc	25.00	24.79
	belebele	22.73	26.14
	bmlama	9.35	9.35
	include	26.79	26.79
	mnli	45.83	35.59
	sib200	21.50	21.50
	truthfulqa	23.21	27.68
	xnli	43.45	31.55
	avg	27.23	25.42

Table 14: Fine-tuning task results for BERT.

Pairing Original Text with Meaning-Preserving Rewrites for Sample-Efficient Small-Data Language Model Pretraining

Shaoxiang Wu[1] **Alexander Fraser**[1,2,3] **Lukas Edman**[1,2]

[1]School of Computation, Information and Technology, TU Munich
[2]Munich Center for Machine Learning
[3]Munich Data Science Institute

shaoxiang.wu@tum.de, lukas.edman@tum.de

Abstract

We investigate whether augmenting training data with LLM-generated rewrites can improve performance in small-data language model pretraining. We pair each training sentence with a meaning-preserving rewrite, so that masked tokens can be predicted using both the original and its paired version in the same training window. On FineWeb, we test three rewrite types: *simplification* (shorter and easier), *explanation* (longer and easier), and *paraphrasing* (similar length but harder to read). We also test simplification on OpenStax+BCCampus and Simple Wikipedia. We add a reverse masking curriculum: 7 epochs of whole-word masking (WWM), where entire words are masked instead of individual subword tokens, followed by 3 epochs of standard token-level masking (WWM7→Token3). The masking procedure is not rewrite-aware: it does not use alignments between the original sentence and its rewrite. All three FineWeb rewrite types improve EWoK and GLUE-style fine-tuning over source-matched baselines, and simplification improves Entity Tracking across all three sources. The EWoK and GLUE gains concentrate on concept-swap and sentence-pair tasks, suggesting that rewrite pairing especially helps models link meaning across different surface forms. Despite using only one-tenth as much training data, our best model surpasses the official BabyLM 100M GPT-2 baseline on EWoK, Entity Tracking, and GLUE, suggesting that paired rewrite augmentation can improve sample efficiency under a fixed small-data budget.

1 Introduction

Language models need far more data than humans do to acquire comparable linguistic ability (Warstadt and Bowman, 2022): a 13-year-old has encountered fewer than 100 million words, orders of magnitude less than typical LM pretraining corpora. The BabyLM Challenge and Workshop series (Warstadt et al., 2023; Choshen et al., 2024; Charpentier et al., 2025; Choshen et al., 2026) studies this gap by constraining pretraining to a developmentally plausible scale, including 10M words in the 2026 STRICT-SMALL track. Edman et al. (2024) recently showed that pairing original sentences with paraphrase and hard-negative rewrites can substantially improve GLUE-style fine-tuning under this budget, providing early evidence that transforming sentences and arranging related variants together can matter under a fixed small-data budget.

In this work, we investigate **meaning-preserving rewrite pairing** as a form of data augmentation under a fixed word budget. We pair each training sentence with a rewrite so that masked tokens can be predicted from both the original and the paired version. For EWoK and GLUE-style fine-tuning, we find gains across all three FineWeb rewrite types, and simplification replicates this pattern across FineWeb, OpenStax+BCCampus, and Simple Wikipedia. These gains concentrate on EWoK concept swap items and GLUE sentence-pair tasks — both of which require linking meaning across different expressions. This suggests that the clearest benefit is not a uniform improvement across all evaluations, but an improved ability to link meaning across different surface forms. We also find that simplification improves Entity Tracking across the same three sources, although this effect is more specific to simplification than the broader EWoK and GLUE gains. Adding a whole-word masking curriculum (WWM7→Token3: 7 epochs masking whole words, 3 epochs masking individual tokens) further strengthens these gains in most conditions.

We make five contributions: (i) a **meaning-preserving rewrite pairing** pretraining strategy, using simplification, explanation, and harder paraphrasing on FineWeb, and simplification on OpenStax+BCCampus and Simple Wikipedia; (ii) evidence that **EWoK and GLUE gains hold across**

Proceedings of the Second BabyLM Workshop, pages 611–623
October 29, 2026 ©2026 Association for Computational Linguistics

Source[a]	Origin	Expected strength
FineWeb	Web sentences from FineWeb-Edu (Penedo et al., 2024)	Broad science-popular and educational web text, covering health/medicine, biology/nature, geography, history/politics, education, and physical science
OpenStax+BCCampus	University-level open textbooks from OpenStax and BCcampus	Education and humanities
Simple Wikipedia	Simple Wikipedia sentences	Simple expressions, broad but shallow topical coverage

Table 1: Three simplification source corpora.

[a]Source websites: FineWeb-Edu, OpenStax, BCcampus, and Simple Wikipedia.

Original: Enlightenment thinkers proposed that human reason, coupled with empirical study of the physical world, would lead to progress—the advancement of science and the improvement of the human condition.
Simplified: Enlightenment thinkers believed that using reason and studying the world would lead to scientific progress and better living conditions. **Explanation:** Enlightenment thinkers thought that using reason and studying the world could make things better. They believed that reason would help science move forward and improve people's lives. **Paraphrase:** Enlightenment thinkers believed reason and science would lead to progress and a better society.

Table 2: The same FineWeb original rewritten under all three of our rewrite types. Simplification uses simpler wording and a more direct sentence structure; explanation unpacks the content into a more explicit restatement; paraphrase preserves meaning while using different wording and more formal vocabulary, with no instruction to simplify.

three rewrite types on FineWeb and, for simplification, across all three sources; (iii) a more selective **Entity Tracking improvement across three sources of Simplification**, while the other FineWeb rewrite types do not show the same benefit; (iv) a **whole-word masking curriculum** (WWM7→Token3) combined with rewrite pairing, which further strengthens gains in most conditions; and (v) a demonstration of **improved sample efficiency**: our 10M-word model surpasses the official BabyLM 100M GPT-2 baseline on EWoK, Entity Tracking, and GLUE.

2 Data Construction

2.1 Three simplification corpora

We construct sentence-level simplification data from three sources, summarized in Table 1.

For each source, an offline rewrite generator, Qwen2.5-7B-Instruct (Qwen Team, 2024), served with vLLM (Kwon et al., 2023), produces a one-line simplified version of each input sentence, instructed to use simpler vocabulary and shorten the sentence without omitting key information (prompt in Appendix B, Figure 1). We concatenate `original + simplified` for each pair, shuffle with a fixed seed, and truncate to a fixed 10M-word budget. For each source, we construct a raw/original-only 10M corpus and a simplified 10M corpus. This setup is not model-level knowledge distillation: the rewrite generator is used only for offline corpus construction and is absent during pretraining. The pretrained model receives no generator logits, probability distributions, hidden representations, task labels, or online supervision; it is trained from scratch with the standard MLM objective on the resulting fixed-budget corpus. We therefore characterize the intervention as offline text augmentation through meaning-preserving rewriting.

Table 2 contrasts all three of our rewrite types — simplification, explanation, and paraphrase — on the same FineWeb original.

Why sentence-level. We use a 64-token context window for training, so sentence-level pairs usually fit in a single sequence. This means a masked word in one sentence can be predicted using the other sentence in the pair. Paragraph-level pairs would often be too long to fit in one window, reducing this effect.

Simplification. Simplification is our main rewrite type and is applied to all three source corpora. It preserves the original sentence's meaning while making the wording shorter and easier to read, mainly through simpler vocabulary and more direct phrasing. This gives the model two aligned expressions of the same content: the original sentence, which retains the source style and domain wording, and a simplified

Source	N	Ratio	Lex. Ov.	FRE		FKGL		DaleC		DiffWd	
				Orig.	Simp.	Orig.	Simp.	Orig.	Simp.	Orig.	Simp.
FineWeb	163,150	0.9	0.5	43.9	**58.9**	13.1	**9.7**	11.0	**10.0**	0.3	**0.2**
OpenStax+BCCampus	179,165	0.6	0.4	39.6	**50.2**	13.6	**10.1**	10.9	**10.6**	**0.3**	0.3
Simple Wikipedia	331,074	0.8	0.6	52.8	**58.8**	11.1	**8.9**	**10.8**	10.8	**0.3**	0.3

Table 3: Difficulty metrics for original vs. simplified text, computed over each source's pairs (OpenStax and BCCampus pooled together). Ratio and Lex. Ov. compare the simplified sentence to its original; FRE, FKGL, DaleC, and DiffWd are reported separately for the original and the simplified side of each pair. Higher FRE is easier; lower FKGL, DaleC, and DiffWd are easier. Bold marks the easier side for each readability metric.

Rewrite	N	Ratio	Lex. Ov.	FRE		FKGL		DaleC		DiffWd	
				Orig.	Rewr.	Orig.	Rewr.	Orig.	Rewr.	Orig.	Rewr.
Simplification	163,150	0.9	0.5	43.9	**58.9**	13.1	**9.7**	11.0	**10.0**	0.3	**0.2**
Explanation	162,478	1.1	0.4	43.9	**67.7**	13.1	**7.7**	11.0	**9.1**	0.3	**0.2**
Paraphrase	159,932	1.0	0.4	**43.9**	31.4	**13.1**	14.5	**11.0**	12.1	**0.3**	0.4

Table 4: Difficulty metrics for FineWeb original vs. rewrite text, for each of our three rewrite types. Same metric definitions as Table 3. Bold marks the easier side for each readability metric.

version, which exposes the same concepts in a more accessible form. We use this setting to test whether pairing original text with an easier meaning-preserving rewrite improves learning beyond training on the original source text alone.

Explanation. To test whether the benefit requires making text shorter, we also generate *explanation* rewrites on FineWeb. Explanation rewrites are longer than the original (they add context rather than compressing), but still aim to make the text easier to read. If explanation also helps, the benefit is not about shortening but about making the text simpler.

Paraphrasing. To test whether the benefit requires the rewrite to be easier to read, we also generate *paraphrases* on FineWeb. Paraphrases keep the same meaning and similar length, but change the wording and sentence structure. As confirmed in §3, the paraphrases are actually *harder* to read than the originals. If paraphrasing also helps, then the benefit does not require making the text easier — just different.

2.2 Raw source-matched baselines

Each corpus is paired with a source-matched raw Token and WWM curriculum baselines of equal size, which lets us answer: *does any observed gain come from rewriting itself, or from the source domain?*

3 Data Difficulty Analysis

This section checks, using automatic readability metrics, that the rewrites change text difficulty in the direction we claim.

3.1 Automatic metrics

We use six automatic metrics, computed with the `textstat` library[1]: rewrite ratio (rewrite/original word count), lexical overlap (word-set overlap between original and rewrite), Flesch Reading Ease (FRE; higher is easier) and Flesch-Kincaid Grade (FKGL; lower is easier) (Kincaid et al., 1975), and Dale-Chall score and difficult-word ratio (DaleC and DiffWd; both lower is easier) (Chall and Dale, 1995).

3.2 Comparison across the three corpora

We compute these six metrics on both the original and the simplified side of every pair: 163,150 FineWeb pairs, 179,165 OpenStax+BCCampus pairs (the two textbook sources pooled together), and 331,074 Simple Wikipedia pairs (Table 3).

All three sources become shorter after simplification (ratio 0.634–0.893) and keep much of the original vocabulary (overlap 0.38–0.56). FineWeb and OpenStax+BCCampus improve on all four readability metrics. OpenStax+BCCampus is compressed the most but remains the hardest to read after simplification, because its source text is very dense. Simple Wikipedia is a partial exception: two metrics improve, but two move slightly in the wrong direction, likely because the original text is already simple.

[1] https://github.com/textstat/textstat

3.3 Comparison against explanation and paraphrase

We apply the same six metrics to the FineWeb original+explanation and original+paraphrase pairs (§2.1), to see how each rewrite type compares to simplification on the same source (Table 4).

The three rewrite types differ clearly. Explanation is longer than the original (ratio 1.104) but becomes much easier to read (FRE goes from 43.87 to 67.74, the largest gain). Paraphrasing keeps a similar length (ratio 0.954) but becomes *harder* to read on all four metrics (FRE drops from 43.90 to 31.38). This confirms that paraphrases are not just simplified text with a different name. Both explanation and paraphrasing also change the wording more than simplification does (lower lexical overlap: 0.383 and 0.354 vs. 0.505).

4 Main Experiments

4.1 Baselines

We report two families of baselines: (i) global baselines trained on the standard BabyLM-provided corpus (`mlm_regular`, multiple seeds), and (ii) source-matched baselines for each of the three dataset sources (§2.2). Official challenge baselines are taken from the BabyLM 2026 evaluation repository (BabyLM Team, 2026).

4.2 Training and evaluation procedure

All models share the same encoder architecture and training procedure as our BabyLM STRICT-SMALL baseline, following the architecture and optimization setup of Edman and Fraser (2025) but not their adaptive MLM mechanism: a DeBERTa-style encoder (He et al., 2021) (hidden size 384, intermediate size 1280, dropout 0.1), trained with the LAMB optimizer (You et al., 2020), a cosine learning-rate schedule, and a sequence-length curriculum that starts at 64 tokens and increases to 256 tokens partway through training, with batch size scaled inversely to sequence length. We use a 40k-vocabulary SentencePiece BPE tokenizer (Kudo and Richardson, 2018; Sennrich et al., 2016) with byte fallback, trained separately for each data condition on its own training corpus.

For each corpus, we train two masking variants: (i) **Token**, standard token-level MLM with 15% masking throughout, and (ii) **WWM7→Token3**, which uses whole-word masking for the first 7 epochs and then switches to token-level masking for the last 3 epochs. Whole-word masking was introduced in Google's subsequent BERT release (Devlin et al., 2019; Google Research, 2019); our contribution is the 7-epoch WWM to 3-epoch token-level curriculum and its combination with rewrite pairing, rather than WWM itself.

Whole-word masking differs from token masking when a word is split into multiple subword pieces. Because we use SentencePiece BPE, word boundaries are recovered from the tokenizer's word-start marker, shown in tokenizer output as an underscore-like prefix (_). Pieces with this prefix start a new word span, and following pieces without it are treated as part of the same word. For example, if one word is segmented into several pieces, WWM masks the entire span rather than only one piece. With token masking, the model may see part of the word and only need to recover the missing subword; with WWM, all pieces of the selected word are masked together, so the model must recover the full word from context. We chose the 7/3 split as a practical balance: fewer WWM epochs may not be enough to establish word-level context learning, while using only WWM with no token-level phase loses fine-grained subword prediction.

All models are trained for 10 epochs on a fixed 10M-word budget, with 3 random seeds per condition; we report seed means unless noted. Paired bootstrap tests over seeds (10,000 samples) show significant EWoK and GLUE gains, as well as Entity Tracking gains for simplification conditions, over source-matched raw baselines ($p < 0.05$; all but FineWeb Explanation Token EWoK reach $p < 0.01$).

After training, each model is evaluated with the official BabyLM 2026 strict-track evaluation pipeline (BabyLM Team, 2026). BLiMP (Warstadt et al., 2020), BLiMP Supplement (Warstadt et al., 2023), EWoK (Ivanova et al., 2024), COMPS (Misra et al., 2023), Entity Tracking (Kim and Schuster, 2023), and Eye-tracking and Self-Paced Reading (de Varda et al., 2023) are scored via the pipeline's standard perplexity-comparison protocol, and GLUE- and SuperGLUE-style tasks (Wang et al., 2018, 2019) are evaluated by fine-tuning the pretrained encoder with the organizers' default fine-tuning hyperparameters. We report both an aggregate zero-shot average and per-task breakdowns where relevant to our claims (§5, §5.4).

Condition	EWoK	±	Entity	±	GLUE	±
FW Raw Token	51.2	1.2	21.2	0.3	65.1	0.3
FW Simp Token	52.9	0.3	26.1	2.3	68.3	0.1
FW Simp WWM	**55.4**	0.6	**28.9**	0.4	**69.9**	0.1
FW Expl Token	52.8	1.6	22.1	1.6	68.7	1.1
FW Expl WWM	54.7	0.5	22.2	2.0	69.8	1.3
FW Para Token	52.7	0.7	19.0	1.0	68.5	0.3
FW Para WWM	54.2	0.6	19.1	0.6	69.2	0.0
OS+BCC Raw Token	50.5	0.9	20.4	1.0	66.0	0.9
OS+BCC Simp Token	53.7	1.7	25.5	2.5	**69.2**	0.5
OS+BCC Simp WWM	**54.9**	0.7	**25.6**	2.4	69.1	0.3
SW Raw Token	50.2	0.1	20.5	0.4	65.0	0.9
SW Simp Token	51.4	0.4	23.2	1.9	68.7	0.8
SW Simp WWM	**53.4**	1.0	**23.8**	1.8	**69.8**	0.5

Table 5: EWoK, Entity Tracking, and GLUE mean±std over 3 seeds for conditions with full seed replication. The FineWeb block includes simplification, explanation, and paraphrasing. EWoK and GLUE gains, and Entity Tracking gains for simplification rows only, are significant over source-matched raw baselines ($p < 0.05$, paired bootstrap, 10,000 samples).

Source	Raw	Tok.	WWM
physical-dynamics			
FineWeb (Simp.)	50.3	61.4	**69.2**
FineWeb (Expl.)	50.3	58.6	**63.3**
FineWeb (Para.)	50.3	63.3	**66.4**
OpenStax+BCCampus	55.0	**62.8**	61.7
Simple Wikipedia	52.8	60.8	**63.9**
quantitative-properties			
FineWeb (Simp.)	51.4	66.1	**70.4**
FineWeb (Expl.)	51.4	63.8	**66.7**
FineWeb (Para.)	51.4	58.7	**60.9**
OpenStax+BCCampus	53.5	66.2	**69.5**
Simple Wikipedia	52.1	64.8	**68.0**
social-properties			
FineWeb (Simp.)	51.4	55.4	**56.9**
FineWeb (Expl.)	51.4	48.3	**53.2**
FineWeb (Para.)	51.4	51.9	**56.2**
OpenStax+BCCampus	47.7	60.3	**60.9**
Simple Wikipedia	**52.2**	48.1	51.3

Table 6: Mean EWoK accuracy (3 seeds) on physical-dynamics, quantitative-properties, and social-properties, for the Raw, and Token/WWM7→Token3 conditions of each source. For FineWeb, all three rewrite types (Simplification, Explanation, paraphrasing) are shown against the same FineWeb Raw baseline.

5 Results

The improvement is most pronounced on EWoK and GLUE-style fine-tuning, so §5.1 and §5.3 concentrate on these two, before §5.4 asks why. We also observe a narrower but consistent effect on the Entity Tracking task: simplification improves it across all three sources, while the other FineWeb rewrite types do not show the same benefit (§5.2). Table 5 reports mean±std for EWoK, Entity Tracking, and GLUE across all conditions where 3 seeds are available. Across the rewrite-pairing conditions, WWM7→Token3 improves performance in most cases. Since WWM masking alone (without rewrite pairing) does not produce meaningful gains over the raw Token baseline, we omit the Raw WWM conditions from all sub-item tables for brevity; full aggregate scores, including Raw WWM and all zero-shot metrics, are reported by source in Appendix C.

5.1 EWoK results

Simplification improves EWoK over the raw Token baseline in all three sources. Both FineWeb ablations, including paraphrasing, improve EWoK by a comparable margin despite its rewrites being harder, not easier, to read (§3.3); we return to why this may be the case in §5.4.

Physical-dynamics and *quantitative-properties* both improve substantially under simplification for all three sources, making them the clearest source-general component of simplification's EWoK benefit. *Social-properties* tells a different story: it is OpenStax+BCCampus's single largest gain, plausibly reflecting that source's humanities and social science content, yet only gains modestly for FineWeb and regresses for Simple Wikipedia. This suggests simplification's EWoK benefit has both a source-general component (*physical-dynamics*, *quantitative-properties*) and a source-specific one (*social-properties*) tied to a source's particular world-knowledge content.

Our two FineWeb ablations diverge on *quantitative-properties*: both explanation and paraphrasing improve it less than simplification does, paraphrasing least of all. On physical-dynamics, paraphrasing's gain is comparable to simplification's and explanation's, and on social-properties it falls between the other two. No single rewrite type dominates on all three sub-items, but all three produce a gain on *physical-dynamics* and *quantitative-properties*, consistent with these being the most source-general and rewrite-type-general component of the EWoK benefit (§5.4).

5.2 Entity Tracking results

The Entity Tracking task shows a more selective pattern than EWoK. Simplification improves Entity Tracking over source-matched raw Token baselines for all three sources.

Unlike EWoK and GLUE, this gain does not generalize to all meaning-preserving rewrite types. On FineWeb, explanation gives only a small improvement, while paraphrasing falls below the raw Token baseline. Thus, the Entity Tracking result appears to be tied specifically to simplification. This contrasts with EWoK and GLUE, where simplification, explanation, and paraphrasing all improve performance.

5.3 GLUE results

As shown in Table 5, simplification substantially improves GLUE over the source-matched raw baselines for all three sources. This pattern is not limited to simplification: explanation and paraphrasing also improve GLUE by similar margins, even though their rewrites are not uniformly easier to read.

Table 7 breaks this down by sub-task (full breakdown in Appendix F) for the five that gain consistently across all three sources and all three FineWeb rewrite types.

MNLI, MRPC, and QQP gain the most consistently across all three sources and both FineWeb rewrite types: all three require relating two given sentences rather than answering questions about a single passage. RTE gains as well under both rewrite types, though more modestly for paraphrasing. BoolQ and WSC move inconsistently; MultiRC gains modestly under simplification but little under paraphrasing. The sentence-pair advantage holds for paraphrasing as well as simplification, consistent with the account in §5.4.

Source	Raw	Tok.	WWM
MNLI			
FineWeb (Simp.)	51.2	59.4	**62.8**
FineWeb (Expl.)	51.2	59.4	**61.1**
FineWeb (Para.)	51.2	58.7	**60.9**
OpenStax+BCCampus	51.4	60.2	**61.8**
Simple Wikipedia	52.1	59.5	**62.5**
MRPC (F1)			
FineWeb (Simp.)	82.4	87.2	**88.6**
FineWeb (Expl.)	82.4	87.0	**88.7**
FineWeb (Para.)	82.4	86.6	**88.6**
OpenStax+BCCampus	82.6	87.2	**88.5**
Simple Wikipedia	82.2	86.2	**88.8**
QQP (F1)			
FineWeb (Simp.)	65.9	71.7	**73.3**
FineWeb (Expl.)	65.9	71.7	**72.9**
FineWeb (Para.)	65.9	71.8	**72.5**
OpenStax+BCCampus	67.3	72.1	**72.5**
Simple Wikipedia	66.1	71.4	**73.1**
RTE			
FineWeb (Simp.)	59.0	63.6	**65.7**
FineWeb (Expl.)	59.0	64.3	**66.4**
FineWeb (Para.)	59.0	61.2	**63.8**
OpenStax+BCCampus	62.1	**65.5**	62.4
Simple Wikipedia	57.3	65.5	**66.9**
MultiRC			
FineWeb (Simp.)	66.0	**67.1**	67.1
FineWeb (Expl.)	66.0	65.9	**67.4**
FineWeb (Para.)	66.0	66.3	**66.5**
OpenStax+BCCampus	66.8	**67.4**	67.0
Simple Wikipedia	64.2	66.8	**67.0**

Table 7: Mean fine-tuning score (3 seeds) on MNLI, MRPC, QQP, RTE, and MultiRC, for the Raw, and Token/WWM7→Token3 conditions of each source, with FineWeb's explanation and paraphrasing rewrites shown alongside simplification. MRPC and QQP are scored with F1; the remaining tasks with accuracy.

5.4 Mechanisms behind the main gains

EWoK: Improved context-to-concept mapping, shared across rewrite types. EWoK (Ivanova et al., 2024) tests world knowledge with minimal pairs scored by perplexity. In *concept swap* items, context and target jointly change concept (e.g., "increasing"/"accelerating" vs. "decreasing"/"slowing down"), requiring the model to map context onto the right concept word. In *variable swap* items, concept words stay fixed and only entity role assignment changes — a different capacity. Our hypothesis concerns concept swap items only. The three specific EWoK sub-items reported in Table 6 — *physical-dynamics*, *quantitative-properties*, and *social-properties* — contain only concept swap items in the evaluation data and therefore fall directly within this analysis.

Under MLM, a masked word in one sentence can be predicted using the other sentence in the pair, even though the two sentences use different words to express the same meaning. This may train the model to associate a given context with a range of words that share the same meaning, rather than always predicting the same word it has seen before.

The observed pattern offers some support for this explanation. Across all rewrite types and sources, Table 8 shows gains on concept swap items (which test whether the model picks the right meaning given context), while variable swap items (which test something different) stay flat near chance. The gains are already present in the Token-only rewrite conditions and are further strengthened by WWM7→Token3, whereas applying WWM7→Token3 to raw data does not produce consistent aggregate gains over Raw Token (Appendix C). This selective pattern is consistent with rewrite pairing improving context-to-concept map-

Condition	Raw	Tok.	WWM
concept swap			
FineWeb (Simp.)	51.4	52.3	**53.7**
FineWeb (Expl.)	51.4	52.9	**54.0**
FineWeb (Para.)	51.4	51.7	**52.8**
OpenStax+BCCampus	49.9	52.6	**53.8**
Simple Wikipedia	49.7	50.1	**51.8**
variable swap			
FineWeb (Simp.)	**50.4**	49.7	50.0
FineWeb (Expl.)	**50.4**	49.9	50.0
FineWeb (Para.)	50.4	**50.5**	49.9
OpenStax+BCCampus	49.9	**50.3**	50.2
Simple Wikipedia	49.8	50.2	**50.5**

Table 8: Mean EWoK accuracy (3 seeds) on concept swap vs. variable swap items, for the Raw, and Token/WWM7→Token3 conditions of each source, with FineWeb's explanation and paraphrasing rewrites shown alongside simplification. Concept swap items test context-to-concept mapping; variable swap items test entity role binding, holding the concept words fixed.

ping rather than producing a broad improvement on all EWoK items. We treat this interpretation as a post-hoc hypothesis supported by the observed decomposition, not as a demonstrated causal mechanism.

This pattern holds for all three rewrite types — including paraphrasing, whose rewrites are harder to read, not easier. This suggests that what matters is not making the text simpler, but pairing it with a different version that expresses the same meaning.

GLUE: Improved performance from rewritten data. The same effect may explain why gains on GLUE concentrate on sentence-pair tasks (MNLI, MRPC, QQP, RTE) — tasks where the model must compare two sentences — rather than single-passage tasks. Training with paired rewrites may generally improve the model's ability to link meaning across different ways of saying the same thing. Edman et al. (2024) also found large GLUE improvements by training with meaning-related rewrites, though their setup (original + paraphrase + hard-negative triplets) differs from ours and the underlying mechanism may not be the same.

Entity Tracking: Simplification may reduce local referential complexity. The Entity Tracking gain is more selective than the EWoK and GLUE gains, because it appears mainly for simplification. A plausible explanation is that simplified rewrites make entity mentions and their local contexts easier to encode by replacing dense noun phrases, embedded clauses, and domain-specific wording with more direct expressions, while preserving the same referential structure. This gives the model a simpler second view of the same entities and relations; explanation and paraphrasing also preserve meaning, but they do not consistently simplify this local entity context.

WWM: Harder predictions may encourage contextual learning. The WWM curriculum is a separate masking factor rather than a rewrite-aware mechanism. By masking all subword pieces of a selected word, WWM removes partial-word shortcuts and makes prediction harder: the model must recover the full word from its surrounding context rather than from visible fragments of the same word. This may encourage the model to use broader contextual relations during pretraining, although the lower BLiMP scores suggest a possible tradeoff with grammar-focused benchmarks.

Overall, the EWoK and GLUE gains are consistent with the model learning to rely more on meaning and context, and less on memorized word patterns, while the Entity Tracking gain may reflect simplification's more direct effect on local entity contexts. Future work could test these hypotheses through probing or causal analysis.

5.5 Submission

Our main submission focuses on the STRICT-SMALL track (10M words), where we submit a model trained with WWM7→Token3 simplification pairing.[2] Table 9 compares this model with the official BabyLM 2026 GPT-2 and interaction baselines (BabyLM Team, 2026) from both the STRICT-SMALL and STRICT tracks. The most pronounced gains are on EWoK, Entity Tracking, and (Super)GLUE. Notably, our 10M-word model reaches an NLP average of 52.97, exceeding the STRICT-SMALL baselines and even the 100M-word STRICT interaction baseline (52.42). It also surpasses the official 100M-word GPT-2 baseline on EWoK (56.07 vs. 54.37), Entity Tracking (28.45 vs. 16.91), and (Super)GLUE (69.79 vs. 67.75). These results show that better downstream performance can be achieved with one-tenth as much training data, demonstrating a substantial improvement in sample efficiency.

We also make one STRICT-track attempt using span masking. Instead of masking indepen-

[2]Resources: code; STRICT-SMALL WWM simplification model; corresponding FineWeb simplification-pair training data; and additional STRICT span-masking model.

dently selected tokens or whole words, span masking masks contiguous token spans whose lengths are sampled from a geometric distribution (Joshi et al., 2020). This model is trained on a 100M-word simplification-pair mixture from FineWeb, OpenStax, and BCcampus. Although this setting is not our main ablation, it also improves strongly over the official 100M-word GPT2 baseline on EWoK (57.73 vs. 54.37) and (Super)GLUE (72.60 vs. 67.75), and slightly improves the NLP average (53.10 vs. 53.03).

6 Conclusion

Our results suggest that the rewrite direction does not matter for our two main outcomes: simplification, explanation, and harder paraphrasing all produce consistent gains on EWoK and GLUE-style fine-tuning, indicating that what matters there is semantic-preserving variation itself rather than any particular transformation direction. Simplification also improves Entity Tracking across all three sources, but this effect is more selective: the explanation condition gives only a small gain, and the paraphrase condition does not improve it. Across these outcomes, WWM7→Token3 further strengthens the gains in most conditions. Ablations across corpora of different domain, rewrite difficulty, and rewrite length consistently reproduce the EWoK and GLUE pattern, confirming the robustness of the approach while showing that different evaluation tasks can respond to different rewrite properties.

The Ewok and Glue gains are selective in ways consistent with a single underlying shift toward contextual reliance over surface statistics. On EWoK, the improvement concentrates on concept swap items and is absent on variable swap items, a pattern that holds across all three rewrite types and all three sources. On GLUE, the gain concentrates on sentence-pair tasks (MNLI, MRPC, QQP, RTE), not passage-based ones. This selectivity is consistent with the hypothesis that rewrite pairing encourages the model to exploit contextual meaning rather than memorized surface form, with the particular rewrite type shaping which capacities benefit most. Edman et al. (2024) also found large GLUE improvements from a different meaning-rewrite setup, though the mechanism may differ.

The key practical implication is a substantial improvement in sample efficiency: our 10M-word strict-small model surpasses the official 100M-word GPT-2 baseline on EWoK, Entity Tracking, and GLUE, suggesting that better organization of limited training data may be as important as increasing its quantity. More broadly, our findings suggest that improving the structure of training data can serve as a complementary scaling axis for small-data language model pretraining — alongside model size, data volume, and compute — one that may be particularly valuable when data is the binding constraint. We leave the design of other targeted data transformations to future work.

Limitations

All three source corpora used in this work consist of factual, expository text — web-based educational content, open textbooks, and encyclopedic articles. It remains an open question whether meaning-preserving rewrite pairing produces similar gains on other text types, such as conversational or spoken language, fictional narratives, or informal social media text. In such genres, sentences may be shorter, more context-dependent, or less self-contained, which could reduce the effectiveness of sentence-level pairing.

We also do not test all rewrite types on all sources. Explanation and harder paraphrasing are evaluated only on FineWeb, while OpenStax+BCCampus and Simple Wikipedia are tested only with simplification. Therefore, our cross-source evidence is strongest for simplification; whether explanation and harder paraphrasing generalize similarly across sources remains open.

Ethics Statement

Our simplification, explanation and paraphrase corpora are generated by prompting an offline rewrite generator (Qwen2.5-7B-Instruct) to rewrite existing text; this introduces stylistic and content biases specific to that generator into the resulting training data, which may not generalize to rewrites produced by other models or pipelines. Our source corpora (FineWeb, OpenStax, BCCampus, Simple Wikipedia) are all openly licensed for research use, and we do not redistribute the generated rewrites beyond what is needed to reproduce our results. All experiments are conducted on English-language data under the BabyLM STRICT-SMALL track and do not involve human subjects.

Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the Er-

Model	Track	Overall	NLP	Human	BLiMP	Supp.	EWoK	Entity	COMPS	GPIQA	GLUE	Read.
Ours (WWM simp.)	strict-small	41.80	52.97	2.71	67.20	56.01	56.07	**28.45**	53.57	**39.67**	69.79	5.42
Ours (span mask. simp.)	strict	**41.93**	**53.10**	**2.83**	72.38	59.86	**57.73**	18.43	53.98	36.69	**72.60**	5.65
GPT2 baseline	strict-small	37.38	48.99	-3.26	65.23	57.25	50.63	19.10	51.81	35.09	63.80	5.63
Interaction baseline	strict-small	38.71	49.04	2.55	63.07	58.13	51.45	19.62	51.29	36.14	63.62	5.10
GPT2 baseline	strict	40.73	53.03	-2.32	**74.73**	65.00	54.37	16.91	**55.85**	36.62	67.75	**6.93**
Interaction baseline	strict	39.89	52.42	-3.96	72.89	**65.09**	54.23	15.89	55.37	36.18	67.28	3.43

Table 9: Leaderboard submission results for our STRICT-SMALL model, our additional STRICT span-masking attempt, and official BabyLM 2026 baselines. Overall, NLP, and Human are leaderboard aggregate scores; Reading is the mean of Eye-tracking and SPR. GPIQA denotes GlobalPIQA. Bold indicates the best score in each column. Our 10M-word STRICT-SMALL model exceeds the 100M-word STRICT interaction baseline in NLP average and exceeds the 100M-word GPT2 baseline on EWoK, Entity Tracking, and (Super)GLUE; our STRICT span-masking model further improves EWoK and (Super)GLUE.

langen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project b279bb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

BabyLM Team. 2026. BabyLM Evaluation 2026: Official evaluation pipeline and baselines. GitHub repository.

Jeanne Sternlicht Chall and Edgar Dale. 1995. *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.

Tyler A. Chang, Catherine Arnett, Abdelrahman Sadallah, and 1 others. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *arXiv preprint arXiv:2510.24081*.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. arXiv preprint arXiv:2602.20092.

Leshem Choshen, Ryan Cotterell, Michael Y Hu, Tal Linzen, Aaron Mueller, Candace Ross, Alex Warstadt, Ethan Wilcox, Adina Williams, and Chengxu Zhuang. 2024. [call for papers] the 2nd babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus. arXiv preprint arXiv:2404.06214.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2023. Cloze probability, predictability ratings, and computational estimates for 205 english sentences, aligned with existing eeg and reading time data. *Behavior Research Methods*, pages 1–24.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Lukas Edman, Lisa Bylinina, Faeze Ghorbanpour, and Alexander Fraser. 2024. Are babylms second language learners? In *Proceedings of the 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 166–173.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Google Research. 2019. BERT: Whole word masking models. Official BERT repository release, May 31, 2019.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

Anna Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H Clark, Carina Kauf, Jennifer Hu, Pramod RT, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyurek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Josh Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (ewok): A cognition-inspired framework for evaluating basic world knowledge in language models. *arXiv preprint arXiv:2405.09665*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. *Naval Technical Training Command Research Branch Report*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Yu, Joseph E Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. Comps: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. arXiv preprint arXiv:2406.17557.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt and Samuel R Bowman. 2022. What artificial neural networks can tell us about human language acquisition. *Algebraic Structures in Natural Language*, pages 17–60.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations*.

A GlobalPIQA Hidden Task Analysis

GlobalPIQA (Chang et al., 2025) was released as a hidden evaluation task late in the BabyLM evaluation cycle. Table 10 reports the 3-seed mean and standard deviation of the GlobalPIQA average score, computed as the simple average of the parallel and non-parallel splits.

Overall, the GlobalPIQA results show relatively high variance, making it difficult to draw a firm conclusion from individual differences. Most of our source-specific conditions have higher mean scores than the official GPT2 and interaction baselines in Table 9, whose GlobalPIQA scores range from 35.09 to 36.62. At the same time, the BB26 Raw Token baseline reaches 39.08 in one seed despite being trained on the official BB26 strict-small data rather than on any of our constructed corpora, suggesting that the task may be sensitive to random seed or evaluation instability. We therefore remain

Source	Condition	GlobalPIQA avg.
BabyLM	Raw Tok.	36.93 ± 2.00
FineWeb	Raw Tok.	37.34 ± 1.59
FineWeb	Raw WWM	**40.79 ± 2.22**
FineWeb	Simp. Tok.	37.16 ± 1.32
FineWeb	Simp. WWM	38.66 ± 2.59
FineWeb	Expl. Tok.	39.29 ± 2.01
FineWeb	Expl. WWM	39.28 ± 3.12
FineWeb	Para. Tok.	39.41 ± 3.21
FineWeb	Para. WWM	38.77 ± 2.07
OS+BCC	Raw Tok.	38.04 ± 2.58
OS+BCC	Raw WWM	39.59 ± 1.44
OS+BCC	Simp. Tok.	40.39 ± 1.63
OS+BCC	Simp. WWM	38.45 ± 0.55
SimpleWiki	Raw Tok.	38.31 ± 1.03
SimpleWiki	Raw WWM	37.49 ± 1.99
SimpleWiki	Simp. Tok.	35.99 ± 2.50
SimpleWiki	Simp. WWM	34.67 ± 2.27

Table 10: GlobalPIQA hidden-task results for the main ablations. Scores are 3-seed mean±standard deviation. The BB26 Raw Token baseline uses our DeBERTa-style architecture trained on the official BabyLM STRICT-SMALL data.

cautious about interpreting GlobalPIQA as clear evidence for any single intervention. At most, the pattern suggests that our three source corpora may be mildly favorable for this task because they are educational or knowledge-oriented texts, but this remains a hypothesis rather than a stable finding.

B Generation Prompts

We give the system prompts used to generate our simplification, explanation, and paraphrase corpora (§2) with the offline rewrite generator (Qwen2.5-7B-Instruct).

```
You are simplifying academic sentences
for a language model training dataset.
For each input sentence, output exactly
one line: a simplified version.

Simplification rules:
- Keep the core meaning intact.
- Use simpler, more common vocabulary.
  Replace jargon with plain words.
- Shorten where possible, but do not
  omit key information.
- Write in a clear, direct style
  accessible to a general audience.
- Do NOT add explanations, labels, or
  meta-commentary.
- Output exactly one line of text,
  nothing else.
```

Figure 1: System prompt used for simplification generation.

```
You are rewriting text passages so that
a middle school student (age 12-14) can
easily understand them.

Rewriting rules:
- Keep ALL the information and key
  details from the original.
- Replace difficult vocabulary and
  technical jargon with common words.
- If a key term must be kept, briefly
  explain what it means in plain
  language.
- Break up long or complicated
  sentences into shorter, clearer ones.
- The output should be roughly the
  same length as the input.
- Write in a clear, natural tone.
- Do NOT add explanations of what you
  are doing, labels, headers, or
  commentary.
- Output only the rewritten text,
  nothing else.
```

Figure 2: System prompt used for explanation generation.

```
You are paraphrasing web text for a
language model training dataset.
For each input passage, output exactly
one line: a paraphrase.

Paraphrase rules:
- Preserve the full meaning. Change
  structure and vocabulary
  significantly.
- Match the original register and
  tense.
- Match the original length as closely
  as possible -- do not compress or
  expand.
- Do NOT add explanations, labels, or
  meta-commentary.
- Output exactly one line of text,
  nothing else.
```

Figure 3: System prompt used for paraphrasing's paraphrase generation.

C Full Aggregate Results

Tables 11–13 report the full aggregate zero-shot, hidden-task, and fine-tuning results, split by source. Raw WWM refers to training on the original (non-rewritten) corpus with WWM7→Token3 masking. Reading is the mean of Eye-tracking and Self-Paced Reading scores. GlobalPIQA is the average of the parallel and non-parallel splits. GLUE is the mean fine-tuning score across the seven GLUE-style tasks.

	Global	FineWeb Raw/Simp.			
Metric	Base (bb26)	Raw Tok.	Raw WWM	Simp. Tok.	Simp. WWM
BLiMP	66.2	67.1	**67.9**	65.7	67.6
Supplement	**58.4**	54.8	55.0	54.1	54.9
EWoK	51.0	51.2	51.4	52.9	**55.4**
Ent. Tracking	20.8	21.2	20.5	26.1	**28.9**
COMPS	51.3	52.4	**52.9**	52.0	52.8
Reading	**6.2**	5.1	5.2	4.7	5.6
GlobalPIQA	36.9	37.3	**40.8**	37.2	38.7
GLUE	64.4	65.1	64.9	68.3	**69.9**

	Global	FineWeb Additional Rewrites				
Metric	Base (bb26)	Raw Tok.	Expl. Tok.	Expl. WWM	Para. Tok.	Para. WWM
BLiMP	66.2	67.1	68.4	**68.6**	67.2	67.2
Supplement	**58.4**	54.8	56.5	55.5	53.6	54.0
EWoK	51.0	51.2	52.8	**54.7**	52.7	54.2
Ent. Tracking	20.8	21.2	22.1	**22.2**	19.0	19.1
COMPS	51.3	52.4	52.5	**53.0**	51.9	52.1
Reading	**6.2**	5.1	5.7	5.5	5.4	4.6
GlobalPIQA	36.9	37.3	39.3	39.3	**39.4**	38.8
GLUE	64.4	65.1	68.7	**69.8**	68.5	69.2

Table 11: Full aggregate results for FineWeb conditions, mean over 3 seeds. The global baseline is `mlm_regular_bb26_40k`, trained on the standard BabyLM-provided corpus. Entity Tracking uses the updated hidden-task evaluation, and GlobalPIQA is the hidden-task split average. FineWeb explanation and paraphrasing variants have no separate raw condition; both compare to the FineWeb Raw baseline.

Metric	Global	OpenStax+BCCampus			
	Baseline (bb26)	Raw Tok.	Raw WWM	Simp. Tok.	Simp. WWM
BLiMP	66.2	67.9	**68.1**	67.2	67.7
Supplement	**58.4**	55.2	54.6	56.9	57.5
EWoK	51.0	50.5	50.9	53.7	**54.9**
Ent. Tracking	20.8	20.4	21.4	25.5	**25.6**
COMPS	51.3	51.0	50.9	51.0	**51.4**
Reading	**6.2**	5.7	5.7	5.0	6.1
GlobalPIQA	36.9	38.0	39.6	**40.4**	38.5
GLUE	64.4	66.0	65.6	**69.2**	69.1

Table 12: Full aggregate results for OpenStax+BCCampus conditions, mean over 3 seeds. The global baseline is `mlm_regular_bb26_40k`. Entity Tracking uses the updated hidden-task evaluation, and GlobalPIQA is the hidden-task split average.

Metric	Global	Simple Wikipedia			
	Baseline (bb26)	Raw Tok.	Raw WWM	Simp. Tok.	Simp. WWM
BLiMP	66.2	64.8	**66.3**	66.1	66.2
Supplement	**58.4**	47.5	46.8	48.2	49.0
EWoK	51.0	50.2	50.0	51.4	**53.4**
Ent. Tracking	20.8	20.5	20.0	23.2	**23.8**
COMPS	51.3	52.1	**53.1**	52.0	52.5
Reading	**6.2**	5.3	4.9	4.7	5.3
GlobalPIQA	36.9	**38.3**	37.5	36.0	34.7
GLUE	64.4	65.0	65.1	68.7	**69.8**

Table 13: Full aggregate results for Simple Wikipedia conditions, mean over 3 seeds. The global baseline is `mlm_regular_bb26_40k`. Entity Tracking uses the updated hidden-task evaluation, and GlobalPIQA is the hidden-task split average.

D Hyperparameters

Table 14 lists the training hyperparameters for both submissions.

Parameter	Strict-small (10M)	Strict (100M)
Model		
Architecture	DeBERTa-v2	DeBERTa-v2
Hidden size	384	768
Intermediate size	1280	2560
Dropout	0.1	0.1
Vocabulary size	40k	40k
Training		
Sequence length	64→256	64→256
Batch size (tokens)	16,384	131,072
Learning rate	7e-3	7e-3
Epochs	10	10
Scheduler	Cosine	Cosine
Mask ratio	0.15	0.15
Random ratio	0.1	0.1
Keep ratio	0.1	0.1
Weight decay	0.01	0.01
Optimizer	LAMB	LAMB
Optimizer ϵ	1e-8	1e-8
Optimizer β_1	0.9	0.9
Optimizer β_2	0.95	0.95
Gradient clipping	1.0	1.0
Masking strategy	WWM7→Token3	Span geo.
Span geo. p	—	0.5

Table 14: Training hyperparameters for the two submissions. Batch size in tokens = sequences × sequence length. The strict-small model uses WWM7→Token3 simplification pairing; the strict model uses span geometric masking (Joshi et al., 2020) with p=0.5.

E Full EWoK Sub-item Breakdown

Table 15 is split into raw/simplification conditions and a continued panel for FineWeb explanation/paraphrasing.

Sub-item	FineWeb			OS+BCC			SimpleWiki		
	Raw	Tok.	WWM	Raw	Tok.	WWM	Raw	Tok.	WWM
social-int.	55.9	53.5	57.4	55.7	56.6	**60.8**	49.3	51.7	55.3
material-dyn.	**55.8**	48.4	54.7	47.9	51.0	53.7	49.7	42.5	47.6
physical-rel.	51.2	51.0	50.5	49.4	50.9	49.4	47.5	52.8	51.6
social-rel.	49.6	50.7	50.3	49.8	50.5	51.4	50.3	51.2	50.1
physical-dyn.	50.3	61.4	**69.2**	55.0	62.8	61.7	52.8	60.8	63.9
physical-int.	50.5	52.3	**53.9**	50.5	52.2	53.7	50.7	49.5	52.0
material-prop.	48.0	47.8	51.8	48.0	43.7	48.4	50.6	50.0	52.2
social-prop.	51.4	55.4	56.9	47.7	60.3	**60.9**	52.2	48.1	51.3
agent-prop.	50.5	50.9	50.3	50.1	50.4	50.5	49.6	50.0	51.1
quant.-prop.	51.4	66.1	**70.4**	53.5	66.2	69.5	52.1	64.8	68.0
spatial-rel.	**48.7**	44.2	43.8	48.4	45.9	44.0	48.0	44.3	44.1
AVG	51.2	52.9	**55.4**	50.5	53.7	54.9	50.2	51.4	53.4

Table 15: Full EWoK sub-item breakdown for raw and simplified conditions (UID accuracy, mean over 3 seeds).

F Full GLUE Sub-task Breakdown

Table 16 is split into raw/simplification conditions and a continued panel for FineWeb explanation/paraphrasing.

	FineWeb Additional Rewrites				
Sub-item	Raw	E.Tok.	E.WWM	P.Tok.	P.WWM
social-int.	55.9	56.7	58.3	54.5	59.9
material-dyn.	**55.8**	53.5	52.1	49.0	52.2
physical-rel.	51.2	51.5	52.5	**53.8**	52.0
social-rel.	49.6	51.3	**51.8**	50.9	50.3
physical-dyn.	50.3	58.6	63.3	63.3	**66.4**
physical-int.	50.5	50.8	52.8	50.6	49.3
material-prop.	48.0	50.6	53.3	52.2	**53.5**
social-prop.	51.4	48.3	53.1	51.9	56.2
agent-prop.	50.5	**51.3**	51.0	50.9	50.6
quant.-prop.	51.4	63.8	**66.7**	58.7	60.9
spatial-rel.	**48.7**	44.5	46.7	43.9	44.4
AVG	51.2	52.8	**54.7**	52.7	54.2

Table 15: Full EWoK sub-item breakdown for FineWeb explanation and paraphrasing variants. These variants have no separate raw condition; both compare to FineWeb Raw.

	FineWeb			OS+BCC			SimpleWiki		
Task	Raw	Tok.	WWM	Raw	Tok.	WWM	Raw	Tok.	WWM
MNLI	51.2	59.4	**62.8**	51.4	60.2	61.8	52.1	59.5	62.5
MRPC	82.4	87.2	88.6	82.6	87.2	88.5	82.2	86.2	**88.8**
QQP	65.9	71.7	**73.3**	67.3	72.1	72.5	66.1	71.4	73.1
RTE	59.0	63.6	65.7	62.1	65.5	62.4	57.3	65.5	**66.9**
BoolQ	68.0	66.5	68.0	67.3	66.1	67.0	68.0	67.0	67.7
MultiRC	66.0	67.1	67.1	66.8	**67.4**	67.0	64.2	66.8	67.0
WSC	63.5	62.8	63.5	64.1	66.0	64.7	65.4	64.7	62.8
AVG	65.1	68.3	**69.9**	66.0	69.2	69.1	65.0	68.7	69.8

Table 16: Full fine-tuning sub-task breakdown for raw and simplified conditions, mean over 3 seeds. MRPC and QQP are scored with F1; the remaining tasks are scored with accuracy.

	FineWeb Additional Rewrites				
Task	Raw	E.Tok.	E.WWM	P.Tok.	P.WWM
MNLI	51.2	59.4	**61.1**	58.7	60.9
MRPC	82.4	87.0	**88.7**	86.6	88.6
QQP	65.9	71.7	**72.9**	71.8	72.5
RTE	59.0	64.3	**66.4**	61.2	63.8
BoolQ	68.0	67.3	68.3	67.5	**68.7**
MultiRC	66.0	65.9	**67.4**	66.3	66.5
WSC	63.5	65.4	64.1	**67.3**	63.5
AVG	65.1	68.7	**69.8**	68.5	69.2

Table 16: Full fine-tuning sub-task breakdown for FineWeb explanation and paraphrasing variants. These variants have no separate raw condition; both compare to FineWeb Raw.

Qwenfant: Train Big Character-level BabyLMs on Concise Parallel Text

Lukas Edman[1,2] **Alexander Fraser**[1,2,3]

[1]School of Computation, Information and Technology, TU Munich
[2]Munich Center for Machine Learning
[3]Munich Data Science Institute

lukas.edman@tum.de

Abstract

This work concerns our submission for the MULTILINGUAL track of the 2026 BabyLM Challenge.[1] We improve on our baseline using data that has been summarized and translated by an LLM, showing that higher-quality data can come from synthetic sources. We further improve with 2 simple changes: using a larger model, and using a character-level vocabulary. Both of these changes require LoRA-style fine-tuning for realizing their full potential. We dub our final Qwen3-based model Qwenfant, which, at the time of submission, ranks 1st in the MULTILINGUAL track.

1 Introduction

Language Models (LMs) are generally considered to be much less data-efficient than humans, requiring orders of magnitude more data to achieve similar performance. However the nature of the data seen by humans is vastly different from that typically seen by these LMs. The majority of LM training data is mined from the web, with little to no regard to its lexical complexity or even grammaticality.

BabyLM and BabyBabelLM have introduced datasets that include training data that would be more likely for a child to encounter, such as child-directed speech, however participants have found that some of this data degrades performance on standard benchmarks (Padovani et al., 2025).

Optimizing a dataset through selection is a difficult, if not intractable, task. Instead, we propose a simpler method: use an existing pretrained Large Language Model (LLM) to improve the conciseness of the data. Pretrained LLMs have remarkable ability to summarize text, removing unimportant information such as filler words, or overly verbose descriptions. LLMs are also adept at translating between languages, allowing us to generate high-quality synthetic parallel training data. We experiment with both summarizing and translating the BabyLM Challenge's provided English data to Dutch and Chinese, the other two languages involved in the MULTILINGUAL track.

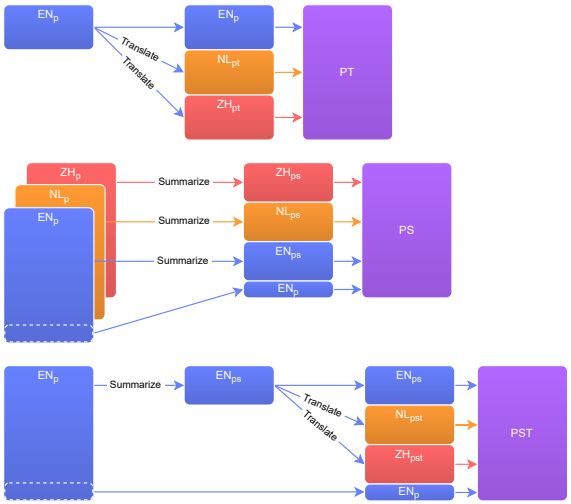


Figure 1: Preprocessing pipelines we investigate, including packed-translated (*PT*), packed-summarized (*PS*), and packed-summarized-translated (*PST*). Packing process not shown for brevity.

In this work, we find that:

- Using 3 straightforward preprocessing steps, packing, summarization, and translation, provides a large performance improvement across all languages.
- Using a larger model is beneficial, providing more capacity for each language, but only when using LoRA for fine-tuning to prevent overfitting.
- Using a character-level tokenization also improves results in larger models, when using LoRA.

[1]https://github.com/Leukas/babylm26

Proceedings of the Second BabyLM Workshop, pages 624–630
October 29, 2026 ©2026 Association for Computational Linguistics

These three improvements culminate in an 11-point increase over the baseline, marking a substantially higher quality multilingual BabyLM.

2 Related Work

2.1 Multilingual Pretraining

Multilingual pretraining is typically approached with a similar strategy as monolingual pretraining: training an LM with masked or causal language modeling, with the only major change being an increased vocabulary size to better accomodate the diverse character sets. This was adopted as the standard since the release of mBERT (Devlin et al., 2019), and has remained effectively the same to this day. One of the latest examples of a performant multilingual model includes Qwen 3 (Yang et al., 2025), which uses this same pretraining strategy, only with some architectural optimizations and adding a reasoning fine-tuning step.

2.2 Synthetic Training Data

A number of works have investigated the use of synthetic pretraining data for training models, both in and outside the BabyLM context. Some BabyLM submissions have tried to generate synthetic data on-the-fly, with limited success (Ulm et al., 2025; AlKhamissi et al., 2024). Theodoropoulos et al. (2024) and Poh et al. (2025) generated synthetic data (bedtime stories and child-directed speech, respectively), finding this data negatively impacted overall performance. Edman et al. (2024a) used a synthetic dataset that includes sentences and contrastive pairs, and showed improved results on GLUE tasks with a similar structure such as MRPC.

Outside of BabyLM, Eldan and Li (2023) created TinyStories, a synthetic dataset mimicking stories for 3-4 year olds, and found that small language models trained on this could generate fluent outputs with a limited vocabulary. Synthetically-generated textbooks have also been successfully used to pretrain a large-scale LLM (Gunasekar et al., 2023). Kang et al. (2025) built upon this and showed that synthetic data alone does not show an improvement in training efficiency, however a mixture of real and synthetic data does.

2.3 Tokenization

Character-level models have been extensively explored, and have been shown to be more sample-efficient in training, despite their higher training cost overall due to longer sequence lengths (Xue et al., 2022). They have been shown to be more performant on tasks requiring character-level manipulation (Pagnoni et al., 2025), and similar-language translation (Edman et al., 2024b).

In the context of BabyLM, character-level models have been seldom explored. Edman and Bylinina (2023) attempted using a character-level tokenizer without much success, and Edman and Fraser (2025) used character-level information to inform an n-hot encoding, which showed improved adjective nominalization ability.

3 Method

We break down our method into the three improvements that we make over the baseline: preprocessing, tokenization, and model size.

3.1 Preprocessing

Our preprocessing involves 3 steps: packing, summarizing, and translation. For the latter two steps, we use `Qwen2.5-3B-Instruct` (Qwen Team, 2024)[2]. We show the prompts used in Figure 2 in Appendix B.

Packing Packing is simply done by concatenating examples together until they form a desired context length. We use a context length of 512 words. This is primarily done because the structure of examples in the datasets provided by BabyLM does not preserve document boundaries, so for most examples, the surrounding examples are from a related context.

Summarization Summarization is done with the purpose of improving the conciseness of the dataset. Some of the data, particularly that which is transcribed speech, has several filler words or phrases (e.g., 'uh', 'you know'), or repetitions that do not add semantic value. While some of these are easily detectable with simple rules, others are more difficult to replace without understanding the context. We therefore use an LLM to shorten the text.

Following the packing step, we feed the packed examples into the model, asking it to shorten the text to 200 words or less. We note that in our prompt we specify to 'shorten' the text, rather than use the word 'summarize'. This is because for chunks of dialogue, a summary may imply a 3rd-person overview of the conversation. We wish to preserve the dialogue format.

[2] `https://hf.co/Qwen/Qwen2.5-3B-Instruct`

Text	Summary
Well it's just that, you know, a pound, or a hundred pounds today, is not the same as a hundred pounds in a year's time, or two, two years' time. Right. So that would be opportunity cost? That's right, yes, that's right, so, in actual fact, this area here, right, is going to be the discounted sum of all future rural incomes.	A pound today isn't the same as a pound next year. This represents opportunity cost. The discounted sum of future rural incomes is represented here.
A: you can smell ether in the air sometimes A: It keeps the neighborhoods happy I guess. B: I guess so. A: Uh, cause they clean the, uh, the lenses for cameras A: and they, they make film here A: but it's, you know, it's under so much control. A: You know, sulphur dioxide is the big emittant from them A: but that's really getting under control now. B: Right. B: I, you know, I don't know in air what they do. B: Uh, I, I haven't run across any major pollutants down there that I've, I've really seen a lot of, you know, about. B: Uh, I, I know I've seen like, uh, B: my grandparents live in Corpus Christi, Texas B: and I know they've, there's a lot of refineries down there B: and that, that's some pretty potent stuff they put up in the air. B: I, but I don't know how, uh, you know, B: there's a difference in what you can smell and what you, uh, you know, what's bad. A: Be interesting to see when, as Mexico develops industrially whether, you know, without	A: You can smell ether in the air sometimes. It keeps the neighborhoods happy. They clean camera lenses and make film. But it's controlled well. Sulphur dioxide is a big emitter, but it's getting under control now. B: I guess so. I don't know much about air pollution. I've never seen many major pollutants around. My grandparents live near Corpus Christi with lots of refineries, which can be potent. But smelling something doesn't mean it's harmful. It would be interesting to see how Mexico develops industrially.

Table 1: Examples of original text and their respective LLM-generated summaries.

We show examples of the summarization in Table 1. The summaries show a complete removal of filler words, resulting in a significant reduction of the text length. The summaries are however not perfect. The dialogue contains less interaction between the persons, and it even mistakenly attributes the last sentence to person B rather than person A. Nevertheless, we do not need the summarized text to be a perfect reflection of the original text, we only need it to be high-quality training data.

Translation Translation is done to create parallel data. Training on parallel data is typical strategy in MT and generally leads to better alignment, but finding high-quality human-translated parallel data is difficult, particularly trilingual data covering English, Dutch, and Chinese. Meanwhile, LLMs can produce high-quality translations. We therefore translate our English data to Dutch and Chinese using the packed examples or the packed-summarized examples.

With these example triplets, we then concatenate each triplet together, using language tags to separate the languages. We show an example below:

```
The sun shines. <dutch> De zon
schijnt. <chinese> 阳光照耀。
```

We vary the order of the languages, using all 6 permutations equally often. The goal of this format is aligned with the auto-regressive nature of a decoder model: The first part teaches the model to predict the next token based only on its knowledge of the first language, while the subsequent parts let the model use its understanding of the previous languages to inform what should come next. This encourages the model to form an internal mapping between languages. In practice, each part of the triplet consists of several sentences (using our summarized context size of 200 English words), so the model is effectively trained on something more akin to paragraph-level translation.

Pipeline Given these 3 preprocessing steps, we assemble them into different pipelines to ablate their effect. The pipelines include packed-only (*P*), packed-translated (*PT*), packed-summarized (*PS*), and packed-summarized-translated (*PST*). We visualize the latter three in Figure 1.

The datasets we use are provided by the organizers: `BabyLM-2026-Strict`[3] for English, `babylm-nld`[4] for Dutch, and `babylm-zho`[5] for Chinese. *P* and *PT* require only one third of the datasets.

For *PS* and *PST*, the summarization process reduces the data to less than 33% of the original size, so we fill the remaining budget (6.2% and 32.3%, respectively) with packed English data.

3.2 Model Size

A natural question is whether the default model size has enough capacity to adequately support

[3] https://hf.co/datasets/BabyLM-community/BabyLM-2026-Strict

[4] https://hf.co/datasets/BabyLM-community/babylm-nld

[5] https://hf.co/datasets/BabyLM-community/babylm-zho

all 3 languages, while simultaneously preventing overfitting. To test this, we increase the hidden dimension size from 768 to 1024, intermediate size from 3072 to 4096, and number of layers from 12 to 24. This results in a roughly 4× larger model.

The main obstacle for training a larger model, aside from additional compute, is an increased risk of overfitting. For the pretraining phase, this is less of an issue, given that the 100M word budget allows for diverse enough text. Finetuning, on the other hand, does become problematic. To handle this, we use LoRA (Hu et al., 2022), which enables us to more conservatively update model parameters for a specific task. We use a LoRA rank of 256, alpha 512, 5% dropout, and learning rate 1e-4, applying LoRA to all linear layers except the gating layer.

3.3 Tokenization

Choosing an optimal vocabulary size is a nontrivial task, particularly for multilingual models, where one must balance tokens across languages. Our preliminary experiments found that overall, a vocabulary size of 20k was most effective. However, upon closer inspection, character-level models perform better on zero-shot tasks, but substantially worse on finetuning tasks, bringing down their overall average.

While out-of-the-box subword models outperform character models, 2 simple fixes reverse this finding. The first fix is identical to our fix for increasing model size: use LoRA. The second fix is similarly simple, increase the default context length used in finetuning from 128 to 512, accounting for the increased sequence length caused by character-level tokenization.

3.4 Experimental Setup

We use the Qwen3 model architecture (Yang et al., 2025) for our experiments. The primary reason for this is that Qwen3 uses RoPE (Su et al., 2021), which provides a useful bias towards local attention, eliminating the need for a scheduled context size increase as done in Edman and Bylinina (2023).

We note the full hyperparameter details in Table 7 in Appendix A. Our evaluation uses the MULTILINGUAL track's zero-shot and fine-tune tasks.

4 Results

4.1 Data Preprocessing

We show the effect of our preprocessing steps in Table 2. The packing step shows the largest positive effect, which makes sense as some of the examples are exceedingly short on their own, such as someone simply saying "Right." The additional context allows a model to learn longer-distance relationships between sentences, which is useful in particular for the downstream tasks requiring finetuning.

Beyond packing, both translation and summarization on their own provide a positive effect, but their combination yields the most improvement. We suspect that translation on its own may be not as effective as summarization then translation because the data being translated is of lower quality, and this is improved by the summarization step. Meanwhile, the summarization on its own, while improving data quality, does not benefit from data being parallel.

Preprocessing	∅	*P*	*PT*	*PS*	*PST*
Zero-shot					
eng	57.8	55.4	55.8	57.2	**59.0**
nld	**59.9**	58.2	54.0	55.6	56.5
zho	51.9	52.2	49.2	52.5	**52.8**
Fine-tune					
eng	29.1	41.0	43.8	40.9	**45.8**
nld	32.3	39.0	39.5	37.4	**40.2**
zho	20.3	36.5	40.5	41.7	**42.5**
avg	39.5	45.8	46.3	46.4	**48.5**

Table 2: Effect of the 3 preprocessing steps: Packing (*P*), Summarization (*S*), and Translation (*T*)

Model size	Base		Large	
LoRA?	No	Yes	No	Yes
Zero-shot				
eng	59.0	59.0	**59.5**	**59.5**
nld	56.5	56.5	**57.8**	**57.8**
zho	52.8	52.8	**52.9**	**52.9**
Fine-tune				
eng	**45.8**	45.5	21.7	45.0
nld	40.2	40.2	21.4	**41.3**
zho	42.5	42.7	20.3	**44.7**
avg	48.5	48.5	36.2	**49.3**

Table 3: Effect of model size and using LoRA for finetuning.

4.2 Model Size

The effect of model size can be seen in Table 3. The larger model has a higher zero-shot performance across all languages, but suffers from overfitting when fully fine-tuned. LoRA resolves this issue, increasing the fine-tuned performance on Dutch and Chinese particularly.

	20k		Char	
LoRA	No	Yes	No	Yes
Zero-shot				
eng	59.0	59.0	**60.9**	**60.9**
nld	56.5	56.5	**58.6**	**58.6**
zho	**52.8**	**52.8**	52.7	52.7
Fine-tune				
eng	**45.8**	45.5	37.0	44.3
nld	40.2	40.2	33.0	**41.0**
zho	42.5	**42.7**	39.8	42.1
avg	48.5	48.5	45.5	**48.8**

Table 4: Comparison of 20k vocabulary to character-level vocabulary, and the effect of LoRA.

4.3 Tokenization

We show the impact of vocabulary size in Table 5. By default, character models outperform a larger vocabulary on zero-shot tasks for English and Dutch, likely due to a better ability to generalize between the two Latin-scripted languages. However, on finetuning tasks, with no changes, we see a drop in performance. Using a larger context size of 512 tokens and LoRA, the performance recovers. Although the fine-tuning results are still slightly lower for English and Chinese, we see later in Table 5 that with a larger model size it outperforms the larger vocabulary for English. As a character-level model has fewer parameters dedicated to the embeddings and prediction layer (roughly 6-7$\times$ fewer), it may be the case that the base character-level model is too small to fully accomodate a character-level representation. More work would be needed to confirm this.

A character-level vocabulary for Chinese is likely less impactful since Chinese characters contain much more information on average, so our models are still effectively operating on a subword level for Chinese. Chinese characters can be decomposed into Kangxi radicals, and this may be a better equivalent for the character-level granularity of English and Dutch. We leave this for future work.

4.4 Overview

We show our progression from the baseline scores to our final model in Table 5. Here, we can see the impact of our methods as a whole. The majority of the performance improvement comes from preprocessing steps, roughly 9 points. Using a larger model and character-level tokenizer only improve the performance marginally. The overall improvement of our model over the organizers' baseline is around 11 points.

Model	*O.B.*	∅	*P*	*PST*	*PST* Large	*PST* Char Large
Zero-shot						
eng	56.4	57.8	55.4	59.0	59.5	**61.4**
nld	57.8	**59.9**	58.2	56.5	57.8	59.1
zho	51.3	51.9	52.2	52.8	**52.9**	**52.9**
Fine-tune						
eng	28.3	29.1	41.1	45.8	45.0	**47.5**
nld	24.8	32.3	39.1	40.2	41.3	**41.4**
zho	27.2	20.3	36.5	42.5	**44.7**	44.4
avg	38.9	39.5	45.8	48.5	49.3	**50.2**

Table 5: Comparison of the official baseline (*O.B.*) compared to various steps added, and final submitted model (right).

Model	*O.B.*	*PST* Char	*PST* Char Large
Zero-shot			
eng	53.6	57.1	**57.4**
nld	55.0	55.3	**56.1**
zho	47.3	49.3	**51.0**
Fine-tune			
eng	43.1	50.4	**53.3**
nld	39.8	47.7	**48.1**
zho	40.7	47.5	**49.6**
avg	45.9	50.9	**52.4**

Table 6: Comparison of the official baseline (*O.B.*) to our two leaderboard submissions, including hidden tasks. *PST Char* and *PST Char Large* are equivalent to *Qwenfant-0.1B* and *Qwenfant-0.4B*, respectively.

Our final model submissions, with averages including the added hidden tasks, is shown in Table 6. We opted to submit our character-level models, one larger 0.4B model for optimal performance, and one smaller 0.1B model, as this is a standard size and roughly equivalent to the official baseline size. We can see that our preprocessing and tokenization strategy bring 5 points of improvement alone, and increasing the model size brings another 1.5 points.

5 Conclusion

When pretraining multilingual models on limited data, we see that we can improve on out-of-the-box methods in a few different ways. The main improvement we see comes from our preprocessing pipeline, where a pack-summarize-translate strategy is particularly effective at both providing higher-quality data and providing parallel data. Models trained on this data perform consistently well, showing that synthetic data can be valuable,

if not better, than natural data. We also found that character-level models and larger models that typically suffer from overfitting can be coupled with a controlled fine-tuning using LoRA, thereby preventing this overfitting.

These methods provide a sample-efficient pretraining strategy that improves performance on English, Dutch, and Chinese. Future work should extend this to further languages, in order to see if these strategies scale to a massively multilingual language model. It also remains to be seen if these methods have diminishing returns in a less sample-efficient setting.

Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project b279bb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683. This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

Badr AlKhamissi, Yingtian Tang, Abdülkadir Gökce, Johannes Mehrer, and Martin Schrimpf. 2024. Dreaming out loud: A self-synthesis approach for training vision-language models with developmentally plausible data. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 244–251, Miami, FL, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lukas Edman and Lisa Bylinina. 2023. Too much information: Keeping training simple for BabyLMs. pages 89–97, Singapore.

Lukas Edman, Lisa Bylinina, Faeze Ghorbanpour, and Alexander Fraser. 2024a. Are BabyLMs second language learners? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 166–173, Miami, FL, USA. Association for Computational Linguistics.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Lukas Edman, Gabriele Sarti, Antonio Toral, Gertjan van Noord, and Arianna Bisazza. 2024b. Are character-level translations worth the wait? comparing ByT5 and mT5 for machine translation. *Transactions of the Association for Computational Linguistics*, 12:392–410.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *Preprint*, arXiv:2306.11644.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks.

Feiyang Kang, Newsha Ardalani, Michael Kuchnik, Youssef Emad, Mostafa Elhoushi, Shubhabrata Sengupta, Shang-Wen Li, Ramya Raghavendra, Ruoxi Jia, and Carole-Jean Wu. 2025. Demystifying synthetic data in LLM pre-training: A systematic study of scaling laws, benefits, and pitfalls. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10739–10758, Suzhou, China. Association for Computational Linguistics.

Francesca Padovani, Bastian Bunzeck, Manar Ali, Omar Momen, Arianna Bisazza, Hendrik Buschmeier, and Sina Zarrieß. 2025. Dialogue is not enough to make a communicative BabyLM (but neither is developmentally inspired reinforcement learning). In *Proceedings of the First BabyLM Workshop*, pages 421–435, Suzhou, China. Association for Computational Linguistics.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret

Li, Chunting Zhou, Lili Yu, Jason E Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. 2025. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9238–9258, Vienna, Austria. Association for Computational Linguistics.

Whitney Poh, Michael Tombolini, and Libby Barak. 2025. What did you say? generating child-directed speech questions to train LLMs. In *Proceedings of the First BabyLM Workshop*, pages 237–245, Suzhou, China. Association for Computational Linguistics.

Shikai Qiu, Charlie Chen, Hoang Phan, Qi Lei, and Andrew Wilson. 2026. Hyperparameter transfer enables consistent gains of matrix-preconditioned optimizers across scales. *Advances in Neural Information Processing Systems*, 38:130867–130911.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Nikitas Theodoropoulos, Giorgos Filandrianos, Vassilis Lyberatos, Maria Lymperaiou, and Giorgos Stamou. 2024. BERTtime stories: Investigating the role of synthetic story data in language pre-training. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 308–323, Miami, FL, USA. Association for Computational Linguistics.

Jannek Ulm, Kevin Du, and Vésteinn Snæbjarnarson. 2025. Contrastive decoding for synthetic data generation in low-resource language modeling. In *Proceedings of the First BabyLM Workshop*, pages 29–41, Suzhou, China. Association for Computational Linguistics.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

A Training Hyperparameters

We show the complete hyperparameters used in Table 7. Notably, we use the Muon optimizer (Jordan et al., 2024) along with AdamW for the embedding parameters (as Muon only works for 2D parameters). We chose Muon as it appears to have a better scaling factor (Qiu et al., 2026).

	Parameter	Value
Model	Architecture	Qwen3
	Hidden Size	768
	Intermediate Size	3072
	Layers	12
	Vocabulary Size	20000
Training	Context Size	512
	Batch Size (in tokens)	32768
	Epochs	10
	Grdient Clipping	1
	Warmup Steps	1000
	Scheduler	Cosine
	Optimizers	Muon, AdamW
	Learning Rates	2e-2, 5e-3
	Weight Decays	0.1, 0.01
	Epsilons	1e-7, 1e-8
	Muon Momentum	0.95
	AdamW Betas	0.9, 0.95

Table 7: Hyperparameters used (except where otherwise indicated).

User: Translate the following text to `{tgt_lang}`. Don't output any other text:

`{text}`

Assistant:

User: Shorten the following text to `{sum_len}` words or less. Don't output any other text:

`{text}`

Assistant:

Figure 2: Prompts used for translation and summarization.

B Prompts

Figure 2 shows the exact prompts used for summarization and translation.

Does a Data2vec Auxiliary Help Small Masked Language Models? A BabyLM-2026 Strict-Small Study

Ivan Dostál **Martin Benedikt** **Robin Klubarski**
Faculty of Information Technology
Czech Technical University in Prague
{dostaiva,benedma2,klubarob}@fit.cvut.cz

Abstract

We ask whether a low-weight data2vec-style Exponential Moving Average (EMA) self-distillation auxiliary (λ=0.1, τ=0.999) improves a small masked language model at the BabyLM-2026 Strict-Small scale. Our base model is a 33M-parameter LTG-BERT encoder with GPT-BERT-style hybrid masked/causal training, trained from scratch on the official 10M-word corpus in 2.3 GPU-hours on a single NVIDIA L40S. A substantial gain in Entity-Tracking during early development from adding the auxiliary motivated the study, but disappeared in later re-evaluation. In a comparison across 6 seeds, no evaluation task changed significantly (aggregate effect p=0.29), and the early gain is best explained by seed noise. Several variants intended to strengthen the auxiliary do no better. We attribute the model's competitive scores to its hybrid training recipe and to EMA weight-averaging, both already established.

1 Introduction

The BabyLM Challenge (Warstadt et al., 2023) asks how much linguistic and cognitive competence a language model can acquire from the amount of text a child would encounter. The 2026 Strict-Small track caps the corpus at 10M words and training at 10 epochs so a strong result must come from the objective, architecture, or data.

We study one candidate for such an objective: self-distillation with an Exponential Moving Average (EMA) teacher (Grill et al., 2020; Baevski et al., 2022; Assran et al., 2023), in which the model regresses onto a slowly updated running average of itself and so needs no extra text. One closely related attempt used latent bootstrapping as the main training goal at BabyLM scale and found it useful but erratic (Samuel, 2023). We test the lighter-touch version: the EMA objective as a small auxiliary (λ=0.1) on top of a GPT-BERT-trained LTG-BERT encoder, which already combines masked and next-word prediction (Charpentier and Samuel, 2024).

The result is negative. Under controlled re-evaluation the pilot gain disappears. Toggling only the auxiliary weight (λ=0.1 $\leftrightarrow$ 0), with recipe, seed, and the official 2026 evaluation pipeline held fixed across six matched seeds, the auxiliary changes *no* zero-shot task significantly (Table 2): the chance-anchored aggregate moves +0.03 (p=0.29) and Entity Tracking +0.1 (p=0.89). The dramatic development gap is best explained by seed variation on the pre-update evaluation pipeline (Section 4.1).

Contribution. Our contribution is a negative result with these findings:

- (i) a seed-controlled ablation showing a low-weight data2vec auxiliary does not improve a hybrid-trained LTG-BERT at 10M words on any of seven zero-shot tasks (Section 4);
- (ii) evidence that the model's above-chance behavior comes from the recipe and from EMA weight-averaging.
- (iii) four single-axis variants meant to strengthen the auxiliary (a higher weight, a decoupled higher-rate mask, a wider target window, a slower teacher), none of which help (Section 4.2);
- (iv) a candidate mechanism (Section 4.2): the auxiliary predicts the teacher at the same masked positions used by masked prediction, so it is largely redundant with it, and a sequence-length guard switches it off for the final phase of training.

We also could not reproduce the Entity-Tracking score on the GPT-BERT-mixed baseline's model card (39.95); our re-run of that recipe reaches

Proceedings of the Second BabyLM Workshop, pages 631–638
October 29, 2026 ©2026 Association for Computational Linguistics

20.1,[1] so all our Entity-Tracking claims are within-pipeline comparisons against our own no-auxiliary baseline, both scored with the official pipeline (see Limitations).

2 Related Work

BabyLM. Strong Strict-Small entries modify the objective or architecture: ELC-BERT (2023 winner) learns per-layer weights (Charpentier and Samuel, 2023); GPT-BERT (2024 winner) unifies fill-in-the-blank and next-word prediction with a single shared head (Charpentier and Samuel, 2024). We adopt its recipe (but with two separate heads) and its LTG-BERT backbone (Samuel et al., 2023).

Self-distillation. In self-distillation the model learns from targets produced by a copy of itself. BYOL (Grill et al., 2020) showed that predicting the features of an EMA copy, a "teacher" that trails the "student" as a slow running average of its weights, learns useful representations without collapsing. data2vec (Baevski et al., 2022, 2023) applies the same idea to masked inputs across modalities, regressing onto an average of the teacher's top layers; I-JEPA (Assran et al., 2023) predicts teacher features for hidden regions; LLM-JEPA (Huang et al., 2025) applies JEPA objectives to LLMs via paired views. We follow the data2vec formulation (Section 3).

Closest work. BootBERT (Samuel, 2023), our main inspiration, combined latent bootstrapping with MLM at BabyLM scale with the weighting inverted relative to ours (bootstrapping as the main objective, MLM as the auxiliary), and found it useful but erratic: better (Super)GLUE, mixed BLiMP, and worse linguistic generalization. We ask whether the reverse weighting, a small latent-prediction auxiliary on top of a strong masked-prediction recipe, behaves any better.

Known risks. Joint-embedding objectives risk representational collapse (Bardes et al., 2022), which EMA alone does not reliably prevent (Mo and Tong, 2024; Halvagal et al., 2023), and multi-objective training is sensitive to loss weighting and gradient interference (Kendall et al., 2018; Chen et al., 2018; Yu et al., 2020). The latter is consistent with the narrow safe range of λ we observe (Section 4.2).

[1]This was before the July 2026 change to the Entity-Tracking evaluation dataset, and we did not train on multiple GPUs as the baseline intended.

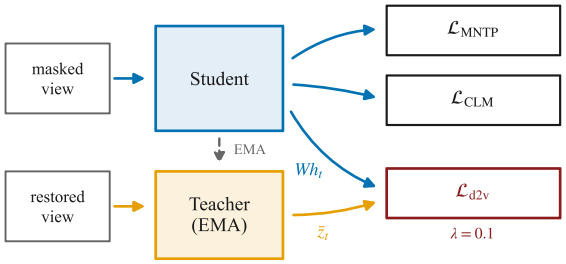


Figure 1: Training setup. MNTP and CLM cross-entropies train the student; the auxiliary $\mathcal{L}_{\text{d2v}}$ (Eq. 2) regresses Wh_t onto the teacher target $\bar{z}_t$ (mean of the layer-normalized top-8 teacher layers, stop-gradient). The teacher is an EMA copy of the student and encodes the restored view. The I-JEPA variant replaces the auxiliary branch (Appendix B).

3 Method and Setup

Data and tokenizer. We train on the official BabyLM-2026 Strict-Small corpus, with the same published 16,384-token BPE vocabulary as the GPT-BERT-mixed baseline.[2]

Base model. The backbone is the LTG-BERT encoder (Samuel et al., 2023) with the GPT-BERT-mixed release's modifications: 12 layers, hidden size 384, 6 attention heads, feed-forward size 1280, inputs up to 512 tokens; 33.4M trainable parameters (36.2M total as exported, encoder-only `LTGBertForMaskedLM`). Training follows the GPT-BERT hybrid scheme (Charpentier and Samuel, 2024): most batches (93.75%) train masked next-token prediction (MNTP): labels are shifted by one position, so each masked token is predicted from the position preceding it. The remaining 6.25% train next-word prediction under a causal mask. Masked spans of 1–3 tokens (Joshi et al., 2020), rather than single tokens (Devlin et al., 2019), are sampled at a rate annealed from 30% to 15%. Unlike GPT-BERT's single shared head, we use two separately parameterized prediction heads, both output-tied to the embeddings. The causal head is dropped at export.

The EMA auxiliary. The student is accompanied by a teacher: a full copy of the student, created at initialization and never trained by gradient descent. After every optimizer step each teacher parameter ϕ is pulled slightly toward its student counterpart θ,

$$\phi \leftarrow \tau\,\phi + (1-\tau)\,\theta, \qquad \tau = 0.999 \text{ (constant)}, \tag{1}$$

[2]`babylm-baseline-10m-gpt-bert-mixed`.

so the teacher is an exponential moving average of the student's own history (visualized in Figure 1). Its outputs serve as regression targets. On an MNTP batch the teacher encodes a restored view of the input (owing to the label shift, each masked target token is placed at its prediction position rather than the original sequence being rebuilt exactly), and the student, through a single linear head W at the masked positions M, regresses onto the mean $\bar{z}_t$ of the teacher's layer-normalized top $K{=}8$ layers, i.e., the teacher's contextual embedding of the target token. The submitted weights are the EMA teacher copy of the final checkpoint. We also trained a segment-level I-JEPA-style variant for comparison; it performed similarly to the data2vec variant, so we did not pursue it further (Appendix B).

Training loss. Each step minimizes

$$\mathcal{L} = \begin{cases} \mathcal{L}_{\text{MNTP}} + w_z \mathcal{L}_z + \lambda\, g\, \mathcal{L}_{\text{d2v}} & \text{MNTP} \\ \mathcal{L}_{\text{CLM}} + w_z \mathcal{L}_z & \text{CLM} \end{cases} \tag{2}$$

where:

- $\mathcal{L}_{\text{d2v}} := \frac{1}{|M|} \sum_{t \in M} \ell_\beta(W h_t, \text{sg}(\bar{z}_t))$,
- $\mathcal{L}_{\text{MNTP}}$ and $\mathcal{L}_{\text{CLM}}$ are cross-entropies,
- $\mathcal{L}_z$ is a LogSumExp^2 logit regularizer,
- ℓ_β the Huber loss ($\beta{=}4$),
- sg the stop-gradient,
- $\lambda{=}0.1$,
- and $g \in \{0, 1\}$ being the loss-dropout gate: $P(g{=}1){=}0.5$ below sequence length 512 and $g{=}0$ in the 512 phase (measured, $\sim$47% of sub-512-window batches carry the auxiliary gradient).

Training. Table 1 lists the hyperparameters used. It is mostly the same as the GPT-BERT-mixed baseline release's published hyperparameters (with adjusted steps to match the strict-small cap and with the auxiliary added). One run takes $\approx$2.3 GPU-hours on one NVIDIA L40S. The submitted model, including all 19 intermediate checkpoints (`chck_1M`–`chck_100M`), is available on the HuggingFace Hub.[3]

[3] https://huggingface.co/poopoobabylm/babylm-2026-strict-small-d2v-s3

Evaluation and scoring. During development we evaluated every saved model zero-shot, without fine-tuning, on BLiMP and its supplement (Warstadt et al., 2020, 2023), EWoK (Ivanova et al., 2025), Entity Tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), and reading-time prediction (SPR/ET) (de Varda et al., 2024). Entity Tracking, the skill we target, tests whether a model can follow the changing state of objects through a passage; each item is a five-way choice, so its chance level is 20.0, whereas the other minimal-pair tasks are binary (chance 50.0) and the reading-time scores are ΔR^2 over a baseline regression, $\times 100$ (chance 0). We report raw per-task scores plus an aggregate defined as

$$\text{agg} = \frac{1}{T} \sum_{i=1}^{T} \frac{s_i - c_i}{b_i - c_i}, \tag{3}$$

where s_i is our model's score on test i, c_i its chance level, and b_i the score of the official GPT-2 anchor baseline (Radford et al., 2019), evaluated with the same updated pipeline. For the final analysis, we exclude tasks with $b_i - c_i < 0.5$: a reference below chance would reward lower accuracy, while a reference too close to chance makes normalization unstable. This excludes EWoK and Entity Tracking ($T{=}5$); both remain in the per-task analysis. Normalized scores are capped at ± 2 when $b_i - c_i < 5$ (binds for Reading-SPR). A score of 1.0 ties the updated anchor on the included tasks.[4] Development and submission selection used the original six-task aggregate with pre-update reference scores; the final analysis reported here uses this revised five-task definition.

4 Results

The auxiliary changes no task significantly. To isolate the auxiliary's effect we run a single-axis ablation: the identical recipe (Table 1) trained with the auxiliary weight set to $\lambda{=}0.1$ (aux) and to $\lambda{=}0$ (no-aux), six seeds each. Table 2 reports the seed-averaged per-task scores, the chance-anchored aggregate of Eq. 3, and the paired difference. No task reaches significance. The aggregate moves $+0.03$ ($p{=}0.29$) and Entity Tracking $+0.1$ ($p{=}0.89$), and the largest effect is a slight EWoK *drop* ($p{=}0.11$). Per-seed differences are the size of seed noise and change sign across seeds. The auxiliary therefore

[4] The reference scores are BLiMP 65.37, Supplement 59.60, COMPS 51.81, Reading-SPR 1.64, and Reading-ET 9.63.

Hyperparameter	Value
Architecture	LTG-BERT encoder
Training scheme	GPT-BERT hybrid
Shape (L/H/d/FFN)	12 / 6 / 384 / 1280
Parameters	33.4M (36.2M exported)
Vocabulary	16,384 (BPE)
Objective mix	93.75% MNTP / 6.25% CLM
Auxiliary	data2vec EMA self-distillation
Aux. λ / EMA τ	0.1 / 0.999
Aux. target	smooth-L_1, top-K=8 layers
Optimizer / LR	LAMB / 7×10^{-3}
LR scheduler	cosine, warmup/cooldown 1.6%
Precision / batch	bf16 / 128 sequences
Steps / epochs	9,656 / 10
Words seen	99.95M (cap: 100M)
Masking	spans 1–3, rate 0.3 $\rightarrow$ 0.15
Seq. length	128 $\rightarrow$ 256 $\rightarrow$ 512
Seed (submission)	3
Compute	1$\times$L40S, 2.3 h, $\approx 3.2 \times 10^{16}$ FLOPs

Table 1: Training recipe for the submitted model (`exp_seed_d2v_s3`).

Task	Chance	Aux (λ=0.1)	No-aux (λ=0)	Δ	p
aggregate	0	**1.369**	**1.342**	+0.03	0.29
BLiMP	50	72.68	72.61	+0.07	0.75
Supplement	50	59.40	58.33	+1.07	0.21
EWoK	50	50.38	51.77	−1.39	0.11
Entity Tracking	20	21.38	21.26	+0.12	0.89
COMPS	50	52.23	52.20	+0.03	0.86
Reading (SPR)	0	4.93	4.81	+0.12	0.15
Reading (eye-tr.)	0	11.13	11.12	+0.01	0.97

Table 2: Seed-controlled ablation of the data2vec auxiliary: mean over six matched seeds, auxiliary toggled λ=0.1 (aux) vs. λ=0 (no-aux). Chance is the per-task chance level; the aggregate follows Eq. 3, excluding EWoK and Entity Tracking (0 is chance, 1 ties the updated GPT-2 anchor on the included tasks). Δ is the mean aux−no-aux difference, p a paired t-test. No task reaches significance.

provides no measurable benefit at the 10M-word scale. Both aux and no-aux sit above the re-scored GPT-2 anchor on Entity Tracking (19.10, itself below the 20.0 chance level), so the above-chance behavior is present with or without the auxiliary. Using the original development aggregate gives the same qualitative conclusion (+0.03, p=0.67).

4.1 The development gain was a single-seed artifact

The study was motivated by a striking development result: before the exposure cap, one configuration reached Entity Tracking ≈28, far above its no-auxiliary counterpart. Two facts dissolve it. First, at this scale the seed matters more than the auxiliary. Table 3 reports the six matched seed pairs individually: the per-seed aux−no-aux difference swings from −2.3 to +3.2, is negative for two seeds and positive for four, and its standard deviation (2.0) is an order of magnitude larger than its mean (+0.1). The mean effect of the auxiliary is small relative to this seed-to-seed spread, so a comparison that rests on one seed per condition, as our development comparison did, can be dominated by noise. The development gap was moreover measured on the pre-update pipeline, where per-seed Entity Tracking ranged more widely (some seeds falling to 15–17); comparing a lucky auxiliary seed to an unlucky baseline seed there manufactures a large gap from noise. Second, the pre-update evaluation pipeline scored Entity Tracking differently (the July 2026 update filters ≈ 29% of items; see Limitations), inflating the apparent margin further. Re-evaluating for both, with matched seeds and one pipeline, the gap vanishes (Table 2). The submitted model (data2vec seed 3) was selected using the pre-update aggregate; it is a strong entry, but its strength is not attributable to the auxiliary.

Seed	Aux (λ=0.1)	No-aux (λ=0)	Δ
0	20.76	23.06	−2.30
1	21.23	23.14	−1.91
2	21.54	21.28	+0.26
3	21.53	20.49	+1.04
4	20.04	19.67	+0.37
5	23.18	19.93	+3.25
mean	21.38	21.26	+0.12
sd	1.04	1.53	2.03

Table 3: Per-seed Entity Tracking for the six matched seed pairs behind Table 2, reported to compare the size of the auxiliary effect with the size of seed variation. The per-seed effect is negative for two seeds and positive for four, and its standard deviation (2.0) is an order of magnitude larger than its mean (+0.1); the paired difference is not significant (p=0.89).

4.2 Possible explanations and attempts to improve the auxiliary

Two structural properties may explain why the auxiliary fails. (i) The auxiliary regresses the student's projected state onto the EMA teacher's contextual embedding *at the same masked positions* that masked prediction already supervises with a cross-entropy target. Predicting a slowly-averaged copy of one's own representation of the masked token may largely restate the masked-prediction objective. (ii) A sequence-length guard disables the auxiliary once inputs reach the full 512-token length (a memory measure), so it is turned off for the final

Variant	Change	Entity Tr.
No-aux (baseline)	λ=0	19.7–23.1
Aux (default)	λ=0.1	20.0–23.2
Higher weight	λ=0.15	21.7–22.6
Decoupled mask	disjoint, rate 0.40	20.3–22.9
Wider target	$\pm$4-token window	22.2
Slower teacher	$\tau\rightarrow$0.9999	19.1

Table 4: Single-axis attempts to rescue the auxiliary. None of the four modifications exceeds the best no-auxiliary Entity-Tracking score.

refinement phase of training (i.e., the phase that most shapes the exported weights).

We retrained the recipe with one change at a time, each targeting a suspected cause: a higher weight (λ=0.15), a decoupled higher-rate auxiliary mask, a $\pm$4-token target window, and a slower teacher ($\tau\rightarrow$0.9999). None helped. Entity-Tracking scores for the four modifications ranged from 19.1 to 22.9, compared with 19.7–23.1 for the no-auxiliary baseline (Table 4). Within the range we tested, the negative result is a property of the objective at this scale, not of a single unlucky hyperparameter.

5 Conclusion

A low-weight data2vec-style EMA self-distillation auxiliary does not improve a 33M-parameter hybrid-trained LTG-BERT at the BabyLM-2026 Strict-Small scale. In a seed-controlled, single-axis ablation the auxiliary moves no zero-shot task significantly (aggregate $+0.03$, p=0.29).

The large Entity-Tracking gain we first observed was a favourable seed draw scored on the pre-update pipeline. Four single-axis variants of the auxiliary (weight, mask, target window, teacher rate) did not change the outcome. The model itself remains a competitive Strict-Small entry, but we attribute that competitiveness to the hybrid training recipe and EMA weight-averaging, both already present in the no-auxiliary baseline, rather than to the auxiliary. We hypothesize that, at this scale, latent prediction would need to provide more complementary supervision to improve on the hybrid baseline.

Limitations

Unreproduced baseline score. We could not reproduce the Entity-Tracking score published on the GPT-BERT-mixed baseline's model card (39.95 under MNTP scoring): our retraining of the published recipe reaches 20.1 under the official 2026 pipeline. All Entity-Tracking claims in this paper are therefore within-pipeline comparisons against our own no-auxiliary baseline, and we cannot rule out that the published number reflects a configuration detail we failed to replicate.

Seed and statistical power. Our conclusion rests on six matched seed pairs. We found no statistically significant improvement, but this comparison has limited power to detect small effects. The submitted model was selected from the auxiliary seeds using the pre-update aggregate, a selection that inflates its pre-update headline number but is orthogonal to the aux-vs-no-aux comparison, which is seed-matched.

Evaluation pipeline update. A July 2026 update to the official pipeline changed the Entity-Tracking data selection. Re-scored under the updated pipeline, the submitted model's full-eval Entity-Tracking average moves from 21.66 to 19.92 (regular subset 23.28 $\rightarrow$ 21.53) and the GPT-2 anchor from 21.68 to 19.10 (regular subset). EWoK also changes slightly (53.41 $\rightarrow$ 51.82, full evaluation) from an unrelated fix in the same update. EWoK remains above chance under both pipeline versions. EWoK and Entity Tracking are excluded from the updated aggregate because their updated GPT-2 reference scores are below chance (Section 3). The updated GPT-2 anchor's regular-only and full Entity-Tracking scores both round to 19.10.

On the updated pipeline the submitted model still scores above the re-scored anchor on Entity Tracking (21.53 vs 19.10 regular; 19.92 vs 19.10 over all three categories), so the within-pipeline comparison that our analysis rests on is preserved. The absolute margin, however, is smaller than under the pre-update pipeline and sits close to the 20.0 chance level on the three-category average, which we flag as a caveat on the strength of the Entity-Tracking result.

Acknowledgments

We are grateful to our instructor Zdeněk Kasner, whose course on neural language models at Czech Technical University in Prague introduced us to the BabyLM Challenge and whose guidance and editorial advice made this submission possible.

References

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann

LeCun, and Nicolas Ballas. 2023. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Alexei Baevski, Arun Babu, Wei-Ning Hsu, and Michael Auli. 2023. Efficient self-supervised learning with contextualized target representations for vision, speech and language. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. 2022. data2vec: A general framework for self-supervised learning in speech, vision and language. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*.

Adrien Bardes, Jean Ponce, and Yann LeCun. 2022. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations (ICLR)*.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every Layer Counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 238–252.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 English sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. 2020. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Manu Srinath Halvagal, Axel Laborieux, and Friedemann Zenke. 2023. Implicit variance regularization in non-contrastive SSL. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hai Huang, Yann LeCun, and Randall Balestriero. 2025. LLM-JEPA: Large language models meet joint embedding predictive architectures. *arXiv preprint arXiv:2509.14252*.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, and 1 others. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics (TACL)*, 8.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Shentong Mo and Shengbang Tong. 2024. Connecting joint-embedding predictive architecture with contrastive self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Technical Report*.

David Samuel. 2023. Mean BERTs make erratic language teachers: the effectiveness of latent bootstrapping in low-resource settings. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 221–237.

David Samuel, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2023. Trained on 100 million words and still in shape: BERT meets British National Corpus. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1954–1974.

Task	Fast	Full
BLiMP	72.60	72.07
Supplement	60.40	60.05
EWoK	53.23	53.41 (51.82 after change)
Entity Tracking	23.28	21.66 (19.92 after change)
COMPS	51.85	51.85
Reading (SPR)	4.77	4.77
Reading (eye-tr.)	11.30	11.30
Reading overall	8.03	8.03

Table 5: Fast versus full zero-shot evaluation of the submitted model (MNTP scoring). Fast Entity Tracking is the regular subset; full is the average over all three categories. Unparenthesized values use the pre-update pipeline. Parenthetical values are the post-update-pipeline scores (July 2026): the Entity-Tracking data filter and an unrelated EWoK scoring fix each move one task.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics (TACL)*, 8.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

A Full-Evaluation Results

The leaderboard submission is produced by the full evaluation pipeline, which differs from the fast evaluation used in the body of the paper. Table 5 compares the two conventions for the submitted model. Under the pre-update pipeline, its Entity-Tracking scores are 23.28 (regular), 23.96 (move-contents), and 17.76 (ambiguous-reference); the ambiguous-reference category is thus the one that pulls the full-eval average below the regular subset. By linguistic domain, BLiMP decomposes into morphology 81.7, syntax 68.9, syntax–semantics 69.3, and semantics 66.4. Table 6 reports the full pipeline's (Super)GLUE fine-tuning results.

Task	Accuracy	F1	MCC
BoolQ	0.673	0.744	0.294
MultiRC	0.659	0.453	0.288
RTE	0.547	0.496	0.085
WSC	0.712	0.595	0.376
MRPC	0.711	0.825	0.255
QQP	0.722	0.625	0.404
MNLI	0.469	—	—

Table 6: (Super)GLUE fine-tuning results of the submitted model under the full evaluation pipeline (accuracy, F1, Matthews correlation).

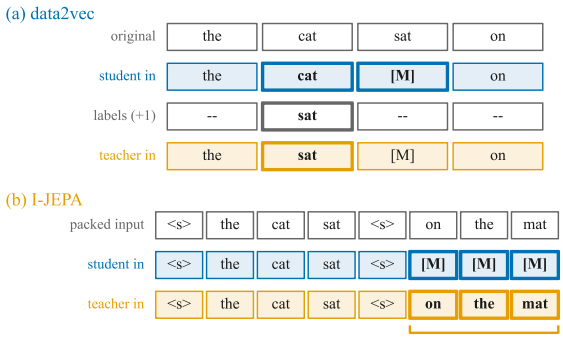


Figure 2: What each network sees (toy inputs). (a) data2vec variant: spans of 1–3 tokens are masked; with the MNTP label shift, the position of "cat" predicts "sat", and the teacher's restored view carries the target token at that prediction position. (b) I-JEPA variant: one whole <s>-delimited line of the packed (restored) input is masked in a fresh copy; the bracket marks the masked line, which is predicted as a single pooled vector.

B Auxiliary Mechanics and the I-JEPA Variant

Both variants share the EMA teacher of Eq. 1 (Section 3).

What each network sees. Figure 2 shows the two auxiliary branches. In the **data2vec** variant (a), spans of 1–3 tokens are masked and, because labels are shifted by one position (MNTP), the position of "cat" predicts "sat"; the auxiliary regresses the projected student state Wh_t at that position onto the teacher's state $\bar{z}_t$, the mean of the teacher's layer-normalized top 8 layers. The teacher's restored view places the target token at the prediction position rather than rebuilding the original sequence exactly (Section 3). In the **I-JEPA** variant (b), one whole <s>-delimited corpus line ($\geq$10 tokens) is masked in a fresh copy of the restored input; the student's states over the masked line are mean-pooled, passed through a two-layer MLP head, and regressed with a cosine loss onto the teacher's mean-pooled final-layer representation of the same line, for at most 8 sequences per batch.

The variants share the EMA teacher, the weight $\lambda{=}0.1$, the 0.5 loss-dropout, and the 512 cutoff; they differ in masking granularity (short spans vs. a whole line), target construction (layer-normalized top-8 mean per token vs. final-layer mean-pool per segment), prediction head (single linear vs. two-layer MLP), and loss (Huber vs. cosine).

Recursive Transformers for BabyLM 2026

Serdar Gülbahar[1] **Alexander Fraser**[1,2,3] **Lukas Edman**[1,2]

[1]School of Computation, Information and Technology, TU Munich
[2]Munich Center for Machine Learning
[3]Munich Data Science Institute

go34tus@mytum.de, lukas.edman@tum.de

Abstract

The BabyLM shared task asks whether language models can acquire robust linguistic behavior from developmentally sized corpora (Hu et al., 2024; Choshen et al., 2026). We study a causal transformer recipe for this setting that increases computation per token without adding a full stack of independent layers. Our RecGPT models reuse one transformer block recurrently, factorize the vocabulary maps to keep small models from being dominated by embeddings, and add a training-only next-latent auxiliary loss that encourages hidden states to support local transitions. We submit RecGPT-10M for Strict-Small and RecGPT-100M for Strict. Across the BabyLM 2026 zero-shot and fine-tuned evaluations, the submitted models improve over GPT-2 baselines and are competitive with GPT-BERT (Charpentier and Samuel, 2024).

1 Introduction

The BabyLM shared task asks how much language learning can be recovered when pretraining data is limited to developmentally motivated scales rather than web-scale corpora (Hu et al., 2024; Choshen et al., 2026). The 2026 edition keeps the two English data budgets used here: Strict-Small, capped at 10M words, and Strict, capped at 100M words. We focus on architecture and training objective choices that can utilize more compute to squeeze more learning out of a fixed data budget while limiting growth in trainable capacity to avoid risk of overfitting.

Our submission builds on RecursiveGPT, introduced by Gülbahar et al. (2026) as an architecture for scaling per-token computation under limited data. It applies the same self-attention and MLP weights R times across depth (Figure 1), following shared-block transformer designs (Dehghani et al., 2019; Lan et al., 2020). The goal is to add per-token computation without proportionally increasing unique parameters or training for more epochs,

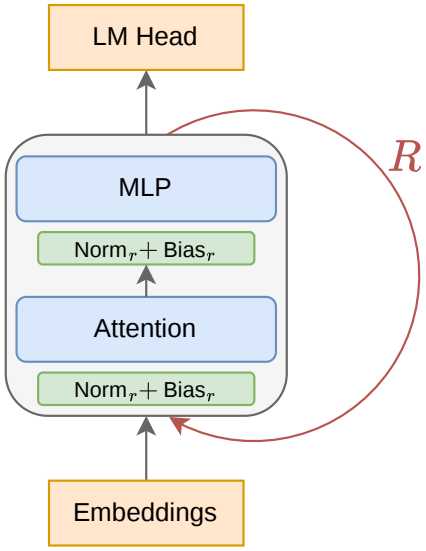


Figure 1: Recursive transformer architecture, adapted from Gülbahar et al. (2026). One attention/MLP block is reused for R recurrent steps, with step-specific normalization and bias; embeddings and the LM head remain outside the recursion.

a useful axis when the data budget is fixed but compute is relatively abundant. Here, we adapt RecursiveGPT to BabyLM 2026 with a new data mixture, an updated optimization recipe, and a NextLat-style auxiliary objective (Teoh et al., 2025). NextLat is orthogonal to recursive weight sharing: it is a training objective that can be applied to any transformer. During training, an auxiliary network learns a local transition model in hidden-state space: given the current final hidden state and the embedding of the next token, it predicts the next final hidden state. The main model receives gradients from this auxiliary loss, encouraging it to compress the sequence into a belief state, rather than relying on attention. Decoding remains unchanged as the auxiliary network is not needed, and is removed after

Proceedings of the Second BabyLM Workshop, pages 639–645
October 29, 2026 ©2026 Association for Computational Linguistics

pretraining.

We report two submitted English models, one for each data budget. RecGPT-10M is a 34.2M-parameter Strict-Small transformer trained with $R = 16$. RecGPT-100M is a 124.0M-parameter Strict transformer trained with $R = 24$.

2 Data

We combine the official BabyLM 2026 Strict corpus with NVIDIA's Nemotron-ClimbMix (Diao et al., 2025), a pretraining corpus produced through mixture optimization over 20 semantically derived clusters. The target mixture is 33% BabyLM Strict and 67% ClimbMix. We exclude ClimbMix clusters 11 and 18 because they contain substantial amounts of source code. The resulting Strict split contains exactly 100M words: 32,994,954 from BabyLM Strict and 67,005,046 from ClimbMix. The Strict-Small split contains exactly 10M words: 3,280,460 from BabyLM Strict and 6,719,540 from ClimbMix.

Each model uses a byte-level BPE tokenizer trained on its corresponding data split: 10M words for Strict-Small and 100M words for Strict. Both tokenizers have 32,768 entries and use a single padding token. Training examples are document packed. Attention and auxiliary transitions are masked by document segment IDs, so packed documents cannot attend to, or predict hidden states across, neighboring documents in the same sequence.

3 Model

The base model follows the RecursiveGPT architecture of Gülbahar et al. (2026): a decoder-only transformer (Vaswani et al., 2017) with rotary position embeddings (Su et al., 2024), RMSNorm, causal self-attention, and a ReLU-squared MLP. Its central feature is recursion: one attention module and one MLP module are reused for every recursive step, while the attention and MLP pre-norm scale and bias parameters are depth-specific. This allows us to scale up computational depth without adding another full set of attention and MLP parameters at each step. Thus RecGPT-10M applies $R = 16$ recurrent transformer steps, and RecGPT-100M applies $R = 24$ steps. Table 1 summarizes the submitted model configurations.

Following RecursiveGPT, we use ALBERT-style learned projections to decouple embedding dimension from transformer hidden size (Gülbahar et al., 2026; Lan et al., 2020). We apply this factorization to separate, untied input and output vocabulary maps, preventing them from dominating the parameter count as we scale down the model. Input token embeddings use a smaller embedding dimension, are projected into the hidden space, and final hidden states are projected back to the embedding space before the output vocabulary projection (Figure 2). We do not tie input and output embeddings.

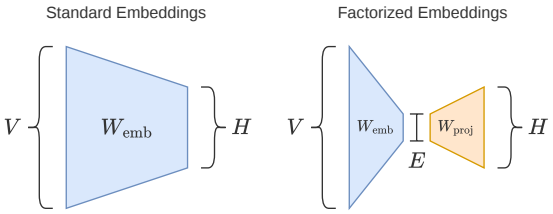


Figure 2: Standard and factorized vocabulary maps, adapted from Gülbahar et al. (2026). The factorized version uses embedding dimension E as an intermediate size rather than mapping the vocabulary directly to hidden dimension H.

Configuration	RecGPT-10M	RecGPT-100M
Recursive depth R	16	24
Hidden size H	768	1,408
Embedding size E	192	768
Heads × head dim.	12×64	22×64
MLP expansion	16×	16×
Sequence length	256	512
Unique words / tokens	10M / 14.2M	100M / 142.5M
Epochs	10	10
Unique Parameters	34.2M	124.0M

Table 1: Submitted RecGPT model configurations.

If we do not use factorized embeddings, these two vocabulary maps would use 50.3M parameters for RecGPT-10M and 92.3M for RecGPT-100M, or 70.3% and 56.3% of the corresponding unfactorized models.

For the 10M budget, we chose this size as the largest configuration before our development runs began to show signs of overfitting. For the 100M budget, we did not see comparable signs of overfitting in the submitted configuration, but stopped scaling due to compute constraints. The unusually high expansion factor is needed because MLP neuron count scales linearly with hidden size, whereas total parameter count scales quadratically. Simply scaling the hidden size of one shared block to reach the target parameter budget would leave too few feed-forward neurons, which can be interpreted as associative memories over key-value pairs (Gülbahar et al., 2026; Zhong et al., 2025).

4 Next-Latent Auxiliary Objective

Standard next-token training does not directly require hidden states to support a compact transition rule. NextLat addresses this by adding self-supervised prediction in latent space (Teoh et al., 2025).

Let h_t be the final normalized hidden state at position t, and let e_{t+1} be the model input embedding of the next token. The auxiliary network predicts $\hat{h}_{t+1} = g(h_t, e_{t+1})$ and is trained against a detached target h_{t+1}. This encourages the main model to produce hidden states that are predictive of future hidden states, which can be interpreted as a form of belief-state compression. We use Smooth L1 loss summed over hidden dimensions. For simplicity, we omit the output-head KL alignment term used by NextLat and train only the hidden-state prediction loss.

We train a two-step rollout, so the auxiliary prediction is also fed back to predict one additional future hidden state. The original paper treats the latent-transition model design space as open, noting that richer transition architectures and belief-state constructions are natural future directions; our BabyLM implementation is a slightly different setup, described in Appendix B. Transitions are valid only within the same packed document segment. The auxiliary loss weight is 0.01 in both reported models (However, we use Smooth L1 loss summed over hidden dimensions, rather than a mean, so this weight is not comparable to the original paper), with an auxiliary learning rate of 0.004 and auxiliary weight decay of 0.01. The auxiliary model is not saved as part of the submitted language model checkpoint.

5 Development Ablation

We selected NextLat from a Strict-Small development ablation. Five $R = 16$ models were trained with NextLat and five without it, using the same 10M split of our dataset, tokenizer, optimizer settings, and data seed. Table 2 reports mean zero-shot scores across these five seeds on BLiMP (Warstadt et al., 2020), BLiMP Supplement, EWoK (Ivanova et al., 2025), and CoMPs (Misra et al., 2023). NextLat improved BLiMP by 0.77 points and EWoK by 0.53 points, while leaving Supplement unchanged and hurting CoMPs by 0.60 points.

Metric	NextLat	No NextLat	Delta
BLiMP	**73.15**	72.38	+0.77
Supplement	**60.90**	**60.90**	+0.00
EWoK	**53.39**	52.87	+0.53
CoMPs	54.44	**55.04**	-0.60

Table 2: Strict-Small development ablation over five seeds. Scores are BabyLM 2026 zero-shot means.

Setting	Value
Optimizer	AdamW
Learning rate	1×10^{-4}
Weight decay	0.01
LoRA rank / alpha	256 / 512
LoRA dropout	0.0
Target modules	`qkv`, `out`, `up`, `down`
Batch size	16 (BoolQ, MultiRC); 32 (other)
Maximum epochs	30 (WSC); 10 (other)
Sequence length	256
Seed	42

Table 3: LoRA settings used for the finetuned evaluation.

6 Pretraining

Both models are trained for the BabyLM budget with the same deliberately small batch size of 32,768 tokens. The main model uses separate learning rates for shared transformer blocks and embedding/projection parameters: 0.02 for block parameters and 0.005 for embedding-side parameters in both models. Two-dimensional attention and MLP weights, including auxiliary-model matrices, are optimized with Aurora, a leverage-aware extension of Muon for rectangular matrices (Jordan et al., 2024; Dewulf et al., 2026). Aurora balances update energy across the rows of tall matrices while preserving Muon's polar-update geometry. This makes it particularly suitable for our architecture, as we use an MLP expansion factor of 16 (Dewulf et al., 2026). Vocabulary maps, factorization projections, normalization parameters, and other non-matrix parameters are optimized with AdamW (Loshchilov and Hutter, 2019). We apply cautious weight decay (Chen et al., 2026) to all parameters. Following Marek et al. (2025), we tune AdamW's second-moment decay in terms of its half-life in tokens; fixing the number of tokens per optimizer step across both scales lets us use the same betas, $(0.9, 0.997)$, for both models. Aurora momentum is 0.95.

Track	Model	Eval	Params	BLiMP	Suppl.	EWoK	Entity Tracking	CoMPs	Global PIQA	SG	NLP Average
10M	GPT-2 2026 baseline	Causal	98.0M	65.23	57.25	50.63	**19.10**	51.81	35.09	63.27	48.91
10M	GPT-BERT causal-focus	Causal	31.0M	71.66	63.21	49.49	-	52.80†	-	65.13	-
10M	GPT-BERT masked-focus	MNTP	31.0M	70.36	**63.71**	49.95	-	53.60†	-	66.02	-
10M	RecGPT-10M	Causal	34.2M	**73.11**	61.73	**52.62**	16.59	**55.43**	**40.68**	**66.64**	**52.40**
100M	GPT-2 2026 baseline	Causal	98.0M	74.73	65.00	54.37	16.91	55.85	36.62	67.35	52.98
100M	GPT-BERT causal-focus	Causal	120.0M	79.29	70.42	52.32	-	58.30†	-	70.05	-
100M	GPT-BERT masked-focus	MNTP	120.0M	**80.75**	**75.34**	51.77	-	58.30†	-	70.58	-
100M	RecGPT-100M	Causal	124.0M	80.06	69.28	**59.05**	**19.50**	**60.55**	**43.15**	**71.84**	**57.63**

Table 4: BabyLM NLP results. Entity denotes Entity Tracking, GlobalPIQA is the macro-average of its parallel and nonparallel variants, and SG is defined in Section 7. NLP Average is the unweighted mean of the seven task columns; dashes mark unavailable scores. Available GPT-2 and GPT-BERT values are from the BabyLM 2026 leaderboard (BabyLM Community, 2026); †CoMPs values are from the 2025 leaderboard (BabyLM Community, 2025). Scores are percentages.

7 Evaluation

We use the BabyLM 2026 strict evaluation pipeline for the main comparison. The zero-shot columns in Table 4 come from the causal backend and best-temperature reports. Table 4 compares the BabyLM NLP tasks, not the human-likeness tasks. No reading-time tuning or AoA-specific post-processing is included in Table 4.

For the finetuned (Super)GLUE evaluation, we freeze the pretrained model and train LoRA adapters (Hu et al., 2022) together with a classification head for each task. We chose LoRA in part to reduce overfitting risk on the small finetuning datasets, although we do not isolate its effect relative to full finetuning. We selected the learning rate and adapter rank through a grid search on a 10M development model, then used the same settings for both submitted models. Table 3 gives the selected settings. For each task, we report the checkpoint with the best validation score. SG is the resulting aggregate: accuracy for BoolQ, MultiRC, RTE, WSC, and MNLI, and F1 for MRPC and QQP.

8 Results

Against GPT-2, the only baseline with scores on all seven reported tasks, RecGPT has a substantially higher NLP Average in both tracks. RecGPT-10M outperforms GPT-2 on six of seven tasks, with Entity Tracking as the exception, while RecGPT-100M outperforms it on all seven. All Entity Tracking scores nevertheless remain below random chance, and we observe high variance between runs. RecGPT also obtains the highest SG score in each track.

GPT-BERT is a stronger comparison, although its results are incomplete. At both scales, RecGPT scores higher than both GPT-BERT variants on EWoK and CoMPs, while GPT-BERT scores higher on BLiMP Supplement. On BLiMP, RecGPT leads at 10M and comes within one point of the best GPT-BERT score at 100M.

We do not compare overall averages with GPT-BERT because its Entity Tracking and GlobalPIQA scores are unavailable. Its CoMPs values also come from the 2025 leaderboard rather than the 2026 evaluation used for the other results. Thus, RecGPT is competitive with GPT-BERT as a purely causal model, but is not better on every task.

9 Related Work

The main BabyLM comparison point for this submission is GPT-BERT (Charpentier and Samuel, 2024), which trains one transformer stack with both masked and causal objectives. GPT-BERT is especially relevant because it gives strong BabyLM zero-shot baselines at both 10M and 100M words, but it also changes the evaluation interface: some rows are evaluated with masked-next-token prediction and some with causal scoring. We therefore report the GPT-BERT rows in Table 4 with their evaluation mode rather than treating them as one homogeneous baseline.

This work directly builds on RecursiveGPT (Gülbahar et al., 2026), which combines recurrent application of a shared transformer block with factorized vocabulary maps for limited-data pretraining. These components draw on prior work on Universal Transformers and ALBERT (Dehghani et al., 2019; Lan et al., 2020). This submission evaluates an updated recipe on BabyLM 2026 and adds NextLat, which is orthogonal to both architectural

choices. NextLat encourages hidden states to be predictive of future hidden states while leaving the architecture and inference path unchanged (Teoh et al., 2025). Here we use it as a practical auxiliary loss for BabyLM-scale pretraining.

10 Discussion

Our results support the compute-allocation strategy proposed by Gülbahar et al. (2026): recursive depth gives more computation per token while factorized vocabulary maps and weight sharing control unique parameters. This matters because BabyLM is not a setting where additional compute is automatically converted into better scores. The third BabyLM findings report observes that new objectives and architectures tended to produce the strongest submissions, and that training FLOPs were not completely correlated with performance (Charpentier et al., 2025). That result suggests that the useful question is not simply whether to spend more compute, but how to expose a small model to additional computation without creating an oversized parameterization that overfits the fixed word budget. This is especially salient under the recent BabyLM exposure cap, which effectively limits models to ten epochs and is motivated by avoiding developmentally implausible repeated exposure while keeping successful training procedures accessible (Charpentier et al., 2025). In that context, spending computation within each exposure through recurrence and latent prediction is a more relevant scaling direction than simply extending training for additional epochs.

RecGPT addresses this by turning recurrent depth into a compute-scaling axis that is partially decoupled from trainable capacity. The shared transformer block can be applied for more steps, while the vocabulary factorization prevents the fixed vocabulary maps from consuming most of the parameter budget. NextLat adds a second, training-only route for spending compute: hidden states are asked to support short latent rollouts, but the submitted checkpoint and inference interface remain the base causal model. Together, these choices make RecGPT a useful base architecture for BabyLM-style methods that need more processing per token but cannot rely on simply increasing data, epochs, or unconstrained model width.

The benefits of NextLat are not yet conclusive: the 10M ablation shows modest, task-dependent effects. It is possible that our models could be tuned further to perform similarly without NextLat, and we do not yet have a clear understanding of how to best leverage this auxiliary loss in the BabyLM setting. This remains an open area for future research.

Limitations

Compute limits prevented a full 100M-scale ablation study. RecGPT-100M therefore evaluates the final RecGPT recipe as a combined system rather than isolating recursion depth, vocabulary factorization, NextLat weight, data mixture, or optimizer choices at that scale. Our comparison with GPT-BERT is incomplete because the 2026 leaderboard does not report Entity Tracking or GlobalPIQA for those models, preventing a directly comparable NLP Average; their CoMPs values are additionally taken from the 2025 leaderboard. The official GPT-2 baseline provides a complete same-track reference, but it is comparatively weak, so improvements over it alone should be interpreted cautiously.

RecGPT uses substantially more training compute than the baselines. Our experiments examine whether this additional compute can be used productively under a fixed data budget, but do not establish that recurrence is the most compute-efficient way to use it; other approaches may yield better returns at the same compute budget.

Finally, we do not report reading-time or AoA results here, as our methods are not designed to optimize for human-likeness, and those benchmarks may change the relative ranking of systems.

Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project b279bb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683. This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

BabyLM Community. 2025. BabyLM leaderboard 2025 all tasks. https://huggingface.co/spaces/BabyLM-community/babylm-leaderboard-2025-all-tasks. Accessed 2026-07-06.

BabyLM Community. 2026. BabyLM leaderboard 2026. https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026. Accessed 2026-07-06.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Lizhang Chen, Jonathan Li, Kaizhao Liang, Baiyu Su, Cong Xie, Nuo Wang Pierse, Chen Liang, Ni Lao, and Qiang Liu. 2026. Cautious weight decay. In *International Conference on Learning Representations*.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

Alec Dewulf, Dhruv Pai, Li Yang, Ashley Zhang, and Ben Keigwin. 2026. Aurora: A leverage-aware spectral optimizer. *arXiv preprint arXiv:2606.27715*.

Shizhe Diao, Yu Yang, Yonggan Fu, Xin Dong, Dan Su, Markus Kliegl, Zijia Chen, Peter Belcak, Yoshi Suhara, Hongxu Yin, Mostofa Patwary, Yingyan Lin, Jan Kautz, and Pavlo Molchanov. 2025. Nemotron-CLIMB: CLustering-based iterative data mixture bootstrapping for language model pre-training. *Preprint*, arXiv:2504.13161.

Serdar Gülbahar, Lukas Edman, and Alexander Fraser. 2026. Squeezing more from limited data with recursive transformers. *arXiv preprint arXiv:2608.26973*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyurek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Martin Marek, Sanae Lotfi, Aditya Somasundaram, Andrew Gordon Wilson, and Micah Goldblum. 2025. Small batch size training for language models: When vanilla SGD works, and why gradient accumulation is wasteful. In *Advances in Neural Information Processing Systems*, volume 38.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S. Hu, Tim Pearce, Pratyusha Sharma, Akshay Krishnamurthy, Riashat Islam, Alex Lamb, and John Langford. 2025. Next-latent prediction transformers learn compact world models. *Preprint*, arXiv:2511.05963.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Shu Zhong, Mingyu Xu, Tenglong Ao, and Guang Shi. 2025. Understanding transformer from the perspective of associative memory. *Preprint*, arXiv:2505.19488.

A Compute

Table 5 reports approximate pretraining compute and GPU time for the submitted models. PFLOPs are the values declared in our leaderboard submissions, and GPU time is the elapsed W&B runtime on one NVIDIA H100.

Model	Training PFLOPs	GPU time
RecGPT-10M	359.71	20 min
RecGPT-100M	15,945.52	10 h

Table 5: Approximate pretraining compute and elapsed GPU time.

B NextLat Auxiliary Model Design

Our auxiliary model follows the core NextLat interface: predict future hidden states from a current hidden state and teacher-forced next-token information. Let h_t be the final normalized hidden state of the causal model, and let e_{t+1} be the next-token input embedding after projection into hidden space. The auxiliary model concatenates h_t with a detached e_{t+1}, projects the pair into an auxiliary residual space, applies a bias-free three-layer GELU MLP, and predicts an update to the current hidden state. The predicted next state is

$$\hat{h}_{t+1} = \mathrm{RMSNorm}\left(h_t + \Delta_\phi(h_t, \mathrm{sg}(e_{t+1}))\right),$$

where sg denotes stop-gradient. Targets are detached, so the auxiliary loss shapes the main model only through the current hidden states that feed the predictor. We also mask auxiliary transitions across padding and packed-document boundaries, which is needed for our document-packed BabyLM training batches.

We train a two-step rollout. The first predicted hidden state is fed back into the same auxiliary model to predict one additional future hidden state under teacher-forced future embeddings. For each valid rollout step, we use Smooth L1 loss summed over hidden dimensions, then average over valid transitions and rollout steps. The auxiliary network has its own learning rate and weight decay, is used only during pretraining, and is discarded from the submitted checkpoint. Inference is therefore exactly the base causal model.

Spend the Wasted Epochs: Preference Post-Training from the Saturation Checkpoint for Sample-Efficient Language Modeling

Ujjwal Mishra
School of Computing
Indian Institute of Information Technology Una
Himachal Pradesh, India
ujjwalmishra238@gmail.com

Sudipta Roy[1,2]
[1]Jio University
Navi Mumbai, India
[2]Comp. Sci. & Engg., University of New South Wales
Bengaluru, India
sudiptaroy01@yahoo.com

Abstract

Sample-efficient pretraining under a developmental data budget, such as the ten million words and ten epochs of the BabyLM Strict-Small track, is usually approached by adding training signal: a new objective, optimizer, curriculum, or teacher. The budget itself receives less scrutiny. We show that much of it is spent after learning has stopped. A 33.0M-parameter GPT-BERT hybrid trained under the track rules saturates on downstream evaluations by roughly 70M words seen, about 70% of its permitted exposure, while training loss continues to fall; the final epochs improve fit, not competence. From this diagnosis we derive three design principles, timing, preservation, and alignment, and instantiate them as *Minimal-Pair Preference Optimization* (MPO): a short post-training phase, begun at the saturation checkpoint, that teaches the model to prefer real corpus sentences over length-preserving corruptions of them under a frozen-reference penalty. A controlled study of seven in-training interventions supports the principles: none improves on the baseline, and the unanchored ones damage the same cluster of long-distance dependencies first. MPO improves five of the seven official task averages, concedes under a point on the other two, is alone in repairing that cluster, and matches the fully trained baseline with 7.2 rather than 10 epochs; our MPO entry places fifth on the track's NLP average. Code and run logs are publicly available at https://github.com/Ujjwal238/MPO-BabyLM-Challenge-2026.

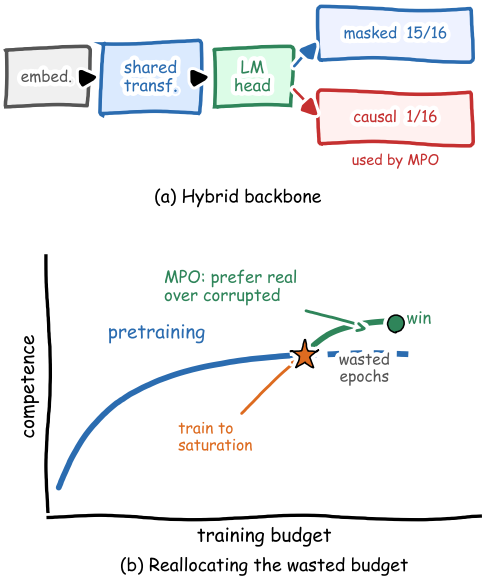


Figure 1: **The idea in one picture.** (a) The hybrid backbone: one shared transformer and a single language-model head, trained on both a masked objective ($15/16$ of batches) and a causal one ($1/16$). The causal mode is what MPO later exploits. (b) Standard pretraining saturates, then wastes its final epochs (dashed). Starting from that saturation checkpoint, MPO spends the leftover budget teaching the model to prefer real sentences over corrupted ones, and matches the fully-trained baseline in fewer epochs.

1 Introduction

A child hears no more than about 100 million words by early adolescence. A language model consumes several orders of magnitude more before it is called trained. The BabyLM Challenge asks what this gap conceals: how much linguistic competence can pretraining extract from a corpus a child could plausibly have heard? (Warstadt et al., 2023; Charpentier et al., 2025) We compete in the Strict-Small track of the 2026 edition (Choshen et al., 2026), the tightest form of the question: at most 10 million words of text, with the corpus traversed at most ten times.

Most entries respond by adding signal. New objectives, curricula, architectures, tokenizers, and distilled teachers arrive every year (Hu et al., 2024; Charpentier et al., 2025). We start from a different question: before adding anything, when does the existing budget do its work? Training a

Proceedings of the Second BabyLM Workshop, pages 646–658
October 29, 2026 ©2026 Association for Computational Linguistics

33.0M-parameter GPT-BERT hybrid (Charpentier and Samuel, 2024) under the track rules, we find that downstream competence saturates by roughly 70M words seen, seven of the ten permitted epochs, while training loss keeps falling. Those last epochs purchase a lower loss, not new competence. A per-phenomenon breakdown adds a second fact: a small cluster of long-distance dependencies is acquired last and stays fragile to the end.

The diagnosis dictates what a useful intervention must look like. It should arrive after saturation, where its gradient no longer competes with language modeling for capacity. It must be anchored, because fragile structure is what strong new gradients destroy first. And it should share the functional form of the evaluation, which for the core benchmarks is a likelihood comparison between minimally different strings. We call these the timing, preservation, and alignment principles.

Their instantiation is Minimal-Pair Preference Optimization (MPO): a short preference-optimization phase (Rafailov et al., 2023) that restarts from the saturation checkpoint and teaches the model to prefer real corpus sentences over length-preserving corruptions of them, with a frozen reference penalizing drift (Figure 1). The principles are tested rather than assumed. A controlled study of seven in-training interventions drawn from BabyLM entries and the broader pretraining literature confirms both predictions: none improves on the baseline, and the unanchored ones damage the fragile cluster first. MPO improves five of the seven official task columns and concedes under a point on the rest, is alone in repairing the fragile cluster, and matches the fully trained baseline after 7.2 rather than 10 epochs of exposure; our MPO entry places fifth on the track leaderboard's NLP average, the fully trained baseline ninth.

Beyond the method, the paper contributes the saturation diagnosis, cheap to run and decisive about where post-training signal belongs, and the controlled study of why published gains fail to transfer here.

2 Related Work

The BabyLM Challenge has run three times, and its findings reports serve as a census of what helps under developmental budgets (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). The recipes that succeed tend to change what the model predicts, not when. GPT-BERT, our backbone, won the 2024 edition by blending masked and causal prediction in a single transformer (Charpentier and Samuel, 2024). In the 2025 edition, reweighting the masking distribution as training progresses outperformed the official strict-small baseline (Edman and Fraser, 2025). BabyLlama distilled an ensemble of teachers into one small student and later served as an official baseline (Timiryasov and Tastet, 2023; Hu et al., 2024). All of these run as a single uninterrupted phase; where in the budget the signal lands is not itself a design variable. That variable is our subject.

Timing has been studied descriptively: Chang and Bergen (2022) chart when individual words are acquired during pretraining, regularly enough to compare against child norms, and the 2026 evaluation includes an age-of-acquisition task built on this lens (Choshen et al., 2026). We push the lens from describing when learning happens to acting on when it stops.

The interventions in our controlled study (§4.1) each arrive with a published pedigree: span corruption (Joshi et al., 2020), selective token losses that concentrate compute on learnable, worth-learning points (Mindermann et al., 2022; Lin et al., 2024), the Muon optimizer (Jordan et al., 2024), weight averaging in its stochastic and soup forms (Izmailov et al., 2018; Wortsman et al., 2022), and morpheme-aware segmentation from a child-inspired entry (Bölücü and Can, 2025). The reported gains come from other regimes: larger models, longer budgets, a previous edition's setting. Whether they hold under a 10M-word budget is empirical, and §4.1's answer is that none does. What informs our method is not the failures but the pattern in how they fail.

Preference optimization entered the toolkit as an alignment method: DPO tunes a model to prefer one completion over another under a KL penalty toward a frozen reference, with pairs elicited from human judgment (Rafailov et al., 2023). Its one BabyLM appearance so far targets dialogue: Padovani et al. (2025) tune a dialogue-pretrained model with DPO toward communicative continuations, gaining on a custom dialogue benchmark while conceding the standard suite. MPO retains the objective and replaces everything around it. Pairs are a corpus sentence and a synthetic corruption of it, so no annotator, reward model, or teacher is involved; the target is grammatical competence, not helpfulness; and the phase is placed by measurement at the saturation checkpoint, not appended to a finished model. The corruption de-

sign itself borrows from minimal-pair evaluation, in which BLiMP isolates one phenomenon per item by contrasting two strings that differ minimally (Warstadt et al., 2020). Learning from corruptions also has a pretraining pedigree: ELECTRA trains a token-level discriminator against sampled replacements throughout pretraining (Clark et al., 2020), where MPO compares whole sequences, after acquisition, under an anchor. We train on the form of comparison the benchmarks measure, computed entirely on text the rules already allow.

3 Method

3.1 The Hybrid Backbone and Training Setup

We train on the official Strict-Small corpus: 10M words across six sources, dominated by transcribed speech (CHILDES 28%, OpenSubtitles 23%, BNC spoken 8%, Switchboard) alongside Project Gutenberg (26%) and Simple Wikipedia (15%). A byte-level BPE tokenizer with vocabulary 16,384 is fit on the training split alone, and documents are packed into a single token stream read at sequence length 128.

The backbone is a GPT-BERT hybrid (Charpentier and Samuel, 2024): masked and causal prediction on a single transformer with one language-model head and shifted next-position targets for both modes,

$$\mathcal{L}(\theta) = \mathbb{E}_B\big[\, z\,\mathrm{CE}_{\text{mask}}(B;\theta) + (1-z)\,\mathrm{CE}_{\text{causal}}(B;\theta)\,\big], \quad (1)$$

with $z \sim \mathrm{Bernoulli}(15/16)$ per micro-batch. The model has 33,046,084 parameters (hidden 384, 12 layers, 6 heads, tied embeddings) and is trained for 10 epochs ($\approx 99M$ words seen) with AdamW, a cosine schedule, and a masking rate annealed from 0.30 to 0.15, on a single consumer laptop (M1, 16 GB, MPS). Appendix A lists every hyperparameter; none was tuned against the official evaluation.

Evaluation is the organizers' 2026 pipeline, unmodified. The zero-shot tasks compare the model's likelihoods over candidate strings: BLiMP (Warstadt et al., 2020) with its challenge supplement (Warstadt et al., 2023), EWoK (Ivanova et al., 2024), COMPS (Misra et al., 2023), entity tracking (Kim and Schuster, 2023), and GlobalPIQA (Chang et al., 2025). The suite adds reading-time prediction, a word age-of-acquisition comparison against child norms following Chang and Bergen (2022), and a (Super)GLUE finetuning battery (Wang et al., 2018, 2019). During development we gate decisions with the pipeline's fast subsets (200 BLiMP and 50 supplement items per subtask) and reserve the full server-scored suite for the submitted models (§4.3).

3.2 Diagnosing the Budget

Before proposing an intervention, we ask a prior question: over the course of training, when does the model stop *learning* as opposed to continuing to *fit*? Two observations about the baseline hybrid answer it, and they shape every design choice that follows.

The first observation concerns timing: evaluation saturates well before the budget is spent. Figure 2 plots fast BLiMP accuracy against words seen, overlaid with the masked training loss. The two curves dissociate over the final third of training. Accuracy rises steeply from 54.21 at 1M words to 68.74 at 50M and then goes flat: 70.13 at 70M words, 70.01 at 100M, a spread inside the seed-noise band of §4.2. Writing $A(w)$ for accuracy after w words seen, define the saturation point as the earliest milestone that no later milestone beats by more than a tolerance ε,

$$w^\star \;=\; \min\Big\{\, w \;:\; \max_{w'>w} A(w') - A(w) \;\le\; \varepsilon \Big\}. \quad (2)$$

On this trajectory $w^\star = 70$M for every ε below 1.2 points, seventeen times the seed band, so nothing about the choice of checkpoint hinges on the tolerance (Appendix C sweeps it). The training loss, in contrast, continues its steady descent over exactly this span (2.71 at 40M $\rightarrow$ 2.19 at 98M). The last $\sim$30% of the epoch budget therefore yields a lower training loss but no downstream competence: those epochs memorize a corpus already seen seven times rather than generalizing from it.

The second observation concerns what is fragile. A per-phenomenon breakdown (Appendix B) shows morphology mastered early and world knowledge near chance throughout, but singles out a small cluster of long-distance dependencies, filler–gap licensing, distractor agreement across a relational noun, and N-bar ellipsis, acquired last and the most fragile; we call this cluster $\mathcal{F}$. As §4.1 shows, $\mathcal{F}$ is not merely hard to learn: every failed intervention degrades it first, a sensitive probe of whether hard-won structure survives.

If late-budget training is wasted and competence is bottlenecked on specific fragile structures, then a candidate intervention should be judged by two questions: *where* in training does it inject its signal,

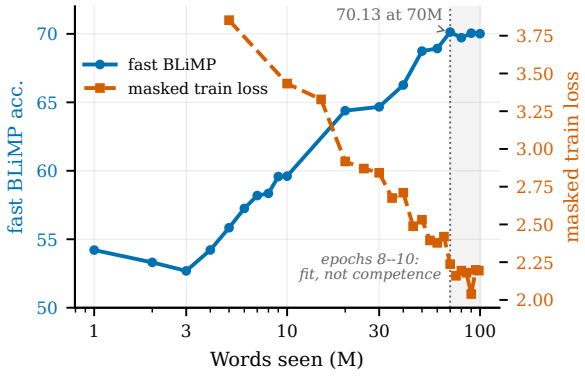


Figure 2: **The budget saturates early.** Fast BLiMP accuracy (left axis) reaches 70.13 by 70M words seen and is flat from there, while the masked training loss (right axis) keeps falling to the end of training. The final epochs lower the loss without improving competence.

and *what* does it risk destroying? These two questions organize our search (§4.1) and motivate our method (§3.3).

3.3 Reallocating the Wasted Budget with MPO

The diagnosis licenses three design principles. The first is *timing*: refinement signal should arrive only after acquisition has saturated, where it no longer competes with the language-modeling gradient for capacity (§4.1 tests the necessity of this). The second is *preservation*: additions must be bounded in how far they can move the model, because $\mathcal{F}$ is what a strong new gradient destroys first. The third is *alignment*: the signal should share the evaluation's functional form, which for BLiMP, its supplement, EWoK, and entity tracking is a relative-likelihood comparison between minimally different strings. Minimal-Pair Preference Optimization (MPO) instantiates all three (Figure 3).

Preference data comes from the corpus itself: no teacher, no reward model, no annotator, and no text beyond the ten million words already budgeted. A real sentence x^+ becomes $x^- = T(x^+)$ with T drawn uniformly from four edit operators, each length-preserving ($|T(x)| = |x|$): an adjacent-token swap, a distant-token swap, a function-word substitution, and a random content-token replacement. On *the cat sat on the mat* these produce, respectively, *the cat on sat the mat*, *the mat sat on the cat*, *the cat sat of the mat*, and *the cat lamp on the mat*. The four target complementary structure, from local word order to exactly the long-range configuration $\mathcal{F}$ tests: the adjacent swap probes local agreement, the distant swap and the function-word substitution probe the long-range dependencies of $\mathcal{F}$, and the random replacement supplies an easily rejected contrast that anchors the low end of the difficulty range. Appendix E ablates each operator alone and confirms the complementarity. The phase touches under 3% of the 834,593 corpus sentence units, derives nothing from any evaluation resource, and introduces no new word types; the corruptions are input noise of the same kind the masked objective has injected since the first step of pretraining, which already replaces sampled tokens at random.

Length matching makes the comparison fair by construction. Every edit operates in place on the tokenized sentence, so x^+ and x^- contain the same number of tokens and their summed causal log-likelihoods,

$$\ell_\theta(x) \;=\; \textstyle\sum_t \log \pi_\theta\big(x_t \mid x_{<t}\big), \qquad (3)$$

are directly comparable with no normalization to tune, mirroring the minimal-pair design of BLiMP itself. (The suite scores the hybrid through a masked backend, §4.3; what the phase shares with the evaluation is this comparison structure, not the scoring function.) Given pairs, we optimize the DPO objective (cf. Rafailov et al., 2023):

$$\mathcal{L}_{\text{MPO}}(\theta) = -\,\mathbb{E}\Big[\log \sigma\Big(\beta\big[\Delta_\theta(x^+) - \Delta_\theta(x^-)\big]\Big)\Big], \quad (4)$$

with $\Delta_\theta(x) = \ell_\theta(x) - \ell_{\text{ref}}(x)$. The reference is a frozen copy of the checkpoint the policy starts from, and in the reward-maximization problem that Eq. 4 solves implicitly, β is the coefficient on $\mathrm{KL}(\pi_\theta \,\|\, \pi_{\text{ref}})$ (Rafailov et al., 2023). Preservation therefore enters the objective directly rather than being assumed: likelihoods may reorder so that real sentences win, but drift from existing competence is penalized at rate $\beta = 0.1$. This property is what selects DPO among pairwise objectives: an anchor-free ranking loss would leave preservation to the learning rate alone, and the nearest anchor-free relative in this paper, the contrastive arm of §4.1, is the study's clearest failure. That arm also supplies the closest control: the same four corruptions, injected during acquisition without an anchor and under a loss free to reshape the model, cost seven BLiMP points. The corruption data was never the problem; what changed between failure and success is when the signal arrives and how tightly it is held. Because preference optimization alone slowly erodes general language-modeling quality, a failure mode

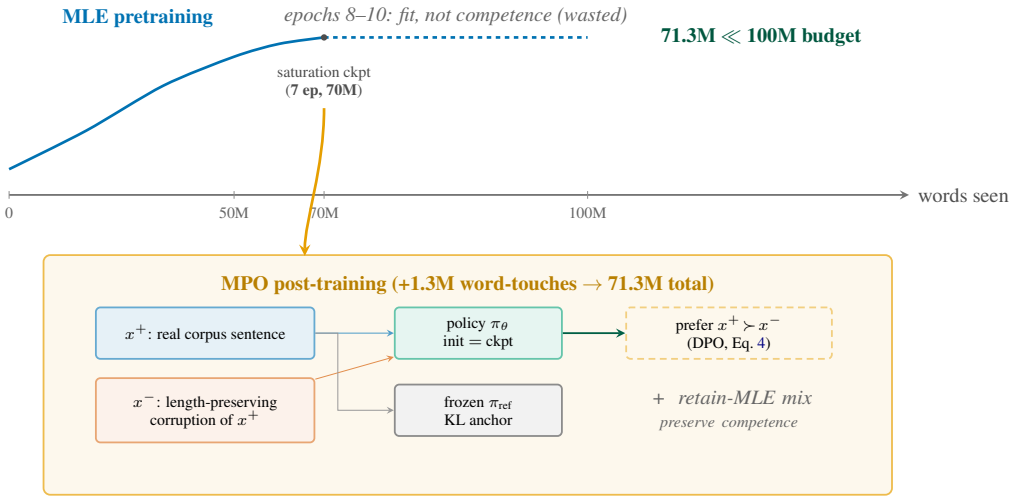


Figure 3: **Reallocating the wasted budget.** MLE pretraining saturates at the **7-epoch / 70M-word** checkpoint (dashed tail = the wasted epochs of Fig. 2). We fork there into **MPO**: the model is taught to prefer a real corpus sentence x^+ over a length-preserving corruption x^- of itself, KL-anchored to a frozen copy of the same checkpoint and mixed with a retain-MLE term (§3.3). Total exposure is 71.3M words, well under the 100M budget, and fewer than the fully-trained baseline.

documented in the RLHF literature, the step-t loss retains the pretraining signal,

$$\mathcal{L}_{\text{phase}}(\theta) = \mathcal{L}_{\text{MPO}}(\theta) + m_t\,\mathcal{L}(\theta), \quad m_t \sim \text{Bernoulli}(0.5), \quad (5)$$

one ordinary hybrid micro-batch (Eq. 1, post-anneal masking rate) at unit weight.

The timing principle fixes the starting point. Policy and reference initialize from the 70M-word saturation checkpoint, not the fully-trained model; running the identical phase from the 99M-word checkpoint reaches the same BLiMP (§4.2) at 29M additional words of budget, direct evidence that epochs eight through ten contributed nothing the method requires. Starting earlier does not substitute for acquisition: from the pre-saturation 30M checkpoint the same phase leaves fast BLiMP at 64.96, barely above that checkpoint's own 64.67. The phase refines competence; it does not create it. Total exposure is 70M plus roughly 1.3M phase word-touches, counted conservatively: every token of both members of every pair and of the retained hybrid batches is charged against the budget, for 71.3M in all, well inside the 100M cap a 10-epoch run would exhaust (Appendix D itemizes the accounting).

The configuration is deliberately conservative: 1,500 AdamW steps at learning rate 10^{-5}, 16 pairs per step, cosine decay, clipping at 1.0, about 22 minutes on the same laptop (roughly 3% of the pretraining wall-clock), checkpoints every 250 steps. §4.2 shows a plateau across those checkpoints rather than a single favorable snapshot, and shows that a stronger preference objective improves its own training metrics while degrading the evaluation; the restraint is functional rather than stylistic.

4 Results

4.1 A Controlled Study of In-Training Interventions

The principles of §3.3 make falsifiable predictions: if timing binds, no intervention injected *during* acquisition should surpass the baseline; if preservation binds, unanchored signals should damage $\mathcal{F}$ before anything else; and if configuration sensitivity is as severe as the alignment principle assumes, gains imported from neighboring setups should not transfer. We test all three with a controlled study of seven in-training interventions spanning the sites where signal enters the update rule (objective, optimizer, weights, masking policy, token-level loss, tokenizer), each run holding budget, data order, and seed fixed while changing one component, and each gated by the same fast BLiMP-and-supplement evaluation. An eighth planned intervention, ensemble self-distillation, was excluded by hardware constraints (Table 2). The scope is deliberate: the predictions concern how the update rule reshapes the model, so the study varies that rule while holding the data stream fixed; data-side methods such as curricula lie outside it, and under a corpus traversed ten times, replay is the default rather than a distinct option. Table 2 collects the

Model	NLP	BLiMP	Supp.	EWoK	Entity	PIQA	COMPS	GLUE
Official baselines								
GPT-2 (BabyLM 2026)	48.99	65.23	57.25	50.63	19.10	35.09	51.81	63.80
Interaction (BabyLM 2026)	49.04	63.07	58.13	51.45	19.62	36.14	51.29	63.62
Other Strict-Small submissions ranked above ours at submission close								
BabySteps-MurphysLaw (rank 1)	**53.59**	71.60	63.93	51.94	27.95	36.15	53.13	**70.44**
wwm_curriculum (rank 2)	52.97	67.2	56.01	**56.07**	**28.45**	39.67	53.57	69.79
Wordpiece-24-4 (rank 3)	52.85	70.20	**66.52**	52.10	21.82	36.08	54.04	69.18
RecGPT-10M (rank 4)	52.40	**73.11**	61.73	52.62	16.59	**40.68**	**55.43**	66.64
Ours								
GPT-BERT (10 ep)	51.61	69.83	63.77	52.18	21.06	33.69	52.23	68.50
+ MPO (7.2 ep)	51.91	69.85	64.84	51.33	21.79	34.69	51.80	69.03

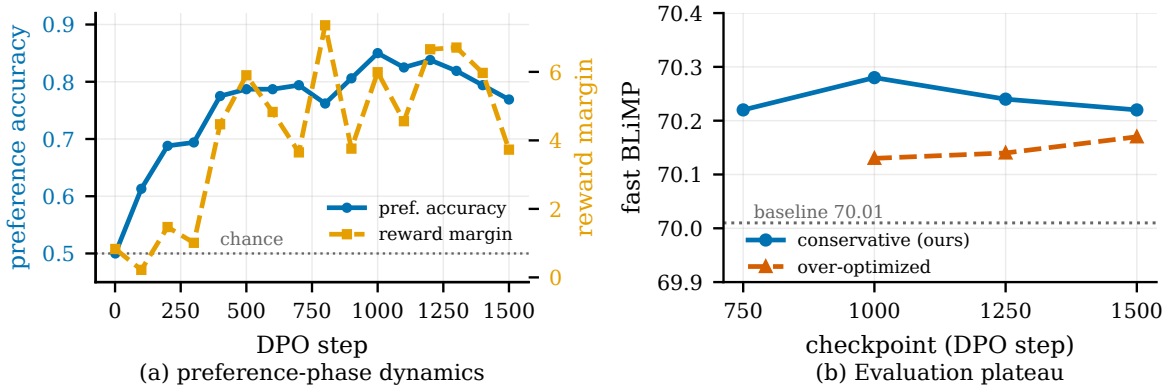
Table 1: **Main results on the official BabyLM 2026 Strict-Small metrics** (Overall NLP average and its seven task components; AoA and reading omitted, since AoA rounds to zero for nearly every submission under the corrected scorer, the GPT-2 baseline at -12.15 being the exception). Entity Tracking is the bias-filtered version; PIQA is GlobalPIQA. Comparison rows are the entries ranked above ours at the close of the submission period (2026-07-20 AoE, 118 strict-small entries); the board remains open, so entries arriving after the deadline are not counted here. Our entries rank fifth and ninth on the NLP average, the board's sort key; **bold** marks the best value per column among the tabulated entries. MPO improves over our fully-trained baseline on five of seven columns while using 7.2 rather than 10 epochs.

Intervention	Site	BLiMP	Suppl.
Baseline hybrid	—	70.01	65.20
Contrastive grammat.†	objective	62.87	64.80
Span masking	objective	68.61	60.00
Muon optimizer	optimizer	65.12	61.20
Checkpoint averaging	weights	69.96	65.20
Adaptive masking	masking	70.04	63.60
Selective LM	token loss	69.26	58.00
Morph.-aware tokenization	tokenizer	68.72	64.00
MPO (§3.3)	post-training	**70.24**	**65.60***

Table 2: **Eight interventions, one improvement.** Fast BLiMP and BLiMP Supplement under an identical budget, recipe, and seed; every intervention changes one component relative to the baseline hybrid. †Contrastive training crashed at 77M words and is compared at the matched 70M checkpoint (reference 70.13/65.60). *Supplement varies with seed on this 50-item-per-subtask fast set (§4.2). A planned self-distillation intervention never produced a comparable run: three resident fp32 teachers exceed 16 GB, which we report as a hardware exclusion rather than a result.

outcomes, and they fall where the principles predict. No in-training intervention improves on the baseline.

The objective-side interventions regress most severely. The contrastive grammaticality loss costs seven BLiMP points while its own auxiliary loss collapses toward zero, a shortcut at the task's expense; span masking, a harder reconstruction task per epoch, underfits the fixed budget. Muon degrades both, a higher training loss than AdamW at equal words seen and weaker transfer besides, a mismatch with this budget-bound regime rather than a verdict on the optimizer. Averaging late checkpoints has no effect; by 80M words they occupy a shared basin. Adaptive masking arrives with a reported one-point BLiMP gain on a DeBERTa encoder (Edman and Fraser, 2025) and transfers to our hybrid as a tie. Morphology-aware tokenization arrives reporting EWoK gains of roughly twenty percent and entity-tracking gains of roughly forty percent at this budget (Bölücü and Can, 2025) and inverts under our recipe: EWoK falls below chance, to 47.73. Selective language modeling (Lin et al., 2024; Mindermann et al., 2022) holds BLiMP but gives up seven supplement points: excess-loss selection de-weights the most predictable text, child-directed dialogue, and with it the pragmatics the supplement measures. The regressions also share a structure, the preservation prediction made visible. In Figure 5, four mechanistically unrelated interventions degrade the same two filler–gap phenomena first, by 15 to 60 points, while typically gaining on easier local subtasks. Long-distance dependencies are both the last structure acquired and the first sacrificed when the signal is perturbed; a viable intervention must leave them alone. Nor does greater intensity rescue a proxy, which is why preservation requires an explicit anchor: the contrastive loss reaches near zero as BLiMP falls, Muon fits the MLM objective harder and generalizes worse, and §4.2 shows the same trade-off within our own method. At 33M parameters and 10M words there is no spare capacity, and a proxy pursued past the



(a) preference-phase dynamics (b) Evaluation plateau

Figure 4: **MPO dynamics and the eval plateau.** (a) Preference accuracy rises to $\sim$0.85 and the reward margin grows: the objective is learned. (b) Fast BLiMP across checkpoints: the conservative run (ours, β=0.1) holds a plateau above the baseline, ruling out checkpoint-selection effects, while the over-optimized variant (β=0.2, harder negatives) plateaus below it despite higher preference accuracy.

anchor's tolerance is paid in competence. Portability fares no better: the masking-policy result does not survive the change from encoder to hybrid, and the tokenizer result inverts under a different recipe and evaluation pipeline. At this scale results are sensitive to interactions among architecture, recipe, and metric; re-validation under one's own budget is a prerequisite. The study thus confirms each prediction: the region of the design space the principles exclude, in-training injection of unanchored signal, is empty as far as this search reaches, and the region they license contains MPO. Appendix F gives a mechanistic reading of the shared pattern in terms of basin geometry and gradient competition.

Init	Negatives	β	Seed	BLiMP	Suppl.
no preference phase (baseline)				70.01	65.20
70M	mixed	0.1	42	**70.24**	**65.60**
70M	mixed	0.1	43	70.14	64.40
70M	mixed	0.1	44	**70.28**	63.60
30M	mixed	0.1	42	64.96	62.80
99M	mixed	0.1	42	70.26	62.40
70M	hard	0.1	42	70.16	64.40
70M	hard	0.2	42	70.14	64.00

Table 3: **MPO ablations** (fast BLiMP / Supplement at the step-1250 operating point). Seeds vary the entire phase, not just data order. The two init rows bracket the saturation checkpoint: from 99M the phase matches the 70M-init BLiMP while spending 29M more words of budget, and from the pre-saturation 30M checkpoint it stays near that checkpoint's own baseline (64.67/62.00), five points below every saturated init. "Hard" reweights corruptions toward the more adversarial edit types; together with the larger β it raises preference accuracy (0.85→0.90) and lowers evaluation scores. The supplement column is measured on 50 items per subtask and fluctuates accordingly.

4.2 MPO

Table 1 reports both submitted models on the official Strict-Small metrics. MPO improves on the fully-trained baseline in five of seven task columns, gives back under a point on the other two (EWoK and COMPS), ties reading (0.27 vs. 0.29, both near zero), and lifts the NLP average from 51.61 to 51.91 at 7.2 rather than 10 epochs ($71.3M$ vs. $99M$ words seen). The two entries place fifth and ninth among 118 on the NLP average, the leaderboard's sort key,[1] We sit near the top of the field on the BLiMP supplement; entity tracking and the world-knowledge tasks (EWoK, GlobalPIQA) are where we trail the leaders, a gap no measurement choice narrows. Across three seeds, fast BLiMP holds at 70.22 ± 0.07 and, uniquely among every intervention we tried, $\mathcal{F}$ *improves* (*wh_vs_that_with_gap* 71.5→75.5, consistent across seeds and configurations); the full-suite gains on entity tracking, GlobalPIQA, and GLUE are from the submitted single-seed run (see Limitations).

The pattern across columns is inconsistent with teaching to the test. The benchmark nearest the training signal moves least: full BLiMP gains 0.02. The clear gains appear where the corruptions never reach, dialogue pragmatics in the supplement, entity tracking, GlobalPIQA, and finetuned GLUE, and Table 3 shows that pushing the proxy harder degrades the evaluation; a model exploiting the format would show the opposite, gains only where the

[1]Model and all checkpoints are public at https://huggingface.co/Ujjwal101/ujjwal-very-bored. Standings are those at the close of the submission period, 2026-07-20 AoE; the board remains open.

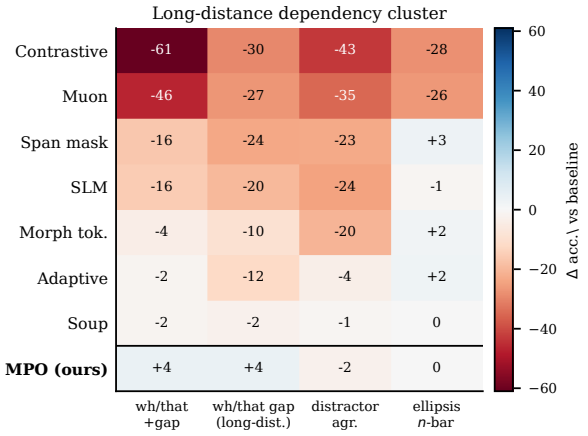


Figure 5: **Every failed intervention degrades the same cluster; only MPO repairs it.** Change in fast-BLiMP accuracy on the fragile long-distance dependency cluster $\mathcal{F}$ relative to the Phase-1 baseline, per intervention. Red = degraded, blue = improved. The objective- and optimizer-side interventions (Contrastive, Muon) reduce filler–gap licensing by 40–60 points; MPO (bottom) is the only intervention that *improves* it.

format matches.

The direction of the small losses follows from what the signal contains. The corruptions manipulate structure, never facts, so the phase sharpens sensitivity to well-formedness but teaches nothing about the world; EWoK and COMPS contrast plausibility rather than grammar, and mass reallocated toward well-formedness leaves such contrasts untouched or marginally taxed. Both drops are under a point, and §3.2 shows world knowledge was never acquired in the first place: nothing there for refinement to refine.

Four alternative explanations could undermine the result; Table 3 collects the runs that test them. *Seed sensitivity.* Seeds 43 and 44 leave fast BLiMP at 70.22 ± 0.07, the $\mathcal{F}$ repair recurs in every saturation-init run, and the 50-item supplement fluctuates as noise. *Checkpoint selection.* Figure 4b shows checkpoints 750 through 1,500 spanning 70.22 to 70.28, a plateau. *Initialization.* The starting point matters in both directions. From the 99M-word checkpoint the identical phase reaches 70.26, the same BLiMP for 29M more words; from the pre-saturation 30M checkpoint it reaches 64.96, a small gain over its own 64.67 and five points short of every saturated-init run. Refinement tracks its starting competence and substitutes for none of it; the diagnosis identifies a causal precondition, not a descriptive landmark. *Objective strength.* Hard negatives at $\beta = 0.1$ give 70.16; β=0.2 gives 70.14 while preference accuracy rises from 0.85 to 0.90, the proxy improving as the evaluation deteriorates, the §4.1 trade-off within our own method. The conservative configuration is not an untuned default but the optimum here.

4.3 Submission

Both models are separate leaderboard entries, the fully trained baseline and the MPO model; each is the organizers' server-scored collation over the complete suite with the required checkpoints, so no number in Table 1 is from our own harness. The hybrid uses the pipeline's masked next-token (`mntp`) backend for the zero-shot tasks and its causal backend for entity tracking and age of acquisition; the MPO entry declares its true 71.3M-word exposure against the 100M cap. The two entries hold fifth and ninth among 118 on that same NLP average; the board is live but the within-submission comparison is not, since the same 33.0M model appears twice, with and without the reallocated tail, every difference a 22-minute phase.

5 Conclusion

Under a fixed developmental budget, the cheapest resource is the one already being wasted. Our diagnosis found the final third of the Strict-Small epoch budget buying a lower training loss and no further competence, the two curves parting company exactly where the schedule keeps spending. That gap is the opening our method works through. Three principles, timing, preservation, and alignment, turn the observation into a recipe, and both a controlled study of seven interventions and the live leaderboard bear it out. The recipe is short: train until the evaluation saturates (Eq. 2), then spend what remains on anchored, evaluation-aligned refinement from that checkpoint, held close enough that the hard-won structure survives. Nothing in it is specific to our backbone, and both steps should carry to any model whose budget outlives its learning. We resist a strong developmental reading, though the shape is suggestive: exposure first, contrast after. At this scale, when the corpus is exhausted, the budget need not be.

Limitations

Our study is deliberately narrow in scope: a single 33.0M-parameter architecture, one English corpus, a single 10M-word budget. Whether the diagnosis and MPO carry to other backbones, larger budgets,

or other languages remains to be shown, and the full-suite evaluation used one seed, though the two claims we rest on both held across three.

References

Necva Bölücü and Burcu Can. 2025. A morpheme-aware child-inspired language model. In *Proceedings of the First BabyLM Workshop*, pages 279–287, Suzhou, China. Association for Computational Linguistics.

Tyler A. Chang, Catherine Arnett, et al. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. ArXiv:2510.24081.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. ArXiv:2602.20092.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations (ICLR)*.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. ArXiv:2405.09605.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Not all tokens are what you need for pretraining. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Sören Mindermann, Jan Brauner, Muhammed Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. 2022. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 15630–15649. PMLR.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Francesca Padovani, Bastian Bunzeck, Manar Ali, Omar Momen, Arianna Bisazza, Hendrik Buschmeier, and Sina Zarrieß. 2025. Dialogue is not enough to make a communicative BabyLM (but neither is developmentally inspired reinforcement learning). In *Proceedings of the First BabyLM Workshop*, pages 421–435, Suzhou, China. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 279–289, Singapore. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 23965–23998. PMLR.

Appendix

A Training Hyperparameters

Table 6 lists both phases end to end. The pretraining configuration stays deliberately close to the released GPT-BERT recipe, and the MPO phase adds no hyperparameter search beyond the ablations of §4.2. Every run in the paper, training, refinement, and evaluation alike, executed on the same laptop.

Two pretraining choices deserve comment. The 128-token sequence length is a hardware accommodation: the backbone's relative-position attention leaves the fast Apple-silicon (MPS) execution path above 128 tokens, and the corpus, dominated by transcribed dialogue, loses little at this length. The effective batch of 128 sequences is reached through gradient accumulation (32×4), and the masking rate anneals from 0.30 to 0.15, so early epochs face a harder reconstruction problem than late ones.

The MPO phase runs at a learning rate fifty times below the pretraining peak, under a tighter gradient clip (1.0 against 2.0). Both choices serve the preservation principle of §3.3: the phase is meant to reorder likelihoods around an already competent solution, not to travel far from it. Checkpoints every 250 steps yield the trajectory of Figure 4b, from which step 1,250 is the operating point; the full phase costs 22 minutes against 12.5 hours of pretraining.

B Per-Benchmark Breakdowns

Phenomenon	10M	30M	50M	70M	99M
Anaphor number agr.	86.0	85.5	88.5	92.5	94.0
Det.–noun agr. (1)	80.5	86.0	88.0	90.5	90.0
Irreg. plural SV agr. (1)	69.5	71.5	72.5	74.5	73.5
Ellipsis N-bar (2)	53.5	83.0	91.0	91.5	91.5
Wh/that with gap	12.5	45.0	67.0	73.5	71.5
Distractor agr. (rel. noun)	22.5	25.0	32.0	57.5	62.0
Wh/that gap, long-dist.	0.5	10.0	26.0	34.0	32.5

Table 4: Acquisition timing on the baseline run: fast BLiMP subtask accuracy by words seen. Local morphology (top) is largely in place by 10M words; the long-distance cluster $\mathcal{F}$ (bottom) is acquired last and never fully.

Table 4 gives the acquisition side of the diagnosis. Local morphology is essentially in place by 10M words, a single epoch, and drifts upward from there. The long-distance cluster is different in kind: filler–gap licensing crosses 50% only after 30M words, its long-distance variant never reaches 35%, and distractor agreement is still climbing when the budget ends, the one exception to the aggregate plateau. World knowledge remains at chance throughout: fast EWoK stays within 47–53 across all nineteen checkpoints and ends at 52.18 on the full set.

	wh gap	long-d.	distr.	ellipsis
Baseline	71.5	32.5	62.0	91.5
Contrastive obj.	10.5	3.0	19.0	63.5
Muon	25.0	5.5	27.0	65.0
Span masking	55.0	9.0	39.0	94.5
Selective LM	56.0	12.5	37.5	90.5
Morph. tokenization	67.5	23.0	41.5	93.5
Adaptive masking	70.0	20.5	58.5	94.0
Checkpoint avg.	70.0	30.0	61.0	91.5
MPO (ours)	**75.5**	**36.5**	59.5	91.5

Table 5: Absolute fast BLiMP accuracy on the fragile cluster $\mathcal{F}$ (the values behind Figure 5): wh gap = *wh_vs_that_with_gap*, long-d. = its long-distance variant, distr. = *distractor_agreement_relational_noun*, ellipsis = *ellipsis_n_bar_2*. MPO is the only intervention that improves the two filler–gap columns.

Two details of the table sharpen the picture. The hardest phenomena do not start at chance but below it: after one epoch the model prefers the corrupted member on *wh*-licensing items (12.5) and almost categorically on their long-distance variants (0.5), so acquisition here means unlearning a systematic early preference, not filling a gap in knowledge. Phenomena also differ in trajectory shape: N-bar ellipsis is acquired in a single burst between 10M and 30M words (53.5 to 83.0) and is stable thereafter, whereas distractor agreement climbs steadily to the end. The 70M column is the saturation point $w^\star$ of Eq. 2; the movement that remains past it is confined to this one phenomenon and gains too little to lift the aggregate.

Table 5 lists the absolute accuracies behind the deltas of Figure 5. All rows are final checkpoints except the contrastive grammaticality objective, which is measured at its 70M-word A/B point, where that run ended (§4.1).

Read column-wise, the table grades the interventions by how strongly they perturb the training signal. The objective-level change is the most destructive, leaving *wh*-licensing at 10.5, below even the one-epoch value of Table 4; the optimizer change follows at 25.0; the masking, token-selection, and tokenizer variants land between 55.0 and 67.5; and checkpoint averaging, which perturbs no gradients at all, sits at the baseline. N-bar ellipsis is the robust member of the cluster, degraded materi-

ally only by the two most destructive interventions (63.5 and 65.0 against 91.5). MPO's repair is confined to the two filler–gap columns: its distractor-agreement value (59.5 against 62.0) is within the sampling noise of the 200-item fast sets, and even the repaired long-distance variant remains below chance. The cluster is improved, not solved.

C Saturation-Point Sensitivity

Pretraining	
Hidden / layers / heads / FFN	384 / 12 / 6 / 1,280
Parameters (tied embeddings)	33,046,084
Vocabulary (byte-level BPE)	16,384
Sequence length	128
Batch × grad. accum.	32 × 4 (eff. 128)
Optimizer	AdamW, lr 5×10^{-4}
(β_1, β_2) / weight decay	$(0.9, 0.98)$ / 0.1
Schedule	cosine, 2% warmup
Gradient clip	2.0
Masked-mode share z	15/16 of micro-batches
Masking rate	$0.30 \rightarrow 0.15$ (annealed)
Epochs / words seen	10 / ≈99M
Seed	42
Wall clock	12 h 27 m
MPO phase	
Init. / reference	70M ckpt (frozen copy)
β	0.1
Steps	1,500 (selected: 1,250)
Optimizer	AdamW, lr 10^{-5}, cosine
Pairs per step	16
Corruption mix	uniform over the four edits
Retained-MLE mix	$p = 0.5$, one hybrid batch
Gradient clip	1.0
Checkpoint interval	250 steps
Seed	42
Wall clock	22 min

Table 6: Complete hyperparameters for both phases. All runs use one M1 laptop (16 GB, MPS). GLUE fine-tuning uses the official recipe at sequence length 128, batch 32 (16 for BoolQ and MultiRC), 10 epochs (30 for WSC), with pipeline defaults otherwise.

The tolerance ε enters the diagnosis only through Eq. 2, and the milestone grid is spaced at 10M words past the first epoch, so $w^\star$ moves in 10M quanta. Table 7 sweeps it. No later checkpoint exceeds the 70M value, so $w^\star = 70$M down to $\varepsilon = 0$; the first alternative selections appear at $\varepsilon = 1.20$ (60M) and $\varepsilon = 1.39$ (50M), thresholds seventeen and twenty times the ±0.07 cross-seed deviation of §4.2. A tolerance set anywhere within an order of magnitude of the measurement noise selects the same checkpoint, and even the first mis-selection, 60M, lies past the acquisition of every phenomenon in Table 4 except distractor agreement.

Tolerance ε	$w^\star$ (Eq. 2)
$[0, 1.20)$	70M
$[1.20, 1.39)$	60M
$[1.39, 3.87)$	50M
$[3.87, 5.46)$	40M

Table 7: Sensitivity of the saturation point to the tolerance. The selection used throughout the paper (70M) holds for every ε below 1.20 fast-BLiMP points.

D Exposure Accounting

Table 8 itemizes every token the submitted MPO model saw. Both members of every preference pair are charged in full, although x^- is an edit of an already counted corpus sentence, and the retained hybrid micro-batches (Eq. 5) are charged at their full sequence length. Component rows are expectations under the sampling distribution (24,000 pairs at the bank's mean sentence length of 13.2 tokens; a Bernoulli(0.5) retain draw per step); the phase total is the realized value logged by the training loop. Tokens convert to words at the corpus ratio of 0.613. No text outside the ten-million-word corpus enters at any point.

Component	**Basis**	**Words**
MLE pretraining to $w^\star$	7 epochs, logged	70.0M
Preference pairs (x^+, x^-)	48,000 sentences	≈0.39M
Retained hybrid batches	≈750 batches	≈0.94M
Phase total (realized, logged)	training loop	1.28M
Total exposure		**71.3M**
Words-seen cap (10 epochs)		100M
Unique-text budget		10M

Table 8: Exposure accounting for the submitted MPO model. Total exposure is 71.3M of the 100M words-seen cap; the unique-text budget is untouched beyond the official corpus.

E Corruption Operators and Per-Operator Ablation

The four corruption operators behind MPO's preference pairs act on the tokenized sentence, preserve its length, and never touch the start-of-sequence position or padding. The adjacent swap exchanges a random pair of neighboring tokens. The distant swap exchanges two positions drawn independently from the whole sentence. The function-word substitution replaces one closed-class token with a different one drawn from the corpus's 120 most frequent non-special tokens, which in English are overwhelmingly determiners, prepositions, auxiliaries, and pronouns. The random replacement sub-

stitutes a uniformly drawn vocabulary token (§3.3 gives worked examples of all four). MPO samples the four uniformly; the hard-negative ablation reweights toward swaps and closed-class substitutions and away from the trivially detectable random replacement.

Table 9 reruns the identical phase with each operator alone (70M init, $\beta = 0.1$, seed 42, step-1250 canary). Three regularities emerge. First, aggregate transfer tracks the operator's proximity to the evaluation's own contrast type: the closed-class substitution, the edit nearest BLiMP's morphosyntactic minimal pairs, is the strongest single choice; the swaps sit near the mixture; and the random replacement, the edit MPO already treats as trivially detectable, transfers worst. Second, the filler–gap repair follows a different ordering: the order-perturbing swaps repair *wh_vs_that_with_gap* most strongly (79.0 against the baseline's 71.5), the substitution repairs it partially, and the random replacement not at all. The operators are complementary rather than redundant, as §3.3 intends. Third, every variant reaches the same preference accuracy (0.85–0.86): the objective is equally learnable from any operator, and the differences lie entirely in transfer, the proxy–evaluation dissociation of §4 in a third form. No single operator dominates: the substitution wins the aggregate, the swaps win the cluster repair, and only the mixture holds the supplement's best value. The submitted model keeps the untuned uniform mixture.

Operator	BLiMP	Suppl.	wh gap	long-d.	p-acc
Adjacent swap	69.85	64.00	**79.0**	**37.5**	0.86
Distant swap	70.08	64.80	**79.0**	**37.5**	0.85
Function-word sub.	**70.54**	65.20	75.0	35.5	0.86
Random replace	69.83	64.00	72.5	32.5	0.86
Mixture (submitted)	70.24	**65.60**	75.5	36.5	0.85
No phase (baseline)	70.01	65.20	71.5	32.5	—

Table 9: Per-operator ablation: the MPO phase run with a single corruption operator, all else identical to the submitted configuration. Columns as in Table 5; **bold** marks the best value per column.

F A Mechanistic Reading

A mechanistic reading consistent with the outcomes of §4.1 and §4.2 is available. By 80M words the late checkpoints share a basin (the checkpoint-averaging result of Table 2), so the saturated model occupies a region of near-equivalent solutions; a small-learning-rate phase held to a frozen reference moves within that region, reordering likelihoods without leaving it, the same role anchoring plays in continual learning (Kirkpatrick et al., 2017). An unanchored signal injected during acquisition instead competes with the language-modeling gradient for the same limited capacity, and what it sacrifices first is whatever that gradient holds most tenuously, the long-distance cluster. We offer this as an organizing account rather than a demonstrated mechanism; the observations it organizes are Tables 2 and 3.

Exploring Linguistic Parameters in LMs

Cristiano Chesi, Achille Fusco, Albina Mukusheva, Guido Formichi, Sarah Rossi, Hanna B. Cygan, Tommaso Sgrizzi, Asya Zanollo
NeTS Lab @ IUSS Pavia, Piazza della Vittoria 15, Pavia, Italy
Correspondence: cristiano.chesi@iusspavia.it

Abstract

This work places the BabyLM Multilingual perspective within the Principles and Parameters framework (Chomsky, 1981; Yang et al., 2017). First, we compared tokenization and architectural strategies that make the incremental nature of language acquisition and processing more or less transparent (recurrent versus attention-based architectures). We then assessed our trained models with respect to specific linguistic generalizations, including morphological decomposition, learning trajectories, and binary parametric options. We found that (i) morphologically informed tokenizers yield consistent benefits in both learning trajectories and performance on downstream minimal-pair tasks; (ii) recurrent architectures are substantially more data-efficient, converging earlier and achieving slightly better performance on downstream tasks; and (iii) the parameter-setting objective yields modest improvements, though not across all expected linguistic domains.

1 Introduction

In the context of Multilingual Language Modeling, a single Language Model (LM) must internally represent the properties of different languages. Adopting a perspective of cognitive plausibility, this means that the same architecture must be able not only to represent *in principle* any grammar, but also to accommodate multiple grammars simultaneously and thus successfully handle points of variation among them. In the generative linguistics tradition, a crucial notion in this respect is that of *parameter*, which differs from that assumed in language modeling. In the Principles and Parameters (P&P) framework (Chomsky, 1981; Yang et al., 2017), a *linguistic parameter* (*LiP*) is conceived as a categorical switch between two structurally possible options compatible with both *universal principles* and the available *linguistic evidence*. For the three languages under investigation (ENG - English, NLD - Dutch, and ZHO - Chinese), several relevant LiPs can be defined. Following Roberts (2019), a LiP can be located at the macro, meso, or micro level, depending on the affected heads and the number of lower-order options it entails. Simplifying (see § 2 for details), *Head-directionality* can be considered a macro-parameter with clear effects on word order. This can be observed by contrasting the Dutch sentence *Je kunt er toch [$_O$ ééntje] [$_V$ maken]* with its English counterpart *You can just [$_V$ make] [$_O$ one]*. The VO order is reversed as a consequence of a minimal structural parameter distinguishing English from Dutch. In this work, we explore this parametric intuition: beyond the structural machinery that, in a pre-theoretical sense, merges one token with another, the model should also internalize the higher-level generalizations captured by linguistic parameters (Chesi, 2023). The underlying assumption is that the computational machinery processing ENG, NLD, and ZHO is identical across languages, corresponding to a shared *Universal Grammar* (Chomsky, 1981) backbone, and that only a small number of parameters need to be set on top of this machinery to systematically alter the model's expectations. In the example above, the head-directionality LiP setting determines whether the model expects a direct object to precede or follow its predicate. Building on this intuition, we approach language modeling by introducing an additional training objective: parametric discrimination across languages (LiP discovery). The architectural variable we focus on is *processing incrementality*. Incremental processing is naturally aligned with recurrent approaches (Gu and Dao, 2024; De et al., 2024), but less so with attention-based models, particularly those relying on key–value caching, such as the BabyLM baselines. A further unavoidable challenge concerns the choice of linguistic units for modeling linguistically plausible parametric distinctions. The multilingual track of the BabyLM 2026 Challenge is

Proceedings of the Second BabyLM Workshop, pages 659–677
October 29, 2026 ©2026 Association for Computational Linguistics

particularly demanding, as the inclusion of Chinese forces us to abandon morphologically grounded, split-based segmentation procedures (Fusco et al., 2025) and instead adopt either a morphologically agnostic BPE/Unigram (bottom-up) approach or a boundary-discovery approach that treats spacing only as a soft segmentation cue.

In the following sections, we briefly explore the notion of LiPs (§ 2) and issues related to tokenization (§ 3). We then discuss specific architectural choices (§ 4) that allow us to implement supervised or unsupervised control of LiPs. In § 5, we present experiments comparing the performance of different tokenizers using the attention-based GPT-2 model, and compare this BabyLM baseline with Recurrent Neural Networks (RNNs) from the Hawk family (De et al., 2024). We also discuss various attempts to integrate parametric switches into a theory-driven computational architecture. The main results (§ 6) indicate that RNNs are more budget-efficient and slightly outperform attention-based baseline models on certain downstream evaluation tasks selected for the BabyLM Multilingual Challenge. Moreover, we find that tokenization strategy matters, with morphologically consistent approaches yielding better results on agreement phenomena. Overall, LiP supervision has only a modest effect on overall model performance, but its impact is more pronounced for specific phenomena, such as islands (§ 7).

2 Linguistic Parameters

A common criticism in linguistics is that Large Language Models (LLMs) flatten sentence structure into string-based representations, whereas humans are biased toward representing sentences as hierarchical phrase structures (Kodner et al., 2023). A related but poorly understood issue is how multilingual LMs represent cross-linguistic variation. The fact that the so-called *curse of multilinguality*—namely, the observation that, beyond a certain point, adding multilingual data appears to degrade model performance—can be mitigated by increasing model size or capacity (Conneau et al., 2020) casts further doubt on whether LLMs learn the relevant linguistic generalizations. Although the P&P framework developed in the 1980s has been substantially reformulated within the Minimalist perspective (Roberts, 2019), one of its enduring ideas is that linguistic variation is constrained: alongside a set of universal principles, each adult grammar consistently selects a particular configuration of LiPs that may favor one word order over another. Simplifying a long-standing debate, we can identify two aspects common to linguistic theories of parameters: (i) the binary nature of parameters and (ii) the implicational nature of LiPs. Parameters are conceived as a finite set of binary options that describe the structural properties of a language, each minimally associated with the presence or absence of a functional feature that licenses particular configurations. For the child, language acquisition consists in setting the relevant parameters on the basis of the input (primary linguistic data) during a critical period (Chomsky, 1986; Guasti, 2017). More recent proposals argue that parameters emerge from the interaction among minimal universal constraints, economy principles (the so-called *third factor*), and input data (Roberts, 2019). We now consider three representative examples of classical LiPs examined in this work: *Head Directionality* (*HD*), *Verb Second* (*V2*), and *Filler–Gap Dependency* (*WH*).

HD parameter Assuming that a *head* is the lexical item that encodes the selectional properties of a constituent (e.g., the lexical verb, V), heads merge with their obligatory complements (e.g., the direct object (O), *a sandwich*, in $[_{VP}\ eat_V\ [_O\ a\ sandwich]]$). The *HD* parameter determines the canonical linear order of a head and its complement within a constituent. In its positive setting, the parameter yields a head-initial order (Head–Complement); in its negative setting, it yields a head-final order (Complement–Head).[1] Taking the verb phrase of a transitive verb as an illustrative case, both ENG and ZHO consistently exhibit VO order, as discussed above. NLD, by contrast, displays a split pattern, also found in other Germanic languages: word order is sensitive to clause type, with matrix clauses exhibiting VO order and embedded clauses exhibiting OV order.

V2 parameter The *V2* parameter determines the presence of the so-called verb-second phenomenon, which is widespread among Germanic languages.

[1] Here, we greatly simplify the linguistic analysis for the sake of clarity (see Limitations section). According to Biberauer and Roberts (2015), HD is determined by a hierarchy of binary parametric choices: (i) Is head-final present? If not, L is head-initial; if yes: (ii) is head-final for all heads? If yes, L is head-final; if not, (iii) is it present in all +V heads? If not, (iv) a subset of +V heads it affected? and so on. In several influential works, HD is recast in terms of (roll-up) movement (Kayne, 2018), which, however, must be justified by specific features.

Under V2, the finite verb in a declarative sentence surfaces in the second position from the left edge of the clause, immediately after the first constituent (Adger, 2003). NLD exhibits this phenomenon: in *Gisteren heeft Jan haar gekust* (lit. 'Yesterday has John her kissed'), the inflected auxiliary HAVE must appear in second position (Zwart, 1997).

WH parameter As a final example, the *WH* parameter concerns long-distance dependencies. In languages such as ENG, a wh-item must be interpreted in a thematic position that may be arbitrarily far from its surface position, as in *What did you eat* $\__{what}$*?*. This follows from the assumption that thematic relations are assigned in a fixed configuration (Hale and Keyser, 1993, 2002); therefore, V and O must be merged together at some point in the derivation. Languages such as ZHO exhibit the opposite pattern: the wh-item remains in its thematic position, yielding structures such as *You eat what?*[2]

Table 1 summarizes a relevant set of classical clausal parameters (LiPs) that distinguish the three languages under analysis and indicates their observable effects on word order.

Parameter	Word-order effects	ENG	NLD	ZHO
Head Directionality (HD)	VO vs. OV	+	+/-	+
Verb-Second (V2)	V2 vs non-V2	-	+	-
Filler-Gap (WH)	null vs overt wh-mov.	+	+	-
Pro-drop (P0)	null vs overt subject	-	-	+
Tense morphology (T0)	null vs overt tense marking	+	+	-
Subject Verb Agreement (SV)	null vs overt agreement on V	+	+	-

Table 1: Clausal parameters (LiPs) across ENG, NLD, ZHO. + = positive setting; – = negative setting. The parameters selected for this study are in light-gray rows.

3 Tokenization issues

Standard tokenization approaches adopt four stances that are systematically implausible from a child-ecological perspective:

(a) *Representational opacity*. Once a span has been assigned a token ID, the model has no direct access to its internal (phonotactic/orthographic) structure.

(b) *Context independence*. A string is segmented identically regardless of its surrounding syntactic, semantic, or discourse context.

(c) *Compression objective*. The segmentation objective optimizes compression or unigram likelihood. Goldman et al. (2024) suggest that this is actually already part of the language modeling (predictive) effort. However, combined with (a), it entails that sufficiently frequent related forms, e.g., *cat* and *cats*, may be represented as unrelated atomic symbols, leaving their systematic relation to be recovered from distributional evidence during subsequent training (see Fusco et al. 2025).

(d) *Frozen learning*. The segmentation is computed once and never revised in light of subsequent structural evidence (no morphological re-analysis).

Tokenization is thus a logical bottleneck in language modeling and BPE's merge table (Sennrich et al., 2016) or the Unigram algorithm (Kudo and Richardson, 2018) are prototypical examples that align perfectly with these anti-ecological perspectives. A way to (partially) dissolve the constraints defined above is to make the boundaries fall where predictability breaks down and systematic variation appear. Harris (1955) located morpheme boundaries at points of high branching factor (low transitional probability), Elman (1990) showed recurrent prediction error spiking at word edges, while Saffran et al. (1999) demonstrated infants segmenting on transitional-probability dips.

Following on this line of work, Fusco et al. (2025) observed that split-based, morphologically robust tokenizers (like *ParadigmFinder* and *MorPiece*) performs significantly better than merge-based tokenizers (like *BPE*) under morphological evaluation (Sigmorphon 2022), though this superiority does not uniformly translate into a better models' performance, for instance on BLiMP tasks. ZHO language forced us to rephrase the split-based idea, since the original spacing hard-cue can hardly be used in Chinese.

3.1 MorPiece (MoP)

To align positions (a)-(d) to children's experience, a *boundary discovery* option is introduced into MorPiece (MoP) training (Chesi, 2026). Word spacing is demoted to a soft cue, while strong punctuation serves as the only reliable cue for segmenting initial candidates. A curriculum learning approach is adopted, whereby shorter pieces are processed first to strengthen segmentation evidence in accordance with the *Tolerance/Sufficiency Principle* (*TP*) (Yang, 2004). Child-directed speech is

[2]This is, again, a linguistic simplification, since interrogative force must nonetheless be interpreted in the relevant left-peripheral position: $_{what}$ *you eat what?* (Cheng, 2009).

also prioritized over other sources (e.g., subtitles or Wikipedia) and is processed first. The basic idea behind MoP is to traverse each input character sequence and populate two *tries* (Fredkin, 1960): one consisting of the left-to-right sequence (*root-trie*) and the other of the reversed sequence (*infl-trie*). A split is postulated whenever TP is satisfied in both the left-to-right and right-to-left directions at a specific node. For example, "wugs" is split into "wug" and "s" if TP applies both ways, between "g" and "s" (i.e., in the root-trie: $freq(node_s) > (freq(node_g)/ln(freq(node_g)))$ and between "s" and "g" (i.e., in the infl-trie: $freq(node_g) > (freq(node_s)/ln(freq(node_s)))$. Each language in the byte-premium balanced multilingual corpus (ENG, NLD, ZHO) was trained independently, and its token contribution was capped at 16K. After vocabulary combination and productivity optimization (suffix fertility > 3, minimum frequency = 10), the final multilingual vocabulary contained 39K tokens. Finally, the MoP-trained tokenizer is evaluated for order of acquisition (`--ooa` option) in English by inspecting productive vocabulary growth every 440k syllables of exposure (roughly three months of English language input). The resulting morpheme acquisition order shows a positive correlation with the developmental gold-standard ordering (Brown, 1973; De Villiers and Roeper, 2011): τ_b = .485, p = .031 (two-tailed, permutation test), N = 12. Because frequency was itself negatively related to the gold ordering (τ_b = -.454), we computed a frequency-controlled partial correlation, which remained positive, $\tau_{partial}$ = .231 (Figure 1). Note that morpheme splitting sometimes overgeneralizes, and the first item selected in the graph may simply correspond to an incorrectly segmented word (e.g., *moth-er*; although unrelated to the comparative marker, this constitutes an interesting generalization worth reporting).

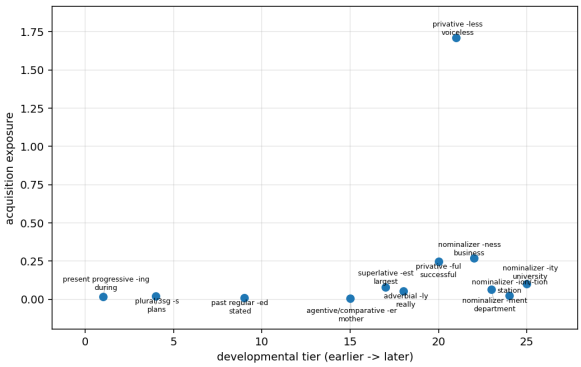


Figure 1: MoP Order of Acquisition. Training parameters: min. frequency for TP application 50; min. token freq. 10; min. suffix productivity 3; vocabulary productivity recorded every 440K syllables of exposure (∼3 months of English exposure).

3.2 ParadigmFinder (PaF)

For this multilingual setting, we also extend the unsupervised ParadigmFinder (PaF) tokenizer first described by Fusco et al. (2025). Building on the paradigm-based approaches of Goldsmith (2001) and Xu et al. (2018), PaF identifies recurring root–suffix combinations in the training vocabulary and organizes them into paradigms, each comprising a set of roots and a corresponding set of suffixes that co-occur with systematic regularity. Paradigm induction begins by enumerating the possible binary splits of each word into candidate roots and suffixes and grouping roots that occur with identical suffix sets. Candidate paradigms are subsequently normalized by factoring out prefixes shared by all their suffixes and transferring this material to the roots. This operation places segmentation boundaries at points supported by actual variation within the paradigm and prevents spurious analyses caused by root-final characters being treated as parts of suffixes. The resulting paradigms are then expanded according to TP: paradigms with smaller suffix inventories can be merged into more extensive ones when the overlap between their root sets provides sufficient evidence that the additional suffixes represent a systematic generalization rather than an accidental co-occurrence. PaF also considers nested suffixation when a word admits multiple compatible analyses. For example, if *singer* and *singers* support both the analyses *singer-∅/singer-s* and *sing-er/sing-ers*, the relation between *-er* and *-ers* can itself be analyzed through the productive ∅, `-s` paradigm. Redundant and subsumed paradigms are subsequently pruned, and the remaining paradigms are ranked by a support score that rewards both the number of participating roots and the number of associated suffixes. Words for which no productive analysis is supported are assigned to a residual paradigm, thereby avoiding unsupported segmentations. At tokenization time, words are first matched against the induced paradigms and segmented into the corresponding root and suffix units; when no complete paradigm match is available, a longest-known-suffix strategy is used to recover partial structure.

PaF was trained on the ENG, NLD, and ZHO por-

tions of the multilingual training corpus. Paradigm induction was carried out independently for each language so that morphological regularities were inferred from language-internal evidence and patterns from one language could not influence the segmentation of another. This produced separate language-specific inventories of paradigmatic roots, suffixes, and residual units, which were subsequently combined into a single multilingual vocabulary. The vocabulary contribution of each language was capped at 16K tokens as in MoP. After the three inventories were combined, training yielded a final vocabulary of approximately 44K tokens.

4 Architectural alternatives

One of our main objectives was to evaluate whether attention-free models (no KV cache) better capture structural dependencies cross-linguistically, confirming other monolingual studies (Wilcox et al., 2023). Because memory in an RNN-based architecture is entirely encoded in the recurrent mechanism, specific wiring choices may provide varying degrees of support for the linguistic variation we aim to probe.

4.1 Attention vs. Recurrent heads

Building on De et al. (2024), we adapted the Hawk model by introducing a temporal-mixing block between two residual blocks, followed by an MLP block, and compared this architecture with the GPT-2 baseline. We opted for a Real-Gated Linear Recurrent Unit (RG-LRU) (Orvieto et al., 2023), matching the Hawk architecture, without the interleaved local attention used in the more performant Griffin variant. The recurrence is the diagonal parallelized affine scan $h_t = a_t \odot h_{t-1} + b_t$ (a $log_2\, T$-depth Hillis–Steele pass replaces the T-step loop, exact to 6×10^{-8}). The recurrent block is a leaner block compared to the attention-based block, moreover the sequence mixing cost is linear with respect to the sequence length in Hawk, quadratic in GPT-2. Table 2 summarizes the relevant differences between the two architectures. According to De et al. (2024), Hawk model exceeds standard Transformers performance, and also more efficient RNN models based on Selective State Spaces such as Mamba (Gu and Dao, 2024), on downstream tasks when trained on half as many token.

Feature	GPT-2	Hawk
Temporal mixing	Multi-head causal self-attention	Real-Gated Linear Recurrent Unit (RG-LRU)
Sequence mixing cost	$O(n^2)$ in sequence length (attention matrix)	$O(n)$ — linear recurrence, fixed-size state
State at inference	Grows with context (KV-cache)	Fixed-size hidden state, independent of context length
Positional info	Learned absolute position embeddings	None needed — recurrence is inherently ordered
Long-range dependency	Direct token-to-token attention weights	Gated accumulation/decay through the recur. state
Training parallelism	Native (attention is parallel)	Associative-scan to parallelize recurrence
Params for depth/width	Attention QKVO projections + MLP	Recurrent block (RG-LRU gates + projections) + MLP

Table 2: Comparison Between GPT-2 and Hawk Architectures.

5 Materials and methods

5.1 Tokenizers evaluation

Tokenizers performance is assessed first based on morphological independent Sigmorphon 2022-like tests. Sigmorphon 2022 benchmark (Batsuren et al., 2022) has been directly adapted for ENG, while NLD and ZHO morphological benchmarks are automatically generated from on-line resources freely available: NLD is based on UniMorph dataset (inflections only); ZHO is based on the morphological annotation in public Childes datasets. We also evaluated the overall performance of each tokenizer directly on morphologically relevant minimal pairs decision tasks. For the comparison we trained each tokenizer on the byte-premium paired clean multilingual corpus of 100M tokens.[3] We focused only on local and non-local agreement phenomena in minimal pairs tests[4] using the same GPT-2 model (BabyLM Multilingual baseline). We compared BPE, UniGram and MoP tokenizers initially with the same 16K (max) per language vocabulary size.

5.2 Linguistic Parameters (LiP) setup

Two parameterization injection strategies are considered:

[3] All text was lowercased and any metalinguistic information (e.g., speaker tiers) was removed.

[4] BLiMP-filtered: anaphor_gender_agreement, anaphor_number_agreement, determiner_noun_agreement_*, distractor_agreement_*, irregular_plural_subject_verb_agreement_*, regular_plural_subject_verb_agreement_*; BLiMP-NL: anaphor_agreement; BLiMPZHO: anaphor_gender_agreement, anaphor_number_agreement, classifier_noun_agreement_*)

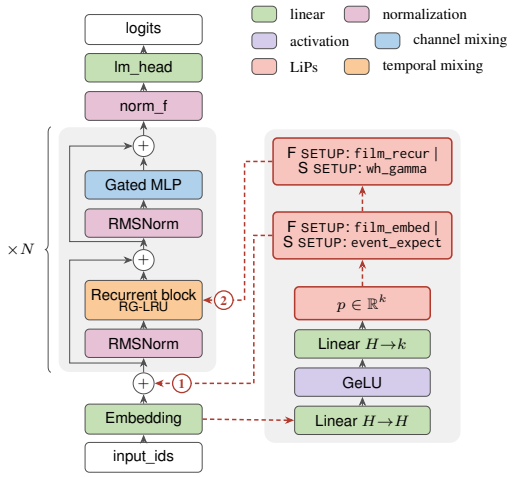


Figure 2: Hawk with linguistic-parameter (LiP) control. The compute path is shown on the left; the two alternative parameterization modes and their injection points are shown on the right. See Figure 3 in Appendix A.2 for more details.

F setup a *Feature-wise Linear Modulation* (*FiLM*) (Perez et al., 2017) is implemented as a router that must infer k soft bits from the input and conditions the backbone diffusely, supervised toward LiPs gold via Binary Cross-Entropy (BCE).

S setup a purely *switch*-based parameterization, where three named single-site switches on LiPs are coded and supervised.

For the current experiments we will concentrate on the three LiPs discussed in Section 2: *HD*, *V2*, and *WH*. The choice is linguistically justified because their scope is different: *HD* hinges directly on the local merge mechanism, *V2* requires the accommodation of a clause-bounded tense re-ordering, while the *WH* parameter might require a cyclic long-distance dependency spanning various clauses. The injection strategy is illustrated in Figure 2 (See Appendix A.2 for technical details).

5.3 Hawk model variants

To assess the locus of potentially linguistically relevant generalizations, we considered minimal variations of a baseline Hawk model, with and without LiPs injection. Relative to the GPT-2 baseline models, the capacity modifications were designed to test the following hypotheses:

Hawk-A Increasing the model depth (number of layers) and the recurrent state size (width of the recurrent units) at the same parameter budget as the baseline might improve both hierarchical sensitivity (through greater depth) and information retention (through wider recurrent units).

Setup	GPT-2	base	A	B	C
Parameters (M)	12	115	112	142	115
Layers	12	12	18	12	12
Embedding	768	704	704	704	704
Recurrent width	–	768	864	1152	768
Kernels	–	4	4	4	8
MLPs	–	3	2	3	3

Table 3: Comparison Between GPT-2 and Hawk different setups.

Hawk-B A facilitation in non-local LiPs channeling (e.g., longer retention of WH dependencies) might be obtained by increasing the recurrent state size (+50%: from 768 to 1152).

Hawk-C A facilitation in local LiPs channeling (e.g., HD and local agreement phenomena such as SV agreement in the absence of local interveners) could be obtained by doubling the convolutional kernel count (from 4 to 8).

All models are trained as a single multilingual causal LM over ENG, NLD, and ZHO on the BabyLM 2026 budget (∼100M byte-premium-adjusted English-equivalent words), 10 epochs, sequence length 512, using a byte-fallback MoP tokenizer (vocab 39,953, see § A.1 for discussion on the impact of byte-fallback). Optimization: AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 0), peak LR $5e-4$ with 1% linear warmup and cosine decay to $5e-5$, grad-clip 1.0, $bf16$ mixed precision, RMSNorm (instead on LayerNorm) and tied input/output embeddings. Regarding "grammatical awareness", it is worth emphasizing that, in the baseline Hawk setup, only 28.1M parameters are allocated to the embedding layer, while the remaining 87.3M are non-embedding parameters. Originally, we adopted a simple sequential recurrence and subsequently implemented Hawk's native parallel (diagonal) scan. This dramatically reduced the training time (about 5% of our naive model's implementation).

6 Results

6.1 Tokenizers evaluation

The SIGMORPHON-based morphological evaluation of each tokenizer is reported in Table 4, with the monolingual baseline results from Fusco et al. (2025) shown in gray. All multilingual tokenizers exhibit a substantial decline relative to their

Tokenizer	Avg. Lev. Dist.	Prec	Rec	F1
BPE	2.08	21.03	26.62	23.50
MoP	1.96	24.54	29.52	26.80
PaF	**1.24**	**38.99**	**30.12**	**33.99**
BPE-ML$_{ENG}$	7.25	**13.52**	**23.15**	**17.07**
MoP-ML$_{ENG}$	**6.40**	12.84	20.89	15.91
PaF-ML$_{ENG}$	12.98	3.73	1.72	2.35
BPE-ML$_{NDL}$	**5.63**	**16.44**	**27.11**	**20.46**
MoP-ML$_{NDL}$	5.97	13.04	22.78	16.59
PaF-ML$_{NDL}$	10.79	14.98	9.69	11.77
BPE-ML$_{ZHO}$	8.25	0.00	0.00	0.00
MoP-ML$_{ZHO}$	**1.18**	**77.04**	**66.96**	**71.65**
PaF-ML$_{ZHO}$	2.36	50.15	32.66	39.55

Table 4: BPE, ParadigmFinder (PaF), and MorPiece (MoP) multilingual (-ML) evaluation (ENG, NLD, and ZHO, 16K per language vocab) on SIGMORPHON-inspired tasks. Gray rows show monolingual tokenizers results (BabyLM 10M).

Tokenizer	ENG	NLD	ZHO
BPE-ML	0.776863	**0.845**	0.437500
UniG-ML	0.515994	0.460	0.420825
MoP-ML	0.801362	0.840	0.467525
PaF-ML	**0.808980**	0.835	**0.475000**

Table 5: BPE, UniGram (UniG), MorPiece (MoP) and ParadigmFinder (PaF) multilingual (-ML) evaluation on local and non-local agreement on BLiMP-tasks.

monolingual counterparts. This degradation is especially pronounced for PaF, whose reliance on matching words against induced, language-specific paradigms makes it particularly sensitive to reduced vocabulary coverage in the multilingual setting. BPE remains comparatively strong for ENG and NLD, whereas MoP achieves more consistent performance across all three languages. In Table 5, we report the performance of the same baseline GPT-2 model trained with different tokenizers.

In the extrinsic evaluation, MoP and PaF outperform the other tokenizer, surpassing BPE by 3 percentage points in both English and Chinese. In Dutch, they trail BPE by at most 1 percentage point. Taken together, the two evaluation tests seem to suggest that although the two multilingual morphological tokenizers might struggle to correctly identify morpheme boundaries in morphologically complex unseen words, they nonetheless give a non-negligible advantage in downstream tasks involving morphological agreement. Nevertheless, in the remainder of this work, we report experiments only with MoP, as it exhibits more consistent performance across the two tokenizer evaluations. Increasing vocabulary size produces a mild improvement in BPE and MoP tokenizers, but not for Unigram.

6.2 Architectural choices impact

One clear result of our experiments is that Hawk is sample-efficient while GPT-2 is not. At the 100M-word budget, Hawk-B beats GPT-2 MoP by 6.5 points (.6110 vs .5463) and even Hawk-A beats it by 5. The Transformer's eventual catch-up happens at a token count the challenge doesn't reward. Recurrence wins at the developmentally-plausible data scale, and the Transformer (GPT-2 baseline model) only equalizes with ~10× the budget the task allows (Table 7 in the Appendix).

On English, Hawk with MoP scores highest on (72.4%), followed by Hawk with MoP+bf (70.4%), FiLM (70.3%), SWITCH (70.0%), Hawk B (69.5%), Hawk C (69.4%), BabyLM GPT-2 baseline (69.2%), and Hawk A (69.0%). Globally, none of the capacity variation on Hawk architecture produced significant changes in performance, while parametric injection (under the SWITCH supervision) only mildly improves both with respect to the un-injected Hawk baseline (+0.005) and with respect to the GPT-2 baseline (+0.017) on BLiMPs averaged (Table 6 in the Appendix).

Focusing on more fine-grained linguistic phenomena, FiLM produces the single largest category-level gain seen anywhere in this study: Island Effects[5] rise from 56.2% (baseline) to 61.9% (+5.8 pts), consistent with the kind of hierarchical/structural sensitivity that hypotheses (Hawk A) are ultimately after. FiLM also gains slightly on Control/Raising and Binding. Against this, FiLM loses on Ellipsis (-7.0 pts), Subject–Verb Agreement (-2.6 pts) and Det-Noun Agreement (-2.2 pts), and on the fine-grained clusters it is negative on every agreement and WH-related cluster (see Tables 8-10 for fine-grained results on BLiMP tasks across the three languages).

SWITCH is closer to a flat rescaling of the baseline: small losses on most categories (NPI Licensing -3.2 pts, Quantifiers -2.4 pts, Anaphor Agreement -0.6 pts), essentially no change on Island Effects, but the largest gain of all seven models on the external MultiBLiMP-EN agreement benchmark (89.5% vs.

[5]In BLiMP: adjunct_island, complex_NP_island, coordinate_structure_constraint_*, left_branch_island_*, sentential_subject_island, wh_island; wh_island, in fact, should be considered a "weak" islands (Szabolcsi and Zwarts, 1993).

87.0% baseline), which is not fully consistent with the modest losses SWITCH shows on the internal SV/Det–Noun agreement clusters. This discrepancy is a useful reminder that "local agreement" is operationalized somewhat differently across benchmarks, and a single mechanism's ranking can depend on which agreement probe is used.

On the capacity variants, none of A, B, or C setups improves the overall BLiMP aggregate relative to the (also LiP-free) Hawk MoP+bf baseline: A is 2.1 points lower, B 1.6 points lower, C 1.9 points lower. Whatever local benefits the capacity changes bring, they come with a broad-based cost elsewhere in the grammar suite. Nevertheless, all three capacity variants show a small increase in Island Effects sensitivity over the baseline (A: +0.6, B: +2.7, C: +2.6 pts), mirroring the FiLM result.

7 Discussion

Consistent with previous findings, (i) the tokenization experiments show that, despite their morphological reliability (on agreement forced-choice tasks), the impact of linguistically motivated tokenizers (MoP and PaF) is not uniform across downstream tasks, including purely morphosyntactic contrasts such as those assessed by the BLiMP benchmark (Fusco et al., 2025) (ii) complex wh-dependencies remain difficult to capture (Lan et al., 2024), even with the apparently more syntactically sensitive and budget-efficient recurrent architectures (Wilcox et al., 2023). The significant improvement on island phenomena under the FiLM approach suggests that the linguistic intuition underlying the LiP may play an important role in this domain. However, the deterioration in performance across other structural domains warrants caution. Repeating the experiments across multiple runs would provide more precise uncertainty estimates and help establish the robustness of these findings. Overall, neither LiPs injection nor the capacity modifications yielded the expected performance gains on the targeted phenomena. Despite isolated, linguistically unpredictable significant improvements, the overall performance remained in line with the baseline of un-injected models. Back to the tokenization strategies, it is worth highlighting that estimating the Order of Acquisition (OoA) of morphologically productive splits is a by-product of the incremental approach adopted by the MoP tokenizer. The positive correlation with the English gold standard is noteworthy and complements the Age of Acquisition (AoA) evaluations included in the BabyLM benchmark.

8 Conclusion

In this paper, we compared the attention-based GPT-2 baseline performance with a state-of-the-art recurrent model with no attention at all. The Hawk model we tested trains much more efficiently than GPT-2 model and reaches top-performance after 300/400M word budget (3 or 4 training epochs). The choice of a RNN was motivated by cognitive plausibily: only models that simulate incremental processing can be considered fully coherent with the way humans generate and parse sentences. We also believed that forcing the model to make early decision on what to retain in memory (recurrent block) and what to forget, might induce the kind of relevant bias affect linguistic processing (Chesi et al., 2026). This conclusion is only partially supported by the mild improvement Hawk models record over the GPT-2 baseline. Another central hypothesis concerned whether linguistically motivated parametric choices could help the model distinguish ENG from NLD and ZHO (Roberts, 2019). To investigate this possibility, we isolated three relevant parameters: Head Directionality (HD), which regulates local predicate–argument ordering (ENG vs. NLD); Verb Second (V2), which requires a tense-bearing element to move to the second position in the clause (NLD vs. ENG/ZHO); and Filler–Gap Dependency (WH), which requires a wh-item to move from its thematic position (ENG/NLD vs. ZHO). We incorporated these parametric switches into the RNN model and allowed it to use them to regulate both the embedding and the recurrent block. We implemented two alternative injection strategies: FiLM and a fully supervised, gold-standard SWITCH. Despite modest improvements in some targeted non-local phenomena under the FiLM approach, none of our attempts produced the expected significant overall improvement relative to the baseline. Given the current (limited and focused) linguistic parametric implementation, these results might suggest that LiPs serve as a scientific probe rather than a capacity crutch. Overall, we observe that the loss landscape is actively hostile to phrase structure and not just merely indifferent: whenever model capacity is given to a specific dimension with a specific structural hypotheses, performance oscillation often bend in unexpected linguistic directions.

9 Limitations

Due to space limits, in this study, we report results only for a specific RNN architecture (Hawk models). We conducted many additional experiments on a range of RNN architectures, all motivated by assumptions about linguistic processing, including eMG-RNNs and Selective State Space models with different channel capacities designed to bias syntactic processing according to locality constraints (local merge, raising and control, long-distance dependencies, phase-based movement, and binding). All trained models are available at NeTSlab/babylm-2026. Source code for training tokenizers and models is availabe at cristianochesi/babylm-2026.

The LiP supervision scheme is highly simplified. Considering the three parameters (HD, V2, and WH) that distinguish English (ENG), Dutch (NLD), and Chinese (ZHO), we represent them using three-dimensional vectors encoding their values across the three languages: HD = [1, 0.5, 1], V2 = [0, 1, 0], and WH = [1, 1, 0]. The injection strategy is consistent with the hypothesis that one parameter (e.g., the HD LiP) should operate directly on the embeddings, whereas another (e.g., the WH LiP) should instead be applied to the recurrent block. This is by no means the only plausible hypothesis, and further experiments are needed to explore alternative injection strategies.

Another fundamental assumption concerning linguistic parameters, which is not addressed in this paper, is that the setting of one parameter constrains the settings of others (Roberts, 2019). This implicational structure is central to the *Poverty of the Stimulus argument* and is consistent with cross-linguistic observations concerning language acquisition: children acquire language rapidly, follow similar developmental paths, and exhibit similar patterns of error (Yang, 2004; Guasti, 2017). Building on this assumption, Biberauer and Roberts (2015) propose a hierarchical model comprising four parameter levels (macro-, meso-, micro-, and nano-parameters), ordered according to the size of the grammatical class targeted by each parameter. Macro-parameters are therefore more general, applying consistently to all heads of the relevant type, and predict the simultaneous clustering of several properties, whereas micro- and nano-parameters capture increasingly idiosyncratic and lexically restricted variation. We assume that a positive setting of a higher-order parameter implies, or renders irrelevant, a cluster of lower-order options, whereas a negative setting blocks them (Pescarini, 2025). In this study, we focus only on three prototypical LiPs, necessarily simplifying the more nuanced linguistic analyses that recast these classical parameters in Minimalist, feature-based terms. No implicational dependencies are considered at this stage.

Regarding the OoA analysis, the gold standard considered here is available only for English. We have comparable gold standards based on Belletti and Guasti (2015) and Caprin and Guasti (2009) available at cristianochesi/babylm-2026; similar resources would need to be developed for ZHO and NLD.

Acknowledgments

This work contains simulations carried out on the High Performance Computing DataCenter at IUSS Pavia, co-funded by Regione Lombardia through the funding programme established by Regional Decree No. 3776 of November 3, 2020.

Author Contributions

CC: conceptualization; supervision; methodology; software (training pipeline and evaluation harness); formal analysis; funding acquisition; resources (compute infrastructure); writing — original draft. **AF:** conceptualization; methodology; software (training pipeline and evaluation harness); formal analysis; writing — original draft; **AM:** data curation; software; writing — original draft **GF, SR:** conceptualization; methodology; formal analysis; writing — original draft. **HBC, TS, AZ:** data curation; methodology; formal analysis; writing — review and editing. All authors read and approved the final manuscript and take responsibility for its content.

Use of AI Assistants

We disclose the use of generative AI assistants in the preparation of this work, in line with the ACL Policy on Publication Ethics.

Writing. ChatGPT (SOL, version 5.6, accessed JUNE-SEPTEMBER 2026) was used exclusively for language editing of text written by the authors: grammar and spelling correction, rephrasing for concision, and LaTeX formatting. No section of the paper was generated from a prompt describing content, and no claim, argument, related-work summary or citation was produced by the model.

All edited passages were checked by the authors against the original text.

Code. An AI coding assistant (CLAUDE OPUS, version 4 and 5, accessed March-July 2026) was used for bug fixing and significant optimization of the original code as well as auxiliary scripting tasks: boilerplate for data loading, refactoring of the evaluation scripts and visualization scripts. The core modeling code was adapted by the authors from https://github.com/knotgrass/Griffin. All AI-assisted code was reviewed, executed and tested by the authors, and is released in the accompanying repository, where the affected files are marked in the `README`.

Research content. No AI assistant was used to generate research ideas, to design experiments, to analyse or interpret results, to retrieve or select references, or to produce any data reported in the paper. The authors take full responsibility for the content of the manuscript, including any text that was language-edited with AI assistance.

References

David Adger. 2003. *Core syntax: a minimalist approach*. Core linguistics. Oxford University Press, Oxford ; New York.

Khuyagbaatar Batsuren, Gábor Bella, Aryaman Arora, Viktor Martinovic, Kyle Gorman, Zdeněk Žabokrtský, Amarsanaa Ganbold, Šárka Dohnalová, Magda Ševčíková, Kateřina Pelegrinová, Fausto Giunchiglia, Ryan Cotterell, and Ekaterina Vylomova. 2022. The SIGMORPHON 2022 shared task on morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 103–116, Seattle, Washington. Association for Computational Linguistics.

Adriana Belletti and Maria Teresa Guasti. 2015. *The Acquisition of Italian: Morphosyntax and its interfaces in different modes of acquisition*, volume 57. John Benjamins Publishing Company.

Theresa Biberauer and Ian Roberts. 2015. Rethinking formal hierarchies: A proposed unification. *Cambridge Occasional Paper in Linguistics*, 7:1–31.

Roger Brown. 1973. *A first language: The early stages.* Harvard U. Press.

Claudia Caprin and Maria Teresa Guasti. 2009. The acquisition of morphosyntax in Italian: A cross-sectional study. *Applied psycholinguistics*, 30(1):23.

Lisa Lai-Shen Cheng. 2009. Wh-in-situ, from the 1980s to now. *Language and Linguistics Compass*, 3(3):767–791.

Cristiano Chesi. 2023. Parameters of cross-linguistic variation in expectation-based Minimalist Grammars (e-MGs). *Italian Journal of Computational Linguistics*, 9(1).

Cristiano Chesi. 2026. MorPiece: A split-based tokenization library. `https://github.com/cristianochesi/morpiece`. Version 1.4.5.

Cristiano Chesi, Matilde Barbini, Veronica Bressan, Achille Fusco, Sofia Neri, Maria Letizia Piccini Bianchessi, Sarah Rossi, and Tommaso Sgrizzi. 2026. From recursion to incrementality: return to recurrent neural networks. *Linguistics Vanguard*.

Noam Chomsky. 1981. *Lectures on government and binding: The Pisa lectures*. 9. Walter de Gruyter.

Noam Chomsky. 1986. *Knowledge of language: its nature, origin, and use*. Convergence. Praeger, New York.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. 2024. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *Preprint*, arXiv:2402.19427.

Jill De Villiers and Tom Roeper, editors. 2011. *Handbook of Generative Approaches to Language Acquisition*, volume 41 of *Studies in Theoretical Psycholinguistics*. Springer Netherlands, Dordrecht.

Jeffrey L. Elman. 1990. Finding Structure in Time. *Cognitive Science*, 14(2):179–211.

Edward Fredkin. 1960. Trie memory. *Communications of the ACM*, 3(9):490–499.

Achille Fusco, Maria Letizia Piccini Bianchessi, Tommaso Sgrizzi, Asya Zanollo, and Cristiano Chesi. 2025. Linguistic Units as Tokens: Intrinsic and Extrinsic Evaluation with BabyLM. In *Proceedings of the First BabyLM Workshop*, pages 496–507, Suzhou, China. Association for Computational Linguistics.

Omer Goldman, Avi Caciularu, Matan Eyal, Kris Cao, Idan Szpektor, and Reut Tsarfaty. 2024. Unpacking tokenization: Evaluating text compression and its correlation with model performance. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2274–2286, Bangkok, Thailand. Association for Computational Linguistics.

John Goldsmith. 2001. Unsupervised learning of the

morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

Albert Gu and Tri Dao. 2024. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint*. ArXiv:2312.00752 [cs].

Maria Teresa Guasti. 2017. *Language acquisition: The growth of grammar*. MIT press, Cambridge, MA.

Kenneth L. Hale and Samuel Jay Keyser. 1993. On argument structure and the lexical expression of syntactic relations. *An annotated syntax reader*, pages 312–327.

Kenneth L. Hale and Samuel Jay Keyser. 2002. *Prolegomenon to a theory of argument structure*. Number 39 in Linguistic inquiry Monographs. MIT Press, Cambridge, Mass.

Zellig S Harris. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Richard S. Kayne. 2018. The place of linear order in the language faculty. *Ms. New York University*.

Jordan Kodner, Sarah Payne, and Jeffrey Heinz. 2023. Why Linguistics Will Thrive in the 21st Century: A Reply to Piantadosi (2023). *arXiv preprint*. ArXiv:2308.03228 [cs].

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Nur Lan, Emmanuel Chemla, and Roni Katzir. 2024. Large Language Models and the Argument from the Poverty of the Stimulus. *Linguistic Inquiry*, pages 1–28.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2023. Resurrecting recurrent neural networks for long sequences. *Preprint*, arXiv:2303.06349.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. 2017. Film: Visual reasoning with a general conditioning layer. *Preprint*, arXiv:1709.07871.

Diego Pescarini. 2025. Quantitative modelling and syntactic microvariation. *Atti del Sodalizio Glottologico Milanese*, 2023(18 n.s.):91–108.

Ian Roberts. 2019. *Parameter Hierarchies and Universal Grammar*. Oxford University Press.

Jenny R Saffran, Elizabeth K Johnson, Richard N Aslin, and Elissa L Newport. 1999. Statistical learning of tone sequences by human infants and adults. *Cognition*, 70(1):27–52.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. *arXiv preprint*. ArXiv:1508.07909 [cs].

Anna Szabolcsi and Frans Zwarts. 1993. Weak islands and an algebraic semantics for scope taking. *Natural Language Semantics*, 1(3).

Ethan Wilcox, Richard Futrell, and Roger Levy. 2023. Using Computational Models to Test Syntactic Learnability. *Linguistic Inquiry*, pages 1–44.

Hongzhi Xu, Mitchell Marcus, Charles Yang, and Lyle Ungar. 2018. Unsupervised morphology learning with statistical paradigms. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 44–54, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Charles D. Yang. 2004. Universal Grammar, statistics or both? *Trends in Cognitive Sciences*, 8(10):451–456.

Charles D. Yang, Stephen Crain, Robert C. Berwick, Noam Chomsky, and Johan J. Bolhuis. 2017. The growth of language: Universal Grammar, experience, and principles of computation. *Neuroscience & Biobehavioral Reviews*, 81:103–119.

C. Jan-Wouter Zwart. 1997. *Morphosyntax of Verb Movement*, volume 39 of *Studies in Natural Language and Linguistic Theory*. Springer Netherlands, Dordrecht.

A Appendix

A.1 Byte_fallback reduces model's sensitivity to non-local dependencies

Two baseline Hawk models were trained one without (default), the other with (+fb) MoP byte_fallback option set. Since the performance does not match significantly on all phenomena, also considering the evaluation standard errors of each phenomenon. The Hawk+bf is not uniformly better than its Hawk predecessor: it gains on Anaphor Agreement (+1.3 pts) and Control/Raising (+0.8 pts), is essentially flat on Determiner–Noun Agreement, Ellipsis, and Subject-Verb Agreement, but loses on Irregular Forms (-3.2 pts), Quantifiers (-4.3 pts), Island Effects (-4.6 pts), NPI Licensing (-5.4 pts), and Binding (-2.2 pts). These are exactly the categories that involve tracking a licensing or structural condition across some intervening material (island boundaries, NPI licensors, quantifier scope, binding domains), so at face value the byte_fallback mechanism, on its own, does not increase, and by these metrics somewhat decreases, sensitivity to long-range structural dependencies.

A.2 Technical details of the linguistic-parameter (LiP) control

A more detailed computational graph is illustrated in Figure 3. A predictor (`Linear`$(H{\to}H)$ $\to$ GeLU $\to$ `Linear`$(H{\to}k)$ over the mean-pooled embeddings) emits a parameter vector $p \in \mathbb{R}^k$, with $H = d_{\text{model}}$ and k the number of inferred parameters; it is regressed onto `default_logits`, and under SWITCH the gold $p =$ `param_targets[lang_ids]` is teacher-forced and the predictor is detached, i.e. it acts as an inference-time predictor only. The same p is injected at two sites. ① *Embedding site*, applied once per sequence right after `wte`: FILM sets $x \mathrel{+}= W_e\, p$ (`film_embed`, `Linear`$(k{\to}H)$, no bias), SWITCH adds the boundary embeddings $m_{\text{HD}} e_{\text{HD}} + m_{\text{V2}} e_{\text{V2}}$ (`event_expect`). ② *Recurrent site*, broadcast to all N layers: FILM adds $W_r\, p$ to the recurrence-gate bias (`film_recur`, `Linear`$(k{\to}d_{\text{rnn}})$, no bias), SWITCH rescales the decay, $\log a_t \leftarrow \gamma_{\text{WH}} \odot \log a_t$ (`wh_gamma`$\in \mathbb{R}^{d_{\text{rnn}}}$).

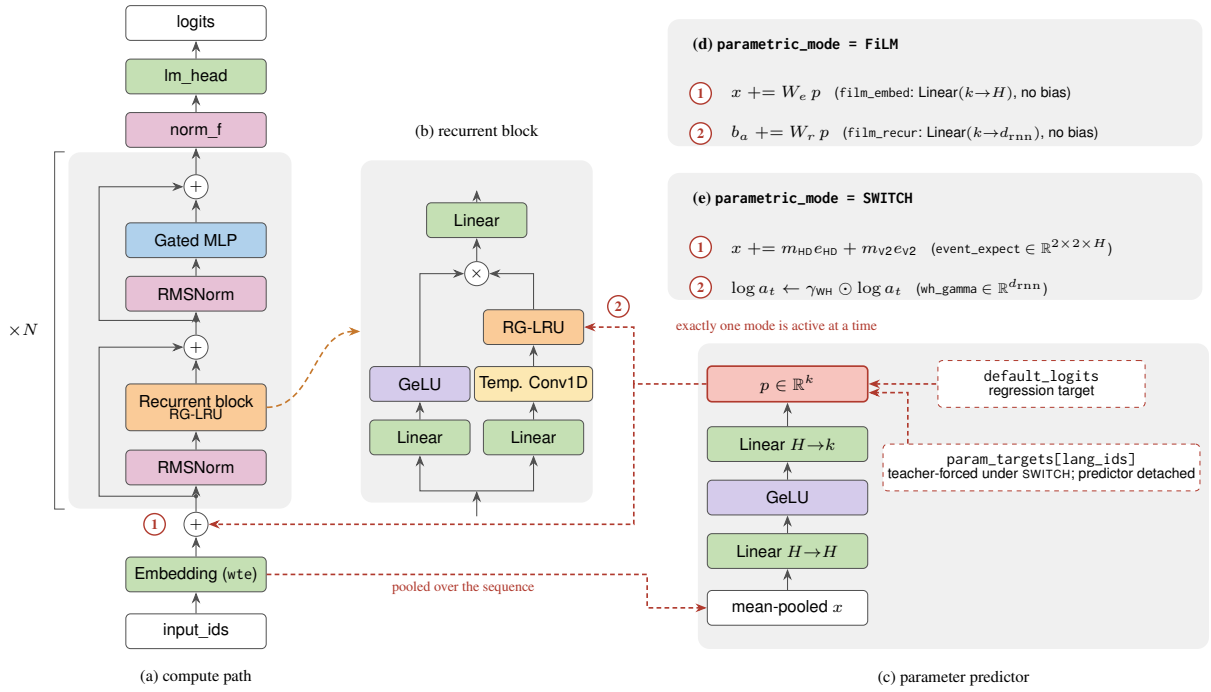


Figure 3: Expanded view of Hawk with linguistic-parameter (LiP) control. (a) The full compute path, with the embedding site ① right after `wte` and the recurrent site ② inside every one of the N layers. (b) The recurrent block opened up: ② acts on the RG-LRU, either on its recurrence-gate bias or on its decay. (c) The parameter predictor, together with the regression target used to train it and the gold parameters that replace its output under SWITCH, where the predictor is detached and therefore used at inference time only. (d)–(e) The update applied at each site under the two parameterization modes; $H = d_{\text{model}}$, k is the number of inferred parameters, d_{rnn} the RG-LRU state width and b_a the recurrence-gate bias. Colour coding is as in Figure 2.

Task	Lang.	GPT-2 baseline	GPT-2 PaF	GPT-2 MoP	HAWK MoP-base	HAWK MoP-A	HAWK MoP-B	HAWK MoP-C	HAWK MoP-FiLM	HAWK MoP-switch
blimp_babylm_filtered	eng	0.692	0.682	0.688	**0.704**	0.690	0.695	0.694	0.703	0.700
hellaswag_en_mubench	eng	**0.267**	0.260	0.262	0.260	0.264	0.263	0.262	0.264	0.264
multiblimp_eng	eng	0.871	0.884	0.855	0.870	0.857	0.871	0.870	0.866	**0.895**
winogrande_en_mubench	eng	0.488	0.509	0.508	0.510	0.501	0.482	0.513	**0.519**	0.506
xstorycloze_en_mubench	eng	0.499	0.496	0.490	0.484	**0.511**	0.501	0.494	0.491	0.496
blimp_nl	nld	0.793	0.781	0.777	0.795	0.773	0.782	0.763	0.790	**0.803**
hellaswag_nl_mubench	nld	0.264	0.264	0.264	0.261	0.262	**0.267**	0.262	0.263	0.259
multiblimp_nld	nld	**0.919**	0.883	0.897	0.915	0.892	0.885	0.890	0.892	0.911
winogrande_nl_mubench	nld	0.493	0.489	0.500	0.497	**0.506**	0.495	0.479	0.495	**0.506**
xcomps_nl	nld	0.527	0.543	0.548	0.561	0.556	0.557	0.562	**0.565**	0.563
xstorycloze_nl_mubench	nld	0.474	0.478	0.487	0.491	0.498	0.490	0.498	**0.502**	0.485
hellaswag_zh_mubench	zho	0.271	**0.273**	0.271	0.268	0.270	0.268	0.270	0.269	0.270
winogrande_zh_mubench	zho	0.497	**0.505**	**0.505**	0.495	0.480	0.487	0.485	0.494	0.479
xcomps_zh	zho	0.539	**0.569**	0.554	0.568	0.551	0.550	0.557	0.557	0.555
xstorycloze_zh_mubench	zho	0.506	0.502	0.503	0.509	0.502	0.514	**0.519**	0.517	0.503
zhoblimp	zho	0.754	0.787	0.796	0.803	0.769	0.772	0.774	0.768	**0.805**
global_piqa_nonparallel_en	eng	–	0.480	–	0.510	0.540	0.480	0.540	**0.570**	0.520
global_piqa_parallel_en	eng	–	0.233	–	0.262	0.282	0.282	**0.311**	0.252	0.204
global_piqa_nonparallel_nl	nld	–	0.490	–	0.520	**0.570**	0.540	0.500	0.550	0.530
global_piqa_parallel_nl	nld	–	0.262	–	0.233	0.272	**0.301**	0.252	0.233	**0.301**
global_piqa_nonparallel_zh	zho	–	0.460	–	0.430	0.480	0.500	**0.520**	0.450	0.440
global_piqa_parallel_zh	zho	–	0.194	–	0.165	0.204	**0.243**	**0.243**	0.184	0.184
arc	eng	**0.308**	0.258	0.237	0.240	0.240	0.265	0.252	0.262	0.252
arc	nld	0.265	0.246	0.260	**0.277**	0.273	0.237	0.275	0.231	0.254
arc	zho	0.279	0.269	**0.290**	0.262	0.265	0.244	0.258	0.265	0.240
belebele	eng	0.205	**0.227**	0.199	**0.227**	0.222	0.193	0.188	0.216	0.216
belebele	nld	0.205	0.222	**0.244**	0.222	0.193	0.233	0.205	0.216	0.233
belebele	zho	**0.256**	0.193	0.176	0.210	0.244	0.210	0.216	0.176	0.244
bmlama	eng	**0.361**	0.229	0.227	0.144	0.145	0.145	0.137	0.138	0.139
bmlama	nld	**0.334**	0.224	0.249	0.152	0.148	0.143	0.142	0.151	0.137
bmlama	zho	**0.342**	0.193	0.204	0.140	0.136	0.142	0.138	0.132	0.127
mnli	eng	**0.666**	0.560	0.552	0.430	0.385	0.387	0.392	0.414	0.436
mnli	nld	**0.612**	0.556	0.570	0.433	0.380	0.392	0.415	0.392	0.426
mnli	zho	**0.671**	0.588	0.532	0.436	0.391	0.396	0.399	0.380	0.449
sib200	eng	0.604	0.795	**0.805**	0.680	0.670	0.595	0.610	0.615	0.765
sib200	nld	0.662	**0.790**	0.785	0.700	0.615	0.530	0.600	0.625	0.730
sib200	zho	0.630	0.800	**0.805**	0.745	0.600	0.530	0.615	0.670	0.660
truthfulqa	eng	0.304	0.277	0.223	0.223	0.268	**0.313**	0.134	0.205	0.214
truthfulqa	nld	**0.313**	0.304	0.214	0.223	0.196	0.241	0.250	0.214	0.232
truthfulqa	zho	**0.295**	0.170	0.179	0.223	0.223	0.277	0.241	0.223	0.250
xnli	eng	**0.645**	0.544	0.545	0.444	0.447	0.438	0.447	0.437	0.458
xnli	zho	**0.599**	0.515	0.515	0.444	0.422	0.450	0.424	0.411	0.449
include	nld	**0.348**	0.223	0.312	0.339	0.384	0.232	0.304	0.277	0.277
include	zho	0.259	0.232	0.250	0.223	**0.304**	0.223	0.232	0.286	0.286
pos	eng	**0.941**	0.898	–	0.922	–	0.920	0.922	0.926	0.923
pos	nld	**0.956**	0.892	–	0.939	–	0.931	0.933	0.934	0.936
pos	zho	**0.917**	0.883	–	0.905	–	0.892	0.892	0.899	0.904

Table 6: Model scores on the BabyLM Multilingual Benchmark by task.

Budget (words)	GPT-2 MoP	Hawk-base	Hawk-A	Hawk-B	Hawk-C	Hawk-FiLM	Hawk-switch
Macro average (mean of the three per-language zero-shot aggregates)							
1M	0.4716	**0.4851**	0.4807	0.4696	0.4726	0.4743	0.4761
2M	0.4687	0.4788	0.4808	**0.4826**	0.4785	0.4819	0.4761
3M	0.4683	0.4840	0.4807	0.4826	0.4796	0.4834	**0.4845**
4M	0.4703	**0.4893**	0.4870	0.4851	0.4854	0.4867	0.4871
5M	0.4767	**0.5026**	0.4941	0.4914	0.4911	0.4952	0.4952
6M	0.4719	0.5036	**0.5064**	0.5007	0.5042	0.5035	0.5041
7M	0.4727	0.5178	0.5167	0.5187	0.5136	**0.5224**	0.5118
8M	0.4774	0.5241	0.5185	**0.5247**	0.5229	0.5204	0.5193
9M	0.4760	0.5271	0.5217	0.5319	**0.5342**	0.5266	0.5316
10M	0.4777	**0.5397**	0.5337	0.5362	0.5369	0.5327	0.5334
20M	0.4843	0.5614	0.5622	**0.5647**	0.5564	0.5627	0.5566
30M	0.4946	0.5735	0.5697	**0.5750**	0.5693	0.5731	0.5677
40M	0.4967	**0.5869**	0.5813	0.5863	0.5798	0.5808	0.5795
50M	0.5135	0.5849	0.5853	0.5864	0.5854	**0.5867**	0.5794
60M	0.5230	0.5927	0.5878	**0.5972**	0.5907	0.5881	0.5864
70M	0.5285	0.5955	0.5919	**0.6023**	0.5889	0.5982	0.5955
80M	0.5372	0.5941	0.5945	**0.6058**	0.5916	0.5972	0.5936
90M	0.5404	0.5975	0.5993	**0.6058**	0.5950	0.5990	0.5976
100M	0.5463	0.5990	0.5960	**0.6110**	0.6002	0.5946	0.6000
200M	0.5758	0.6115	0.6070	**0.6163**	0.6120	0.6147	0.6136
300M	0.5913	0.6157	0.6129	**0.6193**	0.6161	0.6164	0.6177
400M	0.6003	0.6177	0.6140	**0.6212**	0.6162	0.6193	0.6166
500M	0.6008	0.6160	0.6106	0.6206	0.6169	**0.6223**	0.6153
600M	0.6056	0.6157	0.6139	0.6194	0.6147	**0.6196**	0.6149
700M	0.6061	0.6116	0.6118	0.6169	0.6154	**0.6189**	0.6139
800M	0.6074	0.6078	0.6089	0.6142	0.6116	**0.6146**	0.6119
900M	0.6081	0.6078	0.6058	0.6099	0.6084	**0.6135**	0.6076
1000M	0.6109	0.6078	0.6029	0.6060	0.6055	0.6101	**0.6119**
best	0.6109	0.6177	0.6140	0.6212	0.6169	**0.6223**	0.6119
Grammar average (mean of the five grammaticality benchmarks)							
1M	0.5602	**0.5914**	0.5706	0.5702	0.5802	0.5626	0.5838
2M	0.5590	0.5853	0.5867	**0.5915**	0.5905	0.5845	0.5903
3M	0.5621	0.5955	0.5930	**0.5974**	0.5950	0.5878	0.5954
4M	0.5734	0.6029	0.5983	**0.6084**	0.5942	0.6016	0.6030
5M	0.5859	**0.6310**	0.6109	0.6187	0.6080	0.6092	0.6158
6M	0.5746	0.6315	0.6343	0.6335	0.6385	**0.6392**	0.6265
7M	0.5726	0.6543	0.6481	**0.6661**	0.6522	0.6654	0.6488
8M	0.5750	0.6753	0.6638	**0.6781**	0.6662	0.6712	0.6534
9M	0.5695	0.6819	0.6653	**0.6914**	0.6738	0.6837	0.6806
10M	0.5746	**0.6935**	0.6760	0.6934	0.6849	0.6842	0.6781
20M	0.5967	0.7393	0.7424	**0.7462**	0.7324	0.7439	0.7260
30M	0.6166	0.7579	0.7568	**0.7611**	0.7513	0.7559	0.7571
40M	0.6231	**0.7776**	0.7700	0.7766	0.7697	0.7731	0.7621
50M	0.6460	0.7805	**0.7821**	0.7800	0.7747	0.7800	0.7740
60M	0.6707	0.7847	0.7843	**0.7887**	0.7846	0.7810	0.7777
70M	0.6796	0.7855	0.7857	**0.7949**	0.7791	0.7915	0.7858
80M	0.6876	0.7921	0.7883	**0.7989**	0.7853	0.7930	0.7885
90M	0.6911	0.7991	**0.7993**	0.7985	0.7926	0.7947	0.7963
100M	0.7045	0.7948	0.7918	**0.8075**	0.7928	0.7935	0.7975
200M	0.7510	0.8106	0.8080	**0.8183**	0.8126	0.8128	0.8156
300M	0.7794	0.8166	0.8155	**0.8220**	0.8152	0.8191	0.8208
400M	0.7850	0.8174	0.8190	0.8220	0.8193	0.8190	**0.8226**
500M	0.7919	0.8141	0.8118	**0.8243**	0.8157	0.8233	0.8156
600M	0.7988	0.8127	0.8121	0.8192	0.8104	**0.8198**	0.8180
700M	0.7989	0.8061	0.8091	0.8150	0.8098	**0.8186**	0.8161
800M	0.8002	0.8017	0.8025	**0.8123**	0.8054	0.8107	0.8105
900M	0.8014	0.8017	0.7978	0.8043	0.8028	**0.8083**	0.8038
1000M	0.8027	0.8017	0.7928	0.7960	0.7958	0.8028	**0.8105**
best	0.8027	0.8174	0.8190	**0.8243**	0.8193	0.8198	0.8226

Table 7: Accuracy as a function of training budget. Rows are the chck_*N*M revisions, i.e. the number of training *words* seen (the trainer checkpoints). The shaded row corresponds to 100M-word budget for efficiency comparison. Higher is better.

Table 8: Accuracy of HAWK models on English BLiMP (*blimp_filtered*) by single phenomena.

Phenomenon	base	base bf	A	B	C	FiLM	switch
adjunct_island	0.800	**0.839**	0.789	0.752	0.777	0.772	0.807
anaphor_gender_agreement	0.929	**0.955**	0.904	0.919	0.931	0.919	0.935
anaphor_number_agreement	0.970	0.971	0.954	0.943	0.950	0.960	**0.979**
animate_subject_passive	0.674	0.724	0.677	**0.735**	0.723	0.726	0.685
animate_subject_trans	**0.817**	0.797	0.738	0.751	0.723	0.741	0.798
causative	**0.675**	0.667	0.654	0.623	0.617	0.655	0.670
complex_NP_island	0.402	0.398	0.381	0.450	0.431	**0.453**	0.345
coordinate_structure_constraint_complex_left_branch	0.529	0.471	0.525	0.538	0.557	**0.617**	0.602
coordinate_structure_constraint_object_extraction	**0.813**	0.753	0.681	0.700	0.679	0.670	0.739
determiner_noun_agreement_1	**0.978**	0.977	0.976	0.961	0.968	0.967	0.969
determiner_noun_agreement_2	**0.980**	**0.980**	0.962	0.934	0.952	0.934	0.960
determiner_noun_agreement_irregular_1	**0.865**	0.860	0.852	0.831	0.858	0.819	0.846
determiner_noun_agreement_irregular_2	0.891	0.902	**0.909**	0.878	0.905	0.906	0.895
determiner_noun_agreement_with_adj_2	0.932	**0.935**	0.912	0.896	0.891	0.897	0.920
determiner_noun_agreement_with_adj_irregular_1	0.811	0.816	0.811	0.801	0.823	0.791	**0.831**
determiner_noun_agreement_with_adj_irregular_2	0.839	0.823	0.832	0.788	0.825	0.826	**0.855**
determiner_noun_agreement_with_adjective_1	**0.953**	0.950	0.931	0.928	0.922	0.930	0.927
distractor_agreement_relational_noun	0.503	0.504	0.402	0.501	**0.536**	0.491	0.448
distractor_agreement_relative_clause	0.433	0.479	0.435	0.421	**0.503**	0.435	0.476
drop_argument	**0.771**	0.699	0.693	0.695	0.675	0.716	0.707
ellipsis_n_bar_1	**0.620**	0.613	0.611	0.559	0.517	0.526	0.539
ellipsis_n_bar_2	0.715	0.717	0.710	0.676	0.665	0.665	**0.754**
existential_there_object_raising	0.746	0.778	0.766	**0.780**	0.740	0.772	0.729
existential_there_quantifiers_1	**0.988**	0.956	0.941	0.937	0.926	0.941	0.947
existential_there_quantifiers_2	0.286	0.422	0.381	**0.599**	0.480	0.401	0.389
existential_there_subject_raising	0.831	**0.853**	0.834	0.842	0.807	0.813	0.827
expletive_it_object_raising	0.721	0.723	0.697	0.714	0.736	**0.744**	0.717
inchoative	0.542	0.556	0.557	0.563	0.530	**0.585**	0.513
intransitive	**0.660**	0.647	0.618	0.632	0.623	0.637	0.619
irregular_past_participle_adjectives	0.906	0.827	0.888	0.849	0.807	0.912	**0.973**
irregular_past_participle_verbs	0.774	0.790	0.736	0.745	**0.806**	0.736	0.709
irregular_plural_subject_verb_agreement_1	**0.765**	0.754	0.720	0.706	0.733	0.684	0.741
irregular_plural_subject_verb_agreement_2	0.731	0.765	0.753	0.752	0.731	0.739	**0.766**
left_branch_island_echo_question	0.467	0.419	0.482	0.588	0.568	**0.599**	0.496
left_branch_island_simple_question	0.659	0.580	0.621	0.621	0.625	**0.691**	0.650
matrix_question_npi_licensor_present	**0.389**	0.350	0.327	0.296	0.286	0.329	0.243
npi_present_1	0.622	0.593	0.589	0.579	**0.629**	0.600	0.558
npi_present_2	0.658	0.612	0.650	0.615	**0.667**	0.613	0.548
only_npi_licensor_present	**0.989**	0.955	0.891	0.781	0.878	0.956	**0.989**
only_npi_scope	**0.723**	0.631	0.528	0.594	0.624	0.553	0.507
passive_1	**0.870**	0.855	0.843	0.821	0.827	0.844	0.832
passive_2	0.807	**0.822**	0.821	0.814	0.790	0.816	0.808
principle_A_c_command	**0.536**	0.430	0.416	0.411	0.480	0.464	0.484
principle_A_case_1	**1.000**	**1.000**	0.997	0.992	0.998	0.997	**1.000**
principle_A_case_2	0.904	0.893	0.877	0.869	0.814	0.888	**0.917**
principle_A_domain_1	0.760	0.874	0.898	0.872	**0.916**	0.871	0.902

Continued on next page

Table 8 – continued

Phenomenon	base	base bf	A	B	C	FiLM	switch
principle_A_domain_2	0.643	**0.696**	0.562	0.616	0.593	0.645	0.632
principle_A_domain_3	0.577	**0.579**	0.493	0.570	0.506	0.567	0.536
principle_A_reconstruction	**0.392**	0.185	0.299	0.349	0.299	0.328	0.353
regular_plural_subject_verb_agreement_1	**0.831**	0.772	0.769	0.729	0.758	0.773	0.806
regular_plural_subject_verb_agreement_2	0.714	0.694	**0.729**	0.716	0.714	0.690	0.693
sentential_negation_npi_licensor_present	0.962	0.905	0.917	0.866	0.934	0.908	**0.968**
sentential_negation_npi_scope	0.551	0.467	**0.625**	0.431	0.485	0.526	0.476
sentential_subject_island	**0.480**	0.378	0.456	0.412	0.390	0.431	0.403
superlative_quantifiers_1	**0.724**	0.539	0.464	0.663	0.538	0.573	0.514
superlative_quantifiers_2	**0.875**	0.785	0.675	0.759	0.695	0.774	0.758
tough_vs_raising_1	0.360	0.323	0.325	**0.444**	0.389	0.440	0.359
tough_vs_raising_2	0.821	**0.841**	0.811	0.754	0.805	0.766	**0.841**
transitive	0.759	0.736	0.745	0.743	**0.765**	0.759	0.762
wh_island	0.710	0.655	0.606	0.647	0.675	**0.724**	0.515
wh_questions_object_gap	**0.782**	0.765	0.740	0.725	0.730	0.729	0.736
wh_questions_subject_gap	**0.910**	0.901	0.831	0.859	0.823	0.878	0.869
wh_questions_subject_gap_long_distance	0.881	**0.898**	0.823	0.775	0.786	0.777	0.887
wh_vs_that_no_gap	0.970	**0.980**	0.956	0.935	0.944	0.957	0.951
wh_vs_that_no_gap_long_distance	0.971	**0.984**	0.949	0.936	0.931	0.952	0.977
wh_vs_that_with_gap	0.313	0.260	0.293	**0.321**	0.288	0.272	0.318
wh_vs_that_with_gap_long_distance	0.063	0.035	0.067	0.095	**0.112**	0.105	0.064

Table 9: Accuracy of HAWK models on Dutch BLiMP (*blimp_nl*) by single phenomena.

Phenomenon	base	base bf	A	B	C	FiLM	switch
adpositional_phrases	0.980	0.955	0.965	0.955	0.945	0.975	**0.990**
adverbial_modification	0.925	0.930	0.940	0.885	**0.950**	0.900	0.940
anaphor_agreement	0.830	**0.860**	0.835	0.855	0.850	0.825	**0.860**
argument_structure	0.789	0.767	0.789	**0.796**	0.783	0.776	0.792
auxiliaries	0.942	**0.956**	0.940	0.864	0.910	0.926	0.898
binding_principle_a	0.955	**0.965**	0.845	0.945	0.925	0.950	0.930
complementive	0.844	0.840	0.772	0.796	0.812	0.830	**0.850**
crossing_dependencies	0.990	**1.000**	0.960	0.980	0.950	0.950	0.990
determiners	**0.807**	0.723	0.762	0.740	0.782	0.785	0.805
extraposition	**0.922**	0.892	0.818	0.882	0.802	0.842	0.902
finite_argument_clause	**0.843**	0.838	0.842	0.827	0.770	0.835	0.833
infinitival_argument_clause	0.833	**0.890**	0.841	0.824	0.816	0.840	0.869
nominalization	**0.900**	0.880	0.735	0.830	0.835	0.875	**0.900**
parasitic_gaps	0.226	0.251	0.294	**0.436**	0.281	0.346	0.149
passive	**0.838**	0.765	0.790	0.762	0.750	0.797	0.820
quantifiers	0.920	**0.940**	0.920	0.880	0.915	**0.940**	0.890
r_words	0.965	0.960	0.850	0.890	0.730	0.840	**0.995**
relativization	0.667	0.687	0.730	**0.797**	0.753	0.750	0.723
topicalization	0.695	0.613	0.623	0.605	0.563	0.647	**0.705**
verb_second	0.985	0.970	0.930	0.865	0.920	0.875	**0.995**
wh_movement	0.750	0.736	0.710	0.725	0.715	0.740	**0.760**
wh_movement_restrictions	0.741	0.743	0.677	0.704	0.661	0.704	**0.744**

Table 10: Accuracy of HAWK models on Chinese BLiMP (*zhoblimp*) by single phenomena.

Phenomenon	base	base bf	A	B	C	FiLM	switch
BA_BEI_subj_drop	0.920	**0.937**	0.840	0.797	0.787	0.723	0.837
BA_deletion	0.663	0.853	0.760	0.560	**0.927**	0.770	0.797
BA_duplicate_argument	**1.000**	**1.000**	0.993	0.953	0.957	0.997	**1.000**
BA_inversion	**0.903**	0.880	0.780	0.833	0.843	0.787	0.893
BA_meiba	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
BA_negation	0.770	0.877	0.760	0.757	0.577	0.680	**0.923**
BA_no_progressive	**0.997**	0.970	0.957	0.977	0.967	0.973	0.947
BA_no_stative_verb	0.180	0.153	0.247	0.333	0.247	0.073	**0.483**
BA_suo_adverbial_a	0.997	**1.000**	0.943	0.997	0.997	0.980	**1.000**
BA_suo_adverbial_b	**1.000**	**1.000**	**1.000**	**1.000**	0.997	**1.000**	**1.000**
BA_verb_le_a	**1.000**	**1.000**	0.993	0.993	0.990	0.987	**1.000**
BA_verb_le_b	0.787	**0.823**	0.670	0.670	0.707	0.570	0.713
BEI_construction_a	**0.873**	0.827	0.613	0.633	0.700	0.713	0.730
BEI_construction_b	**0.907**	0.857	0.707	0.803	0.700	0.740	0.850
BEI_deletion	0.300	0.417	0.653	0.573	0.647	**0.780**	0.770
BEI_preposition	0.967	**0.977**	0.857	0.807	0.833	0.763	0.960
PN_numP_a	**1.000**	0.993	0.997	0.990	0.997	0.997	**1.000**
PN_numP_b	**0.993**	0.983	0.827	0.893	0.853	0.870	0.990
adjective_transitive_dui	0.813	0.720	**0.887**	0.790	0.737	0.877	0.787
agent_animacy_adv	**1.000**	**1.000**	0.983	0.980	0.970	0.980	**1.000**
agent_animacy_passive	0.630	**0.733**	0.693	0.713	0.723	0.717	0.680
agent_animacy_subj	0.633	0.710	0.680	**0.747**	0.687	0.680	0.663
agent_causative	0.690	0.687	0.623	0.653	0.713	0.677	**0.753**
agent_deletion	0.767	0.593	**0.850**	0.640	0.643	0.647	0.703
anaphor_gender_agreement	0.560	0.577	0.530	0.547	0.540	**0.583**	0.557
anaphor_number_agreement	0.237	0.117	0.143	**0.333**	0.253	0.217	0.050
causative_shi_ba	0.740	0.713	0.727	0.627	0.730	0.700	**0.873**
classifier_noun_agreement	**0.990**	0.957	0.977	0.977	0.953	0.953	0.987
classifier_noun_agreement_no_gap	0.237	0.283	0.230	0.327	0.250	**0.343**	0.187
classifier_noun_subj	**1.000**	0.993	0.913	0.963	0.927	0.870	**1.000**
control_modal_vs_raising_modal	0.997	**1.000**	0.970	0.957	0.993	0.947	0.990
ellipsis_adj	0.463	0.487	0.500	**0.543**	0.513	0.500	0.510
ellipsis_double_object	0.990	0.947	0.947	0.983	0.957	0.980	**1.000**
ellipsis_n_bar_class	**0.767**	0.660	0.683	0.690	0.683	0.670	0.733
existential_there_subject_raising	0.807	**0.810**	0.630	0.753	0.687	0.633	0.747
fci_renhe_dou	**1.000**	**1.000**	0.930	0.810	0.987	0.997	0.993
fci_renhe_prepP	**1.000**	**1.000**	**1.000**	0.997	**1.000**	**1.000**	**1.000**
fci_renhe_ruguo	0.997	**1.000**	0.970	0.880	0.993	0.840	**1.000**
fci_renhe_subj	0.880	**1.000**	0.933	0.497	0.393	0.733	0.823
fci_renhe_suoyou	**1.000**	**1.000**	**1.000**	0.993	**1.000**	0.993	**1.000**
intransitive_double_obj	**0.917**	0.877	0.893	0.843	0.817	0.763	0.867
intransitive_no_obj	**0.707**	0.700	0.703	0.623	0.653	0.680	0.690
left_adverbial_b	**0.970**	**0.970**	0.933	0.920	0.913	0.940	**0.970**
left_adverbial_d	0.960	0.927	0.873	0.917	0.907	0.933	**0.970**
left_adverbial_e	0.833	0.883	0.793	0.743	0.790	0.627	**0.890**
left_adverbial_negation	**0.780**	0.663	0.637	0.680	0.723	0.637	0.727
left_dou	**1.000**	**1.000**	0.973	0.963	0.993	0.987	**1.000**
modal_raising_hui	0.943	**1.000**	0.823	0.887	0.900	0.900	0.943
modal_raising_topicalization	0.890	0.797	0.820	0.780	0.753	0.813	**0.933**
nominal_definite_men	**0.997**	0.987	0.983	0.973	0.963	0.987	**0.997**
nominal_modal_insertion	0.733	0.753	0.650	0.693	**0.843**	0.657	0.833
noun_adjective_shi	0.630	0.593	**0.683**	0.580	0.620	0.670	0.630

Continued on next page

Table 10 – continued

Phenomenon	base	base bf	A	B	C	FiLM	switch
noun_phrase_conjunction_jian	0.983	0.973	0.970	0.970	0.983	**0.993**	**0.993**
npi_renhe_A_not_A_question	0.043	0.097	0.137	**0.330**	0.233	0.087	0.027
npi_renhe_conditional	0.047	0.073	0.017	0.003	**0.227**	0.013	0.017
npi_renhe_neg_scope_locP	0.890	**0.933**	0.437	0.680	0.727	0.700	0.617
npi_renhe_neg_scope_subj	0.997	**1.000**	0.957	0.953	0.990	0.920	0.997
npi_renhe_wh_question_obj	0.883	**0.990**	0.847	0.880	0.827	0.957	0.950
npi_renhe_wh_question_subj	0.737	0.760	0.570	0.713	0.707	0.727	**0.860**
passive_agent_deletion_long_left	0.250	0.370	0.693	0.540	**0.850**	0.687	0.283
passive_agent_deletion_long_right_a	0.003	0.000	0.250	0.287	**0.293**	0.200	0.013
passive_agent_deletion_long_right_b	0.000	0.000	0.013	0.013	**0.027**	0.017	0.000
passive_agent_deletion_short	0.000	0.003	0.100	0.103	0.130	**0.440**	0.000
passive_body_part	0.863	0.907	0.857	**0.920**	0.873	0.913	0.817
passive_intransitive	0.890	**0.910**	0.820	0.867	0.843	0.840	0.897
passive_no_adj	**0.943**	0.920	0.927	0.903	**0.943**	0.900	**0.943**
passive_suo	0.207	0.467	0.427	0.457	**0.517**	0.487	0.410
plural_cardinal_men_a	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
plural_cardinal_men_b	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
preposition_deletion	**0.967**	0.910	0.847	0.817	0.843	0.853	0.927
preposition_insertion	**0.987**	0.977	0.890	0.943	0.927	0.953	**0.987**
principle_A_c_command	0.417	**0.567**	0.490	0.440	0.430	0.450	0.373
principle_A_c_command_number	0.197	0.100	0.030	**0.377**	0.230	0.227	0.063
principle_A_domain	0.433	0.427	0.373	0.413	0.313	**0.447**	0.390
principle_A_domain_number	0.987	**0.990**	**0.990**	0.853	0.963	0.947	0.980
question_A_not_A	**1.000**	0.983	0.910	0.910	0.960	0.950	0.983
question_A_not_A_daodi_a	0.937	0.967	0.903	0.943	0.943	0.803	**0.993**
question_A_not_A_daodi_b	0.940	0.963	0.907	0.923	0.893	0.820	**0.980**
question_A_not_A_indirect	**1.000**	**1.000**	**1.000**	0.993	0.990	**1.000**	**1.000**
question_V_not_VP_1	**0.920**	0.810	0.713	0.800	0.847	0.480	0.857
question_V_not_VP_2	0.943	0.560	0.873	0.740	0.657	0.440	**1.000**
question_daodi_nandao_1	**1.000**	0.970	0.863	0.937	0.977	0.937	**1.000**
question_daodi_nandao_2	0.707	**0.940**	0.740	0.590	0.687	0.783	0.577
question_daodi_nandao_A_not_A_intran	0.580	**0.973**	0.593	0.737	0.830	0.723	0.530
question_daodi_nandao_A_not_A_tran	0.803	0.790	0.717	0.793	**0.857**	0.767	0.677
question_daodi_negation	0.863	**0.943**	0.927	0.610	0.690	0.770	0.777
question_nandao_negation	**1.000**	**1.000**	0.997	0.993	**1.000**	0.997	0.997
question_nandao_raising_1_a	**1.000**	**1.000**	**1.000**	**1.000**	0.993	**1.000**	**1.000**
question_nandao_raising_1_b	**1.000**	**1.000**	**1.000**	0.973	0.997	0.997	**1.000**
question_nandao_raising_2	**0.980**	0.930	0.920	0.910	0.943	0.880	0.877
question_nandao_raising_3	0.997	0.997	0.893	0.920	0.923	0.940	**1.000**
question_nandao_scope_1	0.913	0.757	0.920	0.917	0.800	**0.957**	0.660
question_nandao_scope_2	**0.887**	0.853	0.747	**0.887**	0.793	0.773	0.860
question_particle_daodi_choice_intran	0.943	0.923	0.983	0.963	**0.993**	0.957	0.987
question_particle_daodi_choice_tran	0.993	**1.000**	0.970	0.970	0.993	0.993	**1.000**
question_particle_nandao	0.777	0.600	0.627	0.537	0.570	0.760	**0.943**
relative_operator_intepretation	0.290	0.287	0.553	**0.643**	0.337	0.250	0.313
relative_operator_who	0.990	**1.000**	0.993	0.993	0.983	0.940	0.990
relativization_movement_no_gap	**1.000**	**1.000**	**1.000**	0.990	0.997	0.987	**1.000**
relativization_movement_when_where	**0.997**	0.993	0.957	0.970	0.957	0.960	0.993
renhe_no_episodic_sentences	0.697	**0.917**	0.550	0.840	0.893	0.630	0.887
renhe_no_superordinate_negation	**0.980**	0.813	0.953	0.777	0.970	0.763	0.940
renhe_non_factive_verb	**0.643**	0.347	0.627	0.470	0.593	0.520	0.513
right_yijing_a	0.943	0.983	0.863	0.940	0.890	0.827	**0.993**
right_yijing_b	0.637	0.773	0.660	0.710	0.507	0.630	**0.857**
singular_PN_but_plural_pron	0.953	0.943	0.957	**0.977**	0.973	0.933	0.940
superlative_quantifiers_1	0.383	0.180	0.097	0.037	0.130	**0.557**	0.360
superlative_quantifiers_2	0.967	0.907	0.723	0.917	0.683	0.783	**0.983**
topicalization_OSV	**0.970**	0.953	0.907	0.907	0.823	0.923	0.957
topicalization_OSV_mei	0.970	**0.977**	0.860	0.923	0.890	0.870	0.970
topicalization_SOV	0.903	0.920	0.900	0.887	0.843	0.913	**0.933**
topicalization_SOV_mei	**0.973**	0.940	0.893	0.943	0.843	0.860	0.943
verb_negation_particle	**0.980**	0.957	0.943	0.813	0.887	0.897	0.967
verb_phrase_left_adverbial	0.957	**0.987**	0.890	0.810	0.850	0.917	0.947
verb_phrase_left_negation	0.883	**0.947**	0.747	0.740	0.590	0.777	**0.947**
ya_insertion	**1.000**	0.987	0.920	0.883	0.697	0.930	0.967
you_quantifier_adj	0.977	0.973	0.977	0.947	**0.993**	0.953	0.983
you_yige	0.953	0.923	0.817	0.923	0.807	0.790	**0.963**

Scaling Compute Instead of Parameters: Exploring Iterative Transformers for Data-Efficient Language Modeling

Wiktor Kamzela[1] **and Ondřej Dušek**[2] **and Mateusz Lango**[1,2]
[1]Poznan University of Technology, Faculty of Computing and Telecommunications, Poland
[2]Charles University, Faculty of Mathematics and Physics, Czechia
wiktor.kamzela@student.put.edu.pl, {odusek,lango}@ufal.mff.cuni.cz

Abstract

Neural language model architectures that iterate over their layers have recently shown superior performance on reasoning benchmarks and proved to be a promising method for reducing model size while maintaining performance at scale. However, the effectiveness of such architectures in data-constrained settings remains unexplored. We present a systematic study of layer-sharing transformer architectures trained on the BabyLM corpus (10M tokens), across various iteration strategies and a range of layer and iteration counts. We find that iterative architectures are highly effective under data-constrained setting: a 3-layer iterated model outperforms a standard 24-layer transformer model (ModernBERT-inspired) on SuperGLUE benchmark.

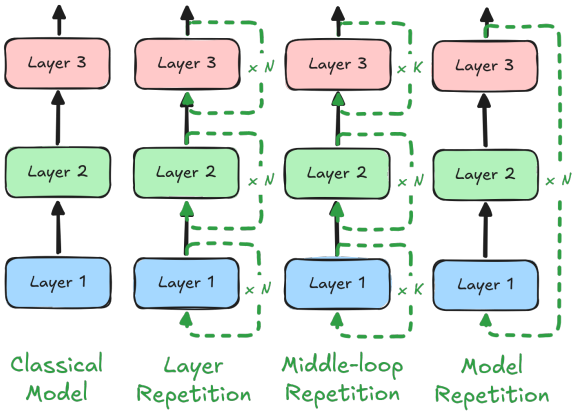


Figure 1: Overview of the classic transformer model and three iteration strategies explored in this paper: layer repetition, middle-loop repetition and model repetition.

1 Introduction

Over the past several years, the dominant strategy for improving pretrained large language models (LLM) has been to scale model size, following empirical scaling laws (Kaplan et al., 2020) that associate larger architectures with better performance. However, this approach substantially increases memory requirements and, according to compute-optimal scaling laws, also requires substantially more training data as model size grows (Hoffmann et al., 2022). With LLMs already reaching hundreds of billions of parameters, further parameter scaling is becoming increasingly inefficient. Consequently, there is growing interest in alternative architectures that can achieve comparable or superior performance with fewer parameters.

One promising direction are *iterative architectures*, in which the same layer is applied multiple times to hidden representations (Yang et al., 2024; Saunshi et al., 2025; Bae et al., 2025). When unrolled, such architectures are equivalent to much deeper networks whose layers share parameters, while substantially reducing memory requirements. Since these techniques are designed to improve performance while reducing the number of trainable parameters, they may be particularly well suited to small-data settings, where a smaller effective capacity can mitigate overfitting, echoing the classical bias-variance trade-off (Hastie et al., 2009).

In this work, we present the first systematic study of iterative, layer-sharing architectures for language modeling in a strictly data-constrained setting, using developmentally plausible corpora (10M tokens BabyLM corpus, Haga et al. 2024). We investigate three families of iterative architectures, explore iteration-specific specialization via LoRA adapters (Hu et al., 2022), as well as investigate further parameter reduction through smaller input representations (reduced vocabulary size and embedding layer factorization). Our experiments span architectures ranging from 1 to 24 transformer layers and from 2 to 16 recursive iterations, resulting in a comprehensive exploration of the trade-offs between model depth, recursion, and parameter efficiency.

The results demonstrate the effectiveness of iterative computation in data-constrained regimes.

Proceedings of the Second BabyLM Workshop, pages 678–688
October 29, 2026 ©2026 Association for Computational Linguistics

For instance, we discover that a model containing only three distinct layers, repeatedly applied through recursion, surpasses the performance of a conventional 24-layer ModernBERT-based architecture (Warner et al., 2025) on Super-GLUE benchmark (Wang et al., 2019) while using the same training data and substantially fewer trainable parameters. This highlights the potential of parameter-sharing architectures for efficient learning from limited data. To boost further research on iterative architectures, we make our code and models publicly available.[1]

2 Related Works

Iterative language models In the context of language modeling, iterative computation has previously proven highly effective in models such as ALBERT (Lan et al., 2020), which combined cross-layer parameter sharing (i.e., single-layer repetition throughout the whole model) with embedding factorization to dramatically reduce model size while maintaining competitive performance.

More recently, the concept of iterative computation has been revisited to improve LLM reasoning capabilities. Geiping et al. (2025) proposed recurrent-depth architecture to scale test-time computation and perform reasoning in implicit latent space. Saunshi et al. (2025) further explored latent thoughts by analyzing transformer models that iterate over layers multiple times, showing their higher efficiency on reasoning benchmarks. These architectures can be extended by adding learnable adapters, individual for every iteration, that make the repeated layer act differently in every iteration while maintaining low parameter count (Bae et al., 2025). Many other different iteration schemas have been proposed, such as dynamic selection of the number of iterations basing on token difficulty (Chen et al., 2025) or iterating in the space of word embeddings (Zeng et al., 2026).

The effort to apply iterative language models at large scale showed that such models achieve competitive performance with several times larger state-of-the-art language models (Zhu et al., 2026).

While these works demonstrate the promise of iterative architectures at large scale or for reasoning-heavy tasks, their effectiveness in data-constrained regimes remains unexplored.

Building models on developmentally plausible corpora Several editions of the BabyLM workshop (Charpentier et al., 2025; Hu et al., 2024; Warstadt et al., 2023) provided significant insight into techniques effective for data-constrained language modeling on developmentally plausible corpora.

While many different training and data techniques were explored, for instance curriculum learning (Roque and Velasco, 2025), specialized training objectives (Kamzela et al., 2025) or data creation (Ulm et al., 2025), works exploring new architectures are less common: Charpentier and Samuel (2023) explored adding layer weighting to the LTG-BERT architecture (Samuel et al., 2023). Other works explored different variants of recurrent neural networks (Haller et al., 2024), state space models (Lee et al., 2025), replacing attention with mLSTM token mixer (Haller et al., 2025) or training only LoRA adapters with specifically initialized layers (Kumar et al., 2025). An iterative architecture, to the best of our knowledge, was not explored.

3 Iterative Language Models

The classical transformer architecture (Vaswani et al., 2017) used for language modeling is built out of several layers $f(x)$ consisting of self-attention and fully-connected sublayers. The entire transformer network consisting of L such layers can be denoted as:

$$y(x) = (f_L \odot \cdots \odot f_2 \odot f_1)(x) \triangleq \bigodot_{i=1}^{L} (f_i)(x)$$

where $\odot$ is the compound function operator.

3.1 Three Iterative Architectures

Iterative transformer architectures aim to obtain higher performance by allowing the model to compute longer, without increasing the number of layers or the number of learnable parameters. This is achieved by iterating over the same layers multiple times and applying them to process the intermediate hidden representations. Such models are more parameter-efficient by reusing a layer's parameters multiple times, training them to perform operations generalizable across iterations, thus effectively achieving higher model depth without actually increasing the number of layers. In this work we explore three main iteration strategies visualized in Fig. 1 and described below.

[1] https://github.com/Wector1/BabyLM_iterative

Layer Repetition is an iteration strategy which applies the same layer consecutively to process the hidden representation and only then returning the result to the next layer. This can be alternatively seen as a model with several first layers, followed by several second layers, etc., as shown in Fig. 2 on the left.

Formally, the Later Repetition architecture with L layers and N iterations is defined as:

$$y(x) = \bigodot_{i=1}^{L} \left(\bigodot_{j=1}^{N} f_i \right) (x)$$

Model Repetition iterates over the entire, L-layered model instead of iterating over layers:

$$y(x) = \bigodot_{j=1}^{N} \left(\bigodot_{i=1}^{L} f_i \right) (x)$$

This can be thought of as a deeper network with $N \cdot L$ layers, constructed by stacking N models on top of each other.

Middle-loop Repetition is inspired by mechanistic interpretability studies (Wendler et al., 2024; Lad et al., 2025), suggesting a simplified conceptual view that LLM inference consists of three coarse phases: encoding the input, reasoning and synthesizing (including knowledge retrieval), and forming the final answer by choosing appropriate tokens. The reasoning phase is considered to be the most complex one, spread across most layers of LLMs (Lad et al., 2025).

Therefore, we propose a 3-layer architecture, reflecting three general operations of the entire processing. While the first and last layer are repeated K times, the middle layer is applied a higher number of M times ($M > K$):

$$y(x) = \left(\bigodot_{j=1}^{K} f_3 \right) \odot \left(\bigodot_{j=1}^{M} f_2 \right) \odot \left(\bigodot_{j=1}^{K} f_1 \right) (x)$$

We assume $N = 2K + M$ for general statements about iterative models.

3.2 Relaxed Iterative Model

Iterative architectures act as deeper $N \cdot L$ layer models while having significantly less parameters, which is achieved by applying the same layer several times. This effectively means, that such models are less expressive than standard $N \cdot L$ layer models because they apply exactly the same operation without any variability.

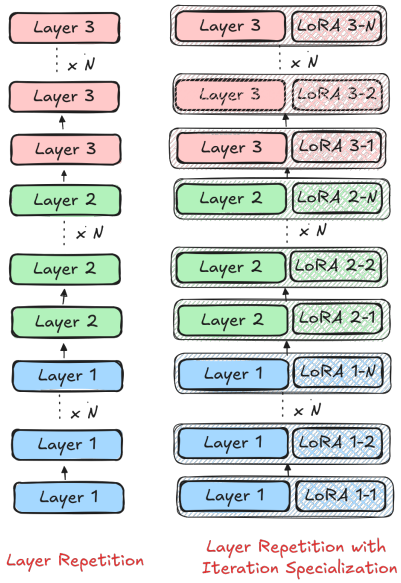


Figure 2: Unrolled visualization of standard Layer Repetition iteration strategy and its variant with iteration specialization.

If we slightly relax the assumption that the number of model parameters does increase at all with the number of iterations chosen, we can enable the model to perform different operations in different iterations, potentially bringing their expressivity closer to deeper classical models while still requiring only a fraction of their parameters. The idea is that while the same layer is applied multiple times to process the hidden representations, it is applied in every iteration with a different LoRA adapter (Hu et al., 2022) that alters the computed operation. In this way, the model has different $N \cdot L$ layers while still having the number of parameters significantly reduced in comparison to classical $N \cdot L$ model. Such architecture is visualized in Fig. 2 on the right and defined below:

$$y(x) = \bigodot_{i=1}^{L} \left(\bigodot_{j=1}^{N} f_i \odot LoRA_{ij} \right) (x)$$

3.3 Embedding Compression

As the parameter count of the smallest iterative models we test is largely influenced by the significant size of the embedding matrix, we explore two additional techniques that reduce the number of trainable parameters: embedding factorization and custom tokenizers.

Embedding factorization was adapted from earlier works on ALBERT (Lan et al., 2020), that

proposed the decomposition of the embedding matrix into a shared linear layer and an embedding matrix with reduced dimensionality. Each token has an embedding with a smaller dimensionality d, which is then scaled up to the size of model's hidden representation h by a linear layer. Since the original embedding matrix has $|V| \times h$ parameters and the new setup has $h \times d + |V| \times d$ parameters, as long as $d << h$, a reduction of parameter count is achieved.

Custom tokenizers Another possibility to reduce the size of embedding matrix is to limit the size of vocabulary $|V|$ the tokenizer operates on. To this end, we used the WordPiece algorithm, originally developed for BERT (Devlin et al., 2019), to construct tokenizers on training corpora that have a smaller vocabulary set V.

4 Experiments

4.1 Experimental Setup

Model architecture Our models are based on ModernBERT (Warner et al., 2025) implementation from the HuggingFace library (Wolf et al., 2020). Our baseline model has 149M parameters, closely following the architecture of original ModernBERT-small with default parameters, except for the number of hidden layers set to 24, the context size of 256 and sliding attention disabled. We tested all three iteration strategies as well additional techniques mentioned in Sec. 3.

Iteration strategies We evaluated models with the number of layers $L = \{1, 3, 6, 12, 24\}$ and iterations $N = \{1, 2, 4, 8, 16\}$, except for the largest models, where computational constraints prevented evaluating $N = 16$ for $L = 12$ and $N = \{16, 8\}$ for $L = 24$. The Middle-loop Repetition variant is by definition $L = 3$ and was tested for $K = \{1, 3\}$ and values of M that will result in architectures having $M + 2K$ unrolled layers in the set of $\{6, 12, 24, 48\}$.

Relaxed iterative models were trained with LoRA adapters of dimensionality 64 and $\alpha = 128$. The models are trained with the adapters from the very first epoch and the training was performed without further changes.

Embedding Compression Embedding factorization was applied with an embedding matrix with a dimensionality of 128. The embeddings were scaled up to the model's 768 dimensionality by an additional linear layer.

As a custom tokenizer, we trained a WordPiece model on the BabyLM 10M-word corpus using the HuggingFace tokenizers library. Prior to vocabulary construction, text was normalized via NFD decomposition, lowercasing, and accent stripping, then pre-tokenized by splitting on whitespace and punctuation. The tokenizer was trained with a vocabulary size of 24,000 and a minimum subword frequency of 2. Note that the original ModernBERT tokenizer has a 50,368 vocabulary size.

Training details Models were trained with AdamW optimizer with 64 batch size on masking modeling objective with masking probability of $p = 0.4$, following the initial setup of Edman and Fraser (2025). Learning rate followed the cosine schedule with 500 warmup steps and a learning rate of 0.0003. A small weight decay of 0.01 was applied. The models were trained for 10 epochs with the best checkpoint selected basing on the best perplexity value on validation set.

Evaluation Our evaluation protocol follows the one provided by the BabyLM Challenge organizers (Charpentier et al., 2025). More concretely, we evaluated on two benchmarks:

- SuperGLUE, which measures language understanding capabilities on tasks like BoolQ, MNLI, MultiRC, RTE, WSC, MRPC, and QQP (Wang et al., 2019);
- BLiMP (the Benchmark of Linguistic Minimal Pairs), consisting of 67 sub-datasets of minimal sentence pairs probing syntax, morphology, and semantics (Warstadt et al., 2020).

Additionally, masked language modeling perplexity was also measured.

For each SuperGLUE task, pretrained models were fine-tuned with default parameters provided by BabyLM organizers, without any further hyperparameter tuning.

5 Results

5.1 Layer Repetition

The results of the Layer Repetition strategy across 5 layer sizes and 5 iteration numbers are shown in Tab. 1. For the baseline model ($N = 1$), the results on both benchmarks grow as expected with the increasing number of hidden layers. Iterative models, while having the same number of parameters, achieve superior results in comparison to the

L (Layers)	N (Iter.)	Params	S. GLUE	BLiMP	Perp.
1	1	44.3M	62.96	61.69	17.69
	2	44.3M	64.66	63.47	14.69
	4	44.3M	63.91	64.07	10.81
	8	44.3M	**65.14**	**66.58**	**9.91**
	16	44.3M	64.13	57.90	28.56
3	1	54.4M	63.59	66.74	10.27
	2	54.4M	65.14	66.60	9.63
	4	54.4M	**65.75**	68.08	9.46
	8	54.4M	62.45	**69.40**	9.54
	16	54.4M	64.92	68.06	**8.95**
6	1	69.4M	65.73	68.12	9.42
	2	69.4M	65.00	68.56	8.79
	4	69.4M	65.66	69.54	**8.58**
	8	69.4M	**66.75**	69.57	8.59
	16	69.4M	65.07	**70.04**	9.23
12	1	99.5M	65.54	70.36	8.42
	2	99.5M	65.55	71.33	**8.04**
	4	99.5M	65.61	70.15	9.01
	8	99.5M	**66.34**	**71.46**	8.77
24	1	159.7M	66.04	71.84	**8.34**
	2	159.7M	65.72	**72.25**	8.37
	4	159.7M	**67.54**	71.50	8.53

Table 1: The results of Layer Repetition strategy across different numbers of layers L (Layers) and the number of iterations N (Iter.). Note that the results for $N = 1$ are standard transformer models. The number of model parameters (Params) and results of Super GLUE (S. GLUE), BLiMP and Perplexity (Perp.) also reported.

L (Layers)	N (Iter.)	Rep. schema	S.GLUE	BLiMP
1	8	Model rep.	65.08	66.54
		Layer rep.	**65.14**	**66.58**
	16	Model rep.	**64.16**	**58.36**
		Layer rep.	64.13	57.90
3	8	Model rep.	**66.43**	**69.40**
		Layer rep.	62.45	**69.40**
	16	Model rep.	63.97	59.30
		Layer rep.	**64.92**	**68.06**
6	2	Model rep.	**66.07**	**68.72**
		Layer rep.	65.00	68.56
	4	Model rep.	**66.83**	**70.22**
		Layer rep.	65.66	69.54
	8	Model rep.	**66.81**	**69.82**
		Layer rep.	66.75	69.57
12	2	Model rep.	**66.14**	70.47
		Layer rep.	65.55	**71.33**
	4	Model rep.	**67.10**	**71.05**
		Layer rep.	65.61	70.15
24	1	Baseline	**66.04**	**71.84**
	2	Model rep.	**66.74**	72.00
		Layer rep.	65.72	**72.25**

Table 2: Comparison of Layer and Model Repetition schema on models with different number of layers L (Layers) and number of iterations N (Iter.).

baseline and their performance seem to generally increase with increasing the number of iterations.

Impressively, the iterative model with a single layer and $N = 8$ iterations achieves a SuperGLUE performance comparable to a baseline model with 12 trainable layers. This shows the effectiveness of iterative models in modeling complex functions with a limited number of layers, and the potential of scaling the computation time while keeping the parameter count fixed.

The best iterative model in our comparison is the iterative model with $L = 24$ layers and $N = 4$ iterations. The best baseline has also $L = 24$ layers and achieves 1.5 p.p. worse result on SuperGLUE and a slightly better result on BLiMP (0.34 p.p., within standard deviation) than the iterative counterpart.

5.2 Iterative Schema Comparison

Model Repetition is compared to Layer Repetition in Tab. 2. It appears to work better than Layer repetition overall; however, the differences between the architectures are typically within ± 1 p.p. for both benchmarks. Interestingly, even Model repetition with $N = 8$ and $L = 3$ layers only achieves better SuperGLUE than the biggest baseline model with $L = 24$. BLiMP seems to be a more challenging benchmark for iterative models, and the improvements over the biggest baseline are obtained only by iterating a model of a similar size.

Middle-loop Repetition A more detailed evaluation on $L = 3$ layered architectures in Table 3 reveals that the differences between the best and the worse iteration schema increase with the increase of the number of iterations. Most importantly, it seems that Layer and Model repetition do not work well in high iterations regimes (48 unfolded layers) and increase of iteration count does not bring further benefits. In contrast, the interpretability-inspired Middle-loop Repetition schema is more stable in high iteration regimes, bringing consistent improvements on SuperGLUE. These models end up having better SuperGLUE than the baseline with $L = 24$ and better BLiMP than the baseline $L = 6$, while reducing the number of trainable layers by 8 and 2 times, respectively.

5.3 Impact of Iteration Specialization

Adding iteration specialization seems to only have a moderate positive impact on iterative models, as

Layers	Unrolled	Iteration schema	Super GLUE	BLiMP
3	6	Layer Rep.	**65.14**	66.60
		Model Rep.	64.72	67.95
		Mid.-loop Rep. ($K = 1$)	64.72	**67.97**
3	12	Layer Rep.	65.75	68.08
		Model Rep.	**66.45**	**69.32**
		Mid.-loop Rep. ($K = 1$)	65.60	68.75
		Mid.-loop Rep. ($K = 3$)	65.81	68.60
3	24	Layer Rep.	62.45	69.40
		Model Rep.	**66.43**	69.40
		Mid.-loop Rep. ($K = 1$)	66.14	69.30
		Mid.-loop Rep. ($K = 3$)	66.41	**69.42**
3	48	Layer Rep.	64.92	68.06
		Model Rep.	63.97	59.30
		Mid.-loop Rep. ($K = 1$)	66.68	68.41
		Mid.-loop Rep. ($K = 3$)	**67.95**	**68.78**

Table 3: Comparison of iteration schemas for models with $L = 3$ hidden layers and a different number of iterations N. For easier comparisons, we report the number of layers in unrolled architectures (Unrolled). For Layer and Model Repetition schemas, the number of iterations N can be computed by dividing this number by $L = 3$. For Middle-loop Repetition, the value of $2K$ needs to be subtracted from it to get M.

shown in the results presented in Fig. 3 (see App.A for raw results). Adding adapters to the baseline architectures results in an inconsistent and small effect on BLiMP but – quite surprisingly – a positive effect on the SuperGLUE benchmark (except for the largest $L = 24$ model). Note that the latter benchmark requires model finetuning, which may indicate better suitability for finetuning. However, the general lack of consistency on SuperGLUE results suggests that the hyperparameters on the standard BabyLM pipeline used for evaluation may be suboptimal for iterative architectures.

Iteration specialization improve results on BLiMP for almost all the values of N and smaller numbers of layers $L \leq 6$, with larger increases for higher values of N. For a larger model ($L = 12$), iteration specialization provides better results only for the highest $N = 4$ tested. For the largest $L = 24$ models, we do not observe any improvement. However, we only tested $N = \{1, 2\}$ in this case due to computational limitations.

In contrast, the results on SuperGLUE are typically worse for models with specialized iterations with the number of iterations N growing. Since this benchmark uses fine-tuning to assess model capabilities, it is possible that the default finetuning parameters are suboptimal for this kind of models.

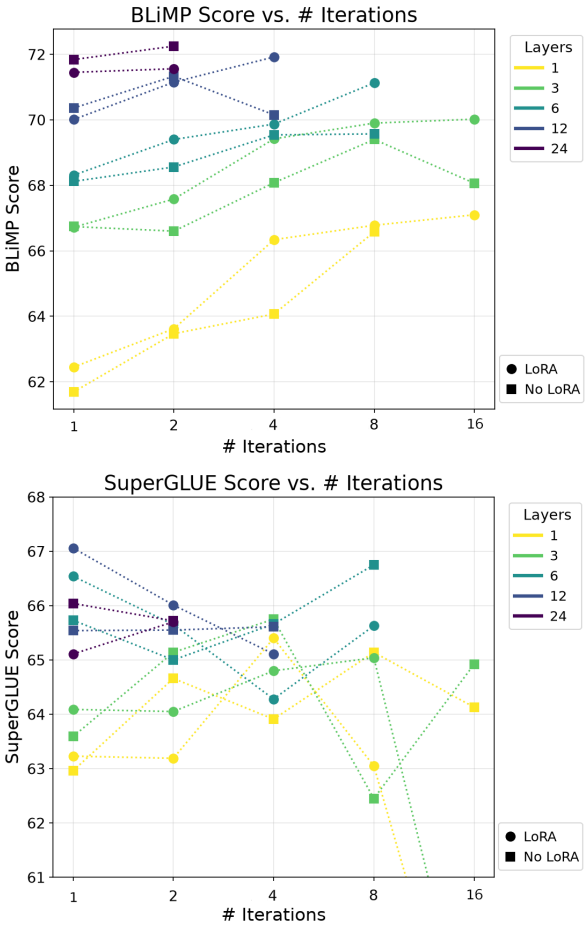


Figure 3: Visualization of models' results with iteration specialization via LoRA compared with standard Layer Repetition iterative models across different numbers of hidden layers L (Layers) and number of iterations N (# Iterations). See the raw numbers in Tab. 5.

Different LoRA parameterizations In Appendix A, we present results with two additional ways of adding LoRA parametrization for consecutive iterations. As LoRA adapters are typically used for fine-tuning the models, we trained the standard iterative model for first few epochs, adding LoRA only in the 8th or 9th epoch. This was accompanied with an increase of learning rate to 0.0003 to allow faster training of newly added parameters.[2] Nevertheless, the standard technique of having iteration specialization from the beginning of training seems to yield the best results.

5.4 Embedding Compression

Results for models constructed with different input representation techniques are shown in Tab. 4; all are reported for the Layer Repetition schema.

[2]In our preliminary experiments, it offered better results than keeping the original learning rate.

Lay.	Iter.	Tok.	Fac.	Param	S.GLUE	BLiMP
1	8	M.BERT	Yes	11.7M	64.55	66.49
		M.BERT	No	44.3M	65.14	**66.58**
		Custom	Yes	8.3M	**65.21**	66.51
		Custom	No	24.1M	64.81	65.82
	16	M.BERT	Yes	11.7M	34.96	49.85
		M.BERT	No	44.3M	64.13	57.90
		Custom	Yes	8.3M	63.93	51.88
		Custom	No	24.1M	**65.23**	**59.58**
3	8	M.BERT	Yes	21.7M	66.43	**69.44**
		M.BERT	No	54.4M	62.45	69.40
		Custom	Yes	18.3M	**68.98**	69.14
		Custom	No	34.1M	64.97	67.47
	16	M.BERT	Yes	21.7M	**67.72**	**69.19**
		M.BERT	No	54.4M	64.92	68.06
		Custom	Yes	18.3M	67.17	68.22
		Custom	No	34.1M	60.76	67.47
6	8	M.BERT	Yes	36.8M	66.76	69.95
		M.BERT	No	69.4M	66.75	69.57
		Custom	Yes	33.4M	**68.73**	**70.00**
		Custom	No	49.1M	67.87	68.68
12	1	M.BERT	Yes	66.9M	68.61	69.87
		M.BERT	No	99.5M	65.54	**70.36**
		Custom	Yes	63.5M	**68.98**	68.68
		Custom	No	79.2M	67.34	69.20
	2	M.BERT	Yes	66.9M	67.47	70.56
		M.BERT	No	99.5M	65.55	**71.33**
		Custom	Yes	63.5M	**70.12**	69.42
		Custom	No	79.2M	67.39	70.67
	4	M.BERT	Yes	66.9M	67.76	**70.76**
		M.BERT	No	99.5M	65.61	70.15
		Custom	Yes	63.5M	**69.58**	69.86
		Custom	No	79.2M	67.32	69.78
24	1	M.BERT	Yes	127.1M	68.13	70.73
		M.BERT	No	159.7M	66.04	**71.84**
		Custom	Yes	123.7M	**69.26**	70.60
		Custom	No	139.4M	68.08	69.93
	2	M.BERT	Yes	127.1M	68.99	70.72
		M.BERT	No	159.7M	65.72	**72.25**
		Custom	Yes	123.7M	**69.23**	69.87
		Custom	No	139.4M	67.36	71.57

Table 4: Comparison of results obtained with ModernBERT and Custom Wordpiece tokenizer, with or without embedding factorization (Fac.) under Layer Repetition schema.

Combining a custom tokenizer with smaller vocabulary size and embedding matrix factorization, results in approx. 36M decrease in parameter count. For the smallest of our models with $L = 1$, this is equal to more than 5 fold compression of the model size. Interestingly, the impact of this reduction in parameter size often results in an improved predictive performance.

Embedding matrix factorization applied across both tokenizers results in an improved SuperGLUE score for almost all layer and iteration numbers. It also improved results on BLiMP for a smaller architecture with 3 and 6 layers, but caused a drop – typically up to 2 p.p. – on other architecture sizes.

The Original ModernBERT tokenizer proved to enable better scores on the zero-shot BLiMP benchmark with only two exceptions: $L = 1, N = 16$ and $L = 6, N = 8$ configurations. On the other hand, our custom tokenizer enabled higher scores on SuperGLUE with only one exception ($L = 3, N = 16$).

Overall, for the majority of configurations, switching from original ModernBERT tokenizer to our custom tokenizer with embedding factorization leads to an improved averaged score of SuperGLUE and BLiMP benchmarks while reducing the number of parameters.

6 Conclusions

In this work, we present the first systematic study of iterative, layer-sharing architectures for language modeling under a data-constrained setting, using the 10M-token BabyLM corpus. We tested three iteration strategies: Layer Repetition, Model Repetition, and an interpretability-inspired Middle-loop Repetition; and showed that reusing a small number of trained layers through repeated computation is an effective alternative to scaling parameter count. Notably, a 3-layer iterated model was able to outperform a conventional 24-layer ModernBERT-style baseline on SuperGLUE and a 6-layer baseline on BLiMP, despite using a fraction of the trainable parameters and identical training data.

Finally, we also explored introducing per-iteration LoRA specialization and reducing parameter count through embedding factorization and a custom, smaller-vocabulary tokenizer. We showed that these techniques shrank the model parameter counts while frequently improving rather than harming the overall performance.

Acknowledgments

Co-funded by the European Union (ERC, NG-NLG, 101039303). This research was conducted using the computational infrastructure of the Poznan Supercomputing and Networking Center (PCSS) and used resources of the LINDAT/CLARIAH-CZ Research Infrastructure (Czech Ministry of Education, Youth, and Sports project No. LM2018101).

Limitations

Due to constraints in computation time, the results presented in this work were not averaged over multiple runs with different random seeds.

Ethical Considerations

We acknowledge the use of GitHub Copilot in drafting parts of our code. Some portions of the manuscript were revised using an AI-assisted grammar checker.

References

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. 2025. Relaxed recursive transformers: Effective parameter sharing with layer-wise LoRA. In *International Conference on Learning Representations*.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candance Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every layer counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 238–252, Singapore. Association for Computational Linguistics.

Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. 2025. Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28241–28259, Vienna, Austria. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Jonas Geiping, Sean Michael McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM challenge: Exploring the effect of variation sets on language model training efficiency. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 252–261, Miami, FL, USA. Association for Computational Linguistics.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 82–94, Miami, FL, USA. Association for Computational Linguistics.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*, pages 175–191, Suzhou, China. Association for Computational Linguistics.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning*. Springer.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, New Orleans, LA, USA.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Wiktor Kamzela, Mateusz Lango, and Ondrej Dusek. 2025. You are an LLM teaching a smaller model everything you know: Multi-task pretraining of language models with LLM-designed study plans. In *Proceedings of the First BabyLM Workshop*, pages 469–487, Suzhou, China. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Nalin Kumar, Mateusz Lango, and Ondrej Dusek. 2025. Pretraining language models with LoRA and artificial languages. In *Proceedings of the First BabyLM Workshop*, pages 525–530, Suzhou, China. Association for Computational Linguistics.

Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. 2025. Remarkable robustness of LLMs: Stages of inference? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Hyunji Lee, Wenhao Yu, Hongming Zhang, Kaixin Ma, Jiyeon Kim, Dong Yu, and Minjoon Seo. 2025. Understanding and enhancing mamba-transformer hybrids for memory recall and language modeling. In *Proceedings of the First BabyLM Workshop*, pages 380–398, Suzhou, China. Association for Computational Linguistics.

Matthew Theodore Roque and Dan John Velasco. 2025. Beyond repetition: Text simplification and curriculum learning for data-constrained pretraining. In *Proceedings of the First BabyLM Workshop*, pages 246–255, Suzhou, China. Association for Computational Linguistics.

David Samuel, Andrey Kutuzov, Lilja Øvrelid, and Erik Velldal. 2023. Trained on 100 million words and still in shape: BERT meets British National Corpus. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1954–1974, Dubrovnik, Croatia. Association for Computational Linguistics.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations*, volume 2025, pages 14855–14881.

Jannek Ulm, Kevin Du, and Vésteinn Snæbjarnarson. 2025. Contrastive decoding for synthetic data generation in low-resource language modeling. In *Proceedings of the First BabyLM Workshop*, pages 29–41, Suzhou, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 6000–6010, Long Beach, California, USA.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547, Vienna, Austria. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Liu Yang, Kangwook Lee, Robert D Nowak, and Dimitris Papailiopoulos. 2024. Looped transformers are better at learning learning algorithms. In *The Twelfth International Conference on Learning Representations*.

Boyi Zeng, Shixiang Song, Siyuan Huang, Yixuan Wang, He Li, Ziwei He, Xinbing Wang, Zhouhan Lin, and 1 others. 2026. PonderLM: Pretraining language models to ponder in continuous space. In *The Fourteenth International Conference on Learning Representations*.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, and 14 others. 2026. Scaling latent reasoning via looped language models. *Preprint*, arXiv:2510.25741.

A Additional results of relaxed iterative methods

The results of iterative methods with iteration specialization (via LoRA) are presented in Tab. 5.

Additional results of relaxed iterative methods with different LoRA training schemas are presented in Tab. 6.

Layers	Iter.	LoRA	Super GLUE	BLiMP
1	1	✓	**63.23**	**62.45**
		–	62.96	61.69
	2	✓	63.19	**63.62**
		–	**64.66**	63.47
	4	✓	**65.40**	**66.34**
		–	63.91	64.07
	8	✓	63.05	**66.78**
		–	**65.14**	66.58
	16	✓	56.66	**67.10**
		–	**64.13**	57.90
3	1	✓	**64.09**	66.72
		–	63.59	**66.74**
	2	✓	64.05	**67.58**
		–	**65.14**	66.60
	4	✓	64.80	**69.42**
		–	**65.75**	68.08
	8	✓	**65.04**	**69.90**
		–	62.45	69.40
	16	✓	57.58	**70.02**
		–	**64.92**	68.06
6	1	✓	**66.54**	**68.31**
		–	65.73	68.12
	2	✓	**65.65**	**69.40**
		–	65.00	68.56
	4	✓	64.28	**69.87**
		–	**65.66**	69.54
	8	✓	65.63	**71.13**
		–	**66.75**	69.57
12	1	✓	**67.06**	70.01
		–	65.54	**70.36**
	2	✓	**66.01**	71.15
		–	65.55	**71.33**
	4	✓	65.11	**71.92**
		–	**65.61**	70.15
24	1	✓	65.11	71.45
		–	**66.04**	**71.84**
	2	✓	65.70	71.56
		–	**65.72**	**72.25**

Table 5: Results of models with iteration specialization via LoRA compared with standard iterative models across different numbers of hidden layers L (Layers) and number of iterations N (Iter.).

Layers	Iter.	LoRA 0th	LoRA 8th	LoRA 9th	Decayed LR	S. GLUE	BLiMP
1	8	✓	✓	✓	0.00030	64.84	65.91
					0.00030	65.13	66.27
					–	63.05	**66.78**
					–	**65.14**	66.58
1	16	✓	✓	✓	0.00030	63.90	58.05
					0.00030	**64.20**	58.23
					–	56.66	**67.10**
					–	64.13	57.90
3	8	✓	✓	✓	0.00030	64.54	69.11
					0.00030	61.20	69.15
					–	**65.04**	**69.90**
					–	62.45	69.40
3	16	✓	✓	✓	0.00030	63.63	67.05
					0.00030	**65.28**	67.26
					–	57.58	**70.02**
					–	64.92	68.06
6	8	✓	✓	✓	0.00030	65.74	69.72
					0.00030	66.58	69.69
					–	65.63	**71.13**
					–	**66.75**	69.57
12	2	✓	✓	✓	0.00030	65.09	70.98
					0.00030	64.24	71.13
					–	**66.01**	71.15
					–	65.55	**71.33**
12	4	✓	✓	✓	0.00030	64.67	71.03
					0.00030	**66.13**	69.91
					–	65.11	**71.92**
					–	65.61	70.15
24	1	✓	✓	✓	0.00030	66.36	70.93
					0.00030	**67.06**	71.15
					–	65.11	71.45
					–	66.04	**71.84**
24	2	✓	✓	✓	0.00030	64.89	71.48
					0.00030	65.19	72.08
					–	65.70	71.56
					–	**65.72**	**72.25**

Table 6: Additional results of relaxed iterative models with different LoRA parameterizations. See Sec. 5.3 for details.

Learning from Correction: Data-Efficient Language Model Training Through Adaptive Teacher-Model Feedback

Sadri Islam[1], Tagore Rao Kosireddy[2], Kevin Li[2], Dylan Gaines[1],

[1]Department of Computer Science, Kennesaw State University, Marietta, Georgia, USA
[2]Department of Computer Science, Michigan Technological University, Houghton, Michigan, USA

Correspondence: dgaine20@kennesaw.edu

Abstract

We present our submission to the STRICT-SMALL track of the 2026 BabyLM Challenge. We study whether a developmentally-inspired curriculum, particularly an adaptive teacher-feedback curriculum based on the student's own productions, can improve language model pre-training under a 10M word budget. We trained a 58M-parameter student from scratch under five separate conditions that shared the same architecture and tokenizer, isolating different aspects of the training data and curriculum structure. We built an official baseline trained on a cleaned version of the official STRICT-SMALL corpus and also a synthetic baseline trained on teacher model generated data. Then, we developed a fixed curriculum to train a model on the same synthetic child-directed text in three developmental stages: infant, toddler, and child. Finally, we introduced an adaptive feedback curriculum in which the teachers generate additional training data by correcting the student's productions during training, which adapts the training data according to the learner's level of understanding. The fixed curriculum demonstrated superior performance on the BLiMP-Supplement, EWoK, and COMPS benchmarks, while the adaptive feedback curriculum substantially improved the Entity Tracking performance in one configuration. However, our official baseline model achieved the highest BLiMP, SuperGLUE, and overall macro-average scores, indicating that our curriculum methods did not produce the overall improvements that we had hypothesized. Our code is available at https://github.com/dcgaines/babylm-2026.

1 Introduction

Human children acquire language from an estimated two to seven million words per year, amounting to fewer than one hundred million words by early adolescence (Gilkerson et al., 2017). The learning regimes of modern language models are disproportionate by any such measure: Chinchilla was trained on 1.4 trillion tokens (Hoffmann et al., 2022), and Llama 3.1 on 15.6 trillion (Grattafiori et al., 2024). Training at this scale is economically and ecologically costly and places state-of-the-art pretraining beyond the reach of most academic groups (Bender et al., 2021). The BabyLM Challenge confronts this efficiency gap directly, incentivizing researchers to optimize pretraining under data limitations inspired by human development (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). We participate in the STRICT-SMALL track, which caps pretraining at ten million words of unique data and evaluates submissions on a shared suite of metrics spanning grammatical knowledge, world knowledge, and downstream language understanding.

Conventional pretraining remains very different than the conditions of a human learning a language. The corpus is assembled, shuffled, and consumed on a fixed timetable, independent of the state of the learner. In real life, when a child produces an incorrect utterance, caregivers or teachers respond to and reformulate it. The caregiver frequently answers with a corrected rendering of the same intended message, so that a meaningful fraction of the child's "training data" is adapted from their own productions (Chouinard and Clark, 2003; Clark, 2018).

In this work, we investigate whether a language model can benefit from a similar reformulation loop when limited training data is available. We also explore the efficacy of building a foundation for the student model through a curriculum of increasing grammatical complexity.

We conduct this comparison under a fixed 10M word budget, with the same architecture and tokenizer, so that curriculum structure is the only variable. Though the BASELINE retains the strongest grammatical knowledge, the ADAPTIVE CURRICULUM substantially improves the entity tracking

Proceedings of the Second BabyLM Workshop, pages 689–701
October 29, 2026 ©2026 Association for Computational Linguistics

and inference-oriented fine-tuning. This suggests that correction-based feedback during pretraining makes referential reasoning and downstream adaptability stronger, though it might not help to improve zero-shot grammaticality.

2 Related Work

Curriculum learning. BabyLM participants have commonly explored developmental plausibility through curriculum learning (Bengio et al., 2009). Previous submissions have scheduled the vocabulary, data, and training objective over the course of training (Diehl Martinez et al., 2023), ordered corpora according to cross-linguistic theories of acquisition (Salhan et al., 2024), and masked tokens of different part-of-speech categories in a staged sequence that follows the typical word-acquisition order of children (Lucas et al., 2024). This accumulated evidence suggests that curricula deliver gains on particular task families but seldom lift aggregate performance above well-tuned non-curriculum models (Hu et al., 2024; Charpentier et al., 2025). These schedules share a structural property that we believe limits them: they are written before training begins. Such curricula advance as scheduled, whether or not the student has mastered the current stage, and have no means of compensating by directing additional effort toward the concepts the student struggles with.

Teacher-student modeling. A second line of work trains a small student model under the guidance of one or more larger teacher models. The idea originates in knowledge distillation (Hinton et al., 2015), where a compact student is trained to match the output distribution of a larger teacher. Here, the teacher supervises the student's outputs without ever observing them. An interactive setup was also recently introduced in BabyLM (Mayer Martins et al., 2025), where a teacher model guided a small student through a storytelling task rather than transferring the teacher's output distributions.

Sample-efficient language modeling. The BabyLM Challenge has encouraged language modeling research under training data limits, where the goal is not only to scale the models and corpora but also to improve how efficiently a model learns from limited input. In this setting, Timiryasov and Tastet (2023) introduced Baby Llama, a 58M-parameter student model trained with only 10M words. Their work showed that a compact LLaMA-style student can benefit strongly from teacher guidance, especially when the training budget is too small for standard pretraining to be fully effective. Tastet and Timiryasov (2024) extended this direction with BabyLlama-2, using a larger student and ensemble-based teacher supervision to improve performance under the same motivation. In both systems, however, the feedback flows through logit distributions computed over the same static corpus where the teachers never observe output from the student.

Prior BabyLM systems show that compact student models can benefit from teacher-guided training. However, most existing approaches rely on fixed teacher outputs or training data prepared before student learning begins. Our work addresses this limitation by comparing a curriculum-free baseline, a fixed developmental curriculum, and an adaptive curriculum where the student learns from a combination of pre-generated developmental data and teacher-generated corrective examples based on its own outputs.

3 Methodology

In this work, we compare three training strategies for a student model based on the Baby Llama architecture (Timiryasov and Tastet, 2023) in the limited data setting of BabyLM 2026:

- BASELINE - Standard training with no curriculum order or teacher feedback. We train two BASELINE models: one on a cleaned version of the official corpus, and one on the same synthetic data used in the other strategies, randomly shuffled.
- FIXED CURRICULUM - A static developmental curriculum trained on synthetic, teacher-generated data. The curriculum progresses through three phases of increasingly complex text to determine whether developmental ordering alone improves the learning.
- ADAPTIVE CURRICULUM - An adaptive feedback curriculum that divides each developmental phase into a fixed foundation part and an adaptive corrective part which is generated during the training. The teacher models iteratively observe the student's productions and generate corrective examples.

The following subsections describe each condition in more detail.

3.1 Common Architecture and Training Setup

We use the official BabyLM 2026 Strict-Small dataset (Choshen et al., 2026), available on Hugging Face as `BabyLM-2026-Strict-Small`, to train our tokenizer. The raw corpus combines child-directed speech, Simple Wikipedia, OpenSubtitles, and associated sources for a total of 9,999,996 words. Since some of these sources are derived via transcription and web scraping, the corpus includes some markup and formatting artifacts that are not a part of natural language. We measured the primary artifacts in the raw corpus using a diagnostic scan; the artifact classes, their counts, and an example of each are shown in Table 1.

Count	Artifact	Example
473K	Speaker labels	`*MOT: more what?`
74.3K	Bracketed directions	`[giving sugar to Colin]`
35.5K	Subtitle dashes	`- And she did for a while.`
22.9K	Unclear speech markers	`*CHI: xxx Fraser napkin.`
16.4K	Wiki section markers	`= = = Method Man = = =`
10.5K	Annotation lines	`%add: COL`

Table 1: Artifact classes found in the raw corpus by the diagnostic scan, with counts and one example of each. Speaker labels mark who is speaking in a transcript, unclear-speech markers are standard placeholders for speech that was inaudible on the recording or intentionally left untranscribed, and annotation lines carry transcription metadata such as the addressee.

Our cleaning removed all of these artifact classes while preserving the utterances after them, collapsed repeated whitespace, and discarded lines that became empty (i.e. the line contained only the artifact). This reduced the corpus from 9,999,996 to 9,016,506 words, a 9.8% reduction. Using the cleaned corpus, we trained a byte-pair encoding (BPE) tokenizer (Sennrich et al., 2016) with a vocab size of 16K.

All conditions used the same tokenizer and student architecture so that the comparison could focus on the training strategy rather than model size or architectural differences. The student was a Llama-style decoder-only language model implemented with Hugging Face Transformers (Wolf et al., 2020) and based on the Baby Llama architecture (Timiryasov and Tastet, 2023). The model had 16 transformer layers, a hidden size of 512, an intermediate size of 1024, 8 attention heads, 8 key-value heads, and a context length of 512 tokens, with a total of 58.34M parameters. All student models were trained from random initialization using causal language modeling.

Across all conditions, we used the same general training pipeline. We tokenized the text with the shared 16K tokenizer and truncated each example to at most 128 tokens with no concatenation or packing across the sentences. We trained all models with the Hugging Face Trainer using AdamW optimization (Loshchilov and Hutter, 2019), cosine learning-rate scheduling, a weight decay of 0.1, `bfloat16` precision, and an effective batch size of 256. We used an initial learning rate of 3×10^{-4} for the BASELINE, FIXED CURRICULUM, and the first phase of the ADAPTIVE CURRICULUM. All training was performed using NVIDIA RTX 5000 Ada Generation GPUs with 32GB of memory. When possible, training was distributed across two GPUs, modifying the per-device batch size to maintain a consistent effective batch size.

3.2 Baseline

We trained one baseline model (BASELINE OFFICIAL) on the cleaned official corpus with the shared 16K tokenizer. With a corpus size of 9M words, we initially trained the model for 10 epochs. However, after observing signs of over-training, we reduced to 5 epochs for our final model, which improved performance and resulted in 45M total words of exposure. We use this baseline to measure the impact of our synthetic data. We note that the choice of 5 epochs and other hyperparameters was informed by evaluation on the final BabyLM pipeline rather than a held-out development set, which is a limitation discussed further in Section 6.

To assess the contribution of curriculum ordering, we additionally trained BASELINE SYNTHETIC using the same synthetic data described in Section 3.3, but randomly shuffled without any curriculum. Since it uses the same architecture, cleaning, tokenizer, training process, and hyperparameters as the other conditions, it allows us to measure whether our curriculum-based approaches provide benefits beyond standard shuffled training. We combined the synthetic corpus of 9,999,997 words and shuffled the sentences. We used the same BabyLlama architecture, shared 16K tokenizer, and optimization settings as in the BASELINE OFFICIAL model. We trained the model for 5 epochs, corresponding to approximately 50M words of exposure.

3.3 Fixed Curriculum

Instead of the official BabyLM corpus, the FIXED CURRICULUM condition used sentences created by a teacher model for three developmental stages: infant, toddler, and child. We defined each developmental stage by a separate instruction prompt that controlled the linguistic complexity of the teacher-generated sentences.

The *infant* stage requested simple, child-directed speech aimed at children under four years of age. It emphasized single-clause sentences, usually following a subject–verb–object structure with present-tense, everyday verbs and concrete, familiar nouns. Abstract ideas, subordinate or relative clauses, and passive constructions were excluded. Although short sentences were encouraged, the prompt explicitly stated that this stage was defined by grammatical simplicity rather than sentence length, and therefore permitted simple coordination with *and*.

The *toddler* stage targeted an age level of approximately four to seven years and allowed one or two clauses joined by simple connectives such as *and*, *but*, *because*, *when*, and *so*. It also permitted descriptive adjectives, prepositional phrases, and a moderate everyday vocabulary, while restricting the content to concrete situations that it could understand.

The *child* stage targeted children aged approximately eight to twelve years old and introduced richer linguistic structures, including subordinate and relative clauses, varied verb tenses, and discourse connectives such as *however*, *therefore*, *although*, *despite*, *while*, and *whereas*. It also included abstract ideas and reasoning. At this stage, syntactic complexity rather than sentence length defined the target level.

The exact text for the prompts used can be found in Appendix A.

For every stage, we appended a manually-selected sentence from the official corpus as an example for the teacher model. The teacher was instructed to use it only as inspiration for topics and words, without copying, translating, or preserving its original meaning. We obtained outputs from two instruction-tuned teacher models, `meta-llama/Llama-3.2-1B-Instruct` (Grattafiori et al., 2024) and `Qwen/Qwen2.5-1.5B-Instruct` (Yang et al., 2024). We merged the output from the two teachers and removed duplicates to produce a single dataset for each developmental stage.

The teachers first produced an initial corpus of approximately 12.3M words: 2.3M infant words, 5M toddler words, and 5M child words. We intentionally exceeded our 10M word budget with the initial teacher generation to create a buffer and allow for data cleaning. We cleaned this pool with the same process as the official corpus (as described in Section 3.2). In addition, we removed meta text from the instruction-tuned models, such as refusals, self-narration, and grammar commentary that did not follow the instructions for teacher-generated data. After cleaning, we subsequently downsampled the remaining corpus to the 10M word budget.

While downsampling, we used the complete cleaned generation pool for the infant phase. Then, we allocated the remaining budget equally between the toddler and child stages. The completed fixed-curriculum corpus comprised 2,301,826 infant words, 3,849,085 toddler words, and 3,849,086 child words, totaling 9,999,997 words which is a 23.0%, 38.5%, and 38.5% phase distribution for the infant, toddler and child stages. We allocated a lower proportion of our budget to the infant stage to avoid too much repetition because the simpler sentences tended to be shorter and addressed a smaller range of topics and ideas.

The student underwent sequential training across the developmental stages, completing 5 epochs in each stage before moving to the next. As with the BASELINE OFFICIAL model, we initially used 10 epochs, but found better performance with 5 epochs.

3.4 Adaptive Curriculum

We added an adaptive teacher feedback loop on top of the fixed curriculum for this approach. We divided each developmental stage into two phases, a fixed foundational phase and an adaptive feedback phase. We prepared the foundational phase before training to give the student an initial start for the current developmental stage. The adaptive feedback phase was generated throughout training by observing the student's current outputs. In each developmental stage, the student first trained on the foundational phase for 5 epochs. Then, the student went through 10 iterations of the adaptive feedback phase, each training for 5 epochs. We used the same hyperparameters as previous conditions.

Each phase (both foundational and adaptive) was trained with a fresh optimizer initialization and cosine schedule, so that it began with a clean opti-

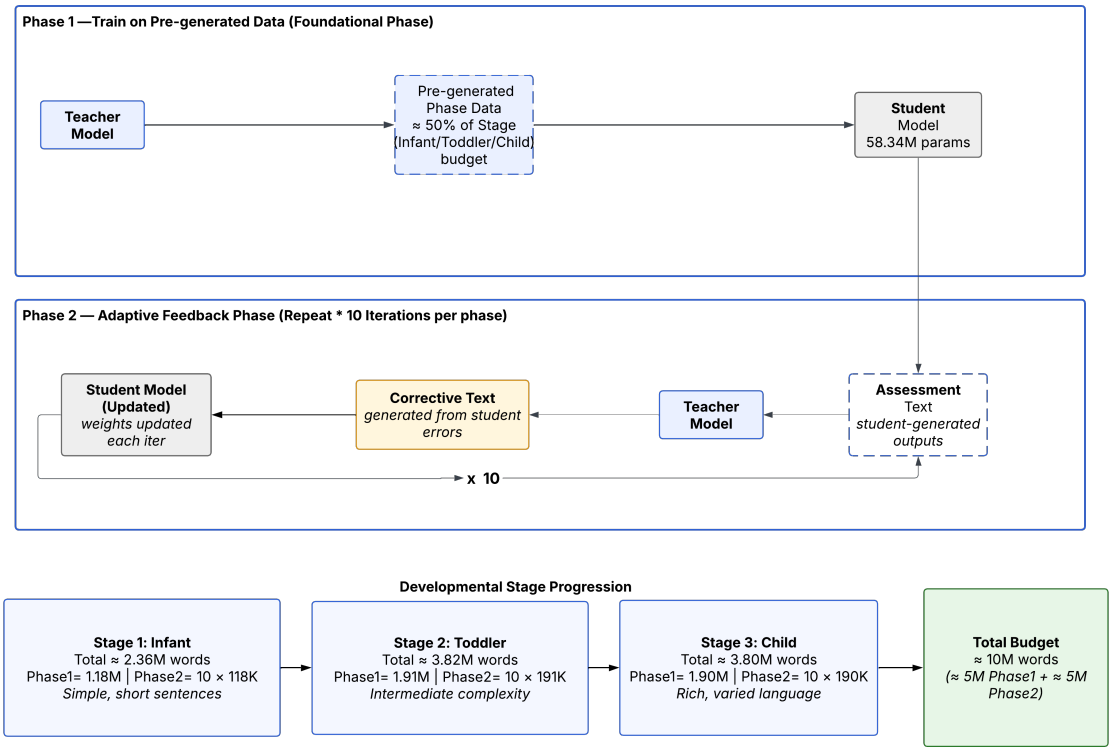


Figure 1: Overview of the adaptive feedback training process. The student is first trained on the foundational phase data. Then, it generates assessment text, which is provided to the teacher. The teacher produces corrective feedback examples, and the student is updated through repeated feedback iterations.

mization state and the model weights were carried forward. The combined budget for both phases was held to the same 10M-word budget as the FIXED CURRICULUM, so the two methods are directly comparable and differ only in how the data was constructed. We divided the budget equally between the two phases. The overall training process is shown in Figure 1.

As indicated in Section 4, this initial model did not perform as well as we had hoped. To try to allow for smaller changes during the feedback phase, we trained a variant of the ADAPTIVE CURRICULUM model that reduced the learning rate from 3×10^{-4} to 1×10^{-4} for the feedback phase (the foundation phase learning rate remained the same). We refer to this model as ADAPTIVE CURRICULUM (VARIABLE LR).

Foundational phase construction. We built the dataset for each foundational phase by randomly selecting half of the words from the same cleaned, teacher-generated corpus as the FIXED CURRICULUM model (Section 3.3). Here, the per-stage sizes were 1,180,005 infant words, 1,910,006 toddler words, and 1,900,006 child words, for a total of 4,990,017 foundational phase words.

Adaptive feedback phase construction. We performed each adaptive feedback iteration in the same manner. First, we sampled 3,000 seed lines from the pool of 452,928 lines in the cleaned official corpus between 5 and 30 words. We designed a series of prompts consisting of the token indicating the beginning of the sequence plus the first three words of each selected seed line. We used the student model to generate up to 20 new tokens from each prompt with sampling (temperature 0.9, top-p 0.95). These continuations were the student's own, often poor productions, and they constituted the assessment text of the iteration.

Then, we used the two instruction-tuned teacher models, `meta-llama/Llama-3.2-1B-Instruct` (Grattafiori et al., 2024) and `Qwen/Qwen2.5-1.5B-Instruct` (Yang et al., 2024), to generate new training data with the goal of correcting mistakes made by the student. Each teacher produced half of the word budget of the iteration. Each student sentence was shown to a teacher once at a time. We asked the teacher to produce four or five grammatically

Stage	Student Generated Sentence	Teacher Suggested Sentences
Infant	*You want I to play with toy.*	*I want you to play with my toy.* *Please let me play with your toys.* *I need you to play with my teddy bear.* *Can you please play with my ball?*
Toddler	*they gonna want to ride the merry-go-round.*	*They want to ride the merry-go-round.* *The kids like to ride the merry-go-round.* *My friend wants to ride the merry-go-round.* *They want to ride the merry-go-round because they like it.*
Child	*Kleinsmith doing a great job in the field of secrecy.*	*Kleinsmith is doing a great job in the field of secrecy.* *Kleinsmith is very good at keeping secrets.* *In the field of secrecy, Kleinsmith is doing very well.* *Kleinsmith is a great secret keeper.*

Table 2: Examples of teacher feedback in the ADAPTIVE CURRICULUM. Each row shows one student-generated sentence and four selected teacher suggestions for the corresponding developmental stage.

accurate sentences expressing the same idea at the current developmental stage, with the same stage descriptions as the FIXED CURRICULUM condition. Example teacher corrections for each developmental stage are shown in Table 2, and the exact model prompts can be found in Appendix B.

The prompts also addressed two special cases. If the student-generated sentence was incoherent, the teacher was instructed to create well-formed sentences on a similar everyday topic. If the student sentence was already grammatically correct, the teacher was instructed to create new appropriate sentences on the same idea. Teachers decoded with temperature 0.75 and top-p 0.9, with a maximum of 128, 160, and 256 new tokens at the infant, toddler, and child levels. After training, we used our two training teachers, Qwen2.5-1.5B-Instruct and Llama-3.2-1B-Instruct, to independently classify all 90,000 student sentences recorded during the ADAPTIVE CURRICULUM training to assess how often these special cases occurred. We performed this assessment using the original 3,000 sentences from each of ten feedback iterations within each of three curriculum stages. We asked each teacher to assign one of three labels: *grammatically correct*, *coherent but with errors*, or *incoherent*. Our prompt included twelve examples presented in a mixed label order and used greedy decoding (full prompt in Appendix C). Qwen classified 40.10% of sentences as grammatically correct, 1.79% as coherent but with errors, and 58.11% as incoherent. Meanwhile, Llama assigned 3.11%, 1.27%, and 95.09% to these categories respectively. The two teachers agreed to the same label rating for 57.95% of the sentences. Most of these agreements were on sentences both labeled incoherent. Of the 36,088 sentences Qwen marked as correct, Llama agreed on only 1,792 of them.

We split teacher output into one sentence per line, with common title abbreviations such as 'Mr.' and 'Dr.' protected from splitting. Each line was then run through a candidate filter. The filter stripped leading bullets, separators, and markdown emphas. Lines were removed if they matched the assistant-meta filter described in Section 3.3, had fewer than two words, or contained correction arrows, equals signs, parenthetical grammar comments, etc. A persistent key file recorded all sentences accepted in each iteration, and the teachers skipped any sentence that was already produced. This guaranteed that the adaptive feedback phase was filled with distinct sentences rather than near duplicates reproduced across the iterations.

Adaptive feedback phase budget. Each developmental stage's (infant, toddler or child) adaptive feedback phase budget mirrored its foundational phase budget size. This adaptive feedback budget was split equally across ten iterations. This resulted in about 118K words per iteration in the infant stage, 191K in the toddler stage, and 190K in the child stage. After each iteration, we trained the student on the newly generated data for 5 epochs. Then, we used the updated student to generate the samples for the next iteration. This process trained the student model on 4,989,229 total adaptive feedback phase words across the 3 developmental stages, for a grand total of 9,979,246 training words for the model.

In this study, the teacher was used only to generate text examples. Unlike standard knowledge distillation approaches which directly transfer teacher distributions (Hinton et al., 2015), the student was not trained on teacher logits, hidden states, or probability distributions. The correction signal is signif-

icantly narrow, consisting of short, targeted corrections of the student's own errors. The foundational phase provides broad lexical exposure, so the adaptive feedback phase can stay focused on correction.

Additional configurations. During our development process, we trained the ADAPTIVE CURRICULUM in a few different configurations, keeping the architecture, tokenizer, teachers, prompts, and filtering unchanged. This included a 10-epoch version — like in the other conditions, this did not work as well as the 5 epochs used in the final model. We also tried shifting the balance between the foundational phase and the adaptive feedback phase to 75/25 instead of 50/50, but this was not effective. In the final variant, we kept the 50/50 balance between the phases, but we mixed the teacher-generated foundational dataset with the cleaned official BabyLM corpus, taking 50% from the official corpus and 50% from the teacher models. This did not improve the model's performance. Again, this did not improve the model's performance.

4 Results

The BabyLM evaluation suite assesses models across multiple dimensions of linguistic competence, including grammatical knowledge (BLiMP), supplementary linguistic phenomena (BLiMP Supplement), world knowledge (EWoK), property knowledge and inheritance (COMPS), multi-step referential reasoning (Entity Tracking), and downstream language understanding (SuperGLUE finetuning).

Table 3 reports each model's performance across the main evaluation metrics. We compare the BASELINE OFFICIAL,BASELINE SYNTHETIC, FIXED CURRICULUM, ADAPTIVE CURRICULUM, and ADAPTIVE CURRICULUM (VARIABLE LR) all using the 5-epoch configuration as our primary submission.

Zero-shot grammatical knowledge (BLiMP). BLiMP (Warstadt et al., 2020) evaluates zero-shot grammatical knowledge through minimal pairs, where a model must assign higher probability to the grammatical sentence over an ungrammatical one. The BASELINE OFFICIAL achieved the highest BLiMP score of 70.31, outperforming the BASELINE SYNTHETIC (66.71), the FIXED CURRICULUM (62.62), the ADAPTIVE CURRICULUM (VARIABLE LR) (59.28), and the ADAPTIVE CURRICULUM (57.77). For the BASELINE OFFICIAL, FIXED CURRICULUM, and ADAPTIVE CURRICULUM (VARIABLE LR), this pattern holds for morphology (83.75 vs. 72.73 vs. 64.04) and syntax (65.29 vs. 57.01 vs. 53.03). However, the ADAPTIVE CURRICULUM (VARIABLE LR) achieved a higher semantics score (64.94) than the BASELINE OFFICIAL (62.60) and the FIXED CURRICULUM (59.00), despite its lower overall BLiMP score. The curriculum models therefore did not outperform either baseline on BLiMP under these conditions, and both adaptive variants scored below the FIXED CURRICULUM on BLiMP. We conjecture that training on teacher-generated synthetic text, which may differ in vocabulary and sentence structure from the natural BabyLM corpus, may have reduced the model's exposure to the statistical patterns that BLiMP probes. BLiMP Supplement scores are similar across conditions, with the FIXED CURRICULUM narrowly leading (56.74), followed by the ADAPTIVE CURRICULUM (VARIABLE LR) (54.41), the BASELINE OFFICIAL (54.35), the BASELINE SYNTHETIC (53.28), and the ADAPTIVE CURRICULUM (52.93).

World knowledge (EWoK) and property knowledge (COMPS). EWoK (Ivanova et al., 2024) tests whether models have acquired real-world knowledge about physical and social domains through contextually appropriate minimal pairs. COMPS (Misra et al., 2023) evaluates property knowledge and whether models can infer that properties of superordinate concepts are inherited by subordinate concepts. Performance on EWoK and COMPS was similar across the five conditions, with scores differing by less than three points on each benchmark. The BASELINE SYNTHETIC achieved the highest scores on both EWoK (51.66) and COMPS (52.59), although its margins over the other conditions were small.

Entity tracking. Entity Tracking (Kim and Schuster, 2023) tests a model's ability to follow referents across multiple operations and clause boundaries. The largest variation across the five conditions occurred on Entity Tracking, where the ADAPTIVE CURRICULUM (VARIABLE LR) achieved the highest score of 36.07, followed by the FIXED CURRICULUM (28.50), BASELINE OFFICIAL (26.70), ADAPTIVE CURRICULUM (17.66), and BASELINE SYNTHETIC (17.32). The best-performing variant also exceeded the challenge organizers' GPT-2 Strict-Small reference score of 23.58. Although the fixed curriculum outperforms

Model	BLiMP	BLiMP Supp.	EWoK	COMPS	Entity Tracking	S.GLUE	Macroavg
BASELINE OFFICIAL	**70.31**	54.35	51.09	51.57	26.70	**63.71**	**59.86**
BASELINE SYNTHETIC	66.71	53.28	**51.66**	**52.59**	17.32	62.52	58.54
FIXED CURRICULUM	62.62	**56.74**	51.16	51.75	28.50	61.82	58.08
ADAPTIVE CURRICULUM	57.77	52.93	50.06	50.93	17.66	61.44	55.55
ADAPT. CURR. (VAR. LR)	59.28	54.41	49.53	49.83	**36.07**	62.54	56.44

Table 3: Evaluation results. Bold indicates best performance per column across our five models.

Model	BoolQ	MultiRC	RTE	WSC	MRPC	QQP	MNLI
BASELINE OFFICIAL	69.48	62.79	**57.55**	**65.38**	**71.57**	**72.55**	**46.62**
BASELINE SYNTHETIC	69.05	62.91	53.24	**65.38**	**71.57**	69.13	46.37
FIXED CURRICULUM	68.56	**63.78**	51.80	61.54	70.59	71.40	45.05
ADAPTIVE CURRICULUM	69.11	61.10	53.24	61.54	70.10	71.12	43.91
ADAPTIVE CURRICULUM (VARIABLE LR)	**69.54**	62.71	54.68	63.46	70.59	72.06	44.72

Table 4: Language understanding (SuperGLUE) sub-task results (accuracy). Bold indicates best performance per column across our five models.

both baselines, the two adaptive variants showed markedly different results. This contrast suggests that the effectiveness of adaptive training on this task may depend on the learning-rate schedule, though an isolated experiment with repeated measures would be required to confirm. The observed gains in the ADAPTIVE CURRICULUM (VARIABLE LR) model's Entity Tracking cannot be attributed to the corrective feedback loop alone, and further controlled comparisons would be needed to establish the direct impact of developmental ordering and corrective feedback.

Language Understanding (SuperGLUE). The SuperGLUE benchmark represents a composite of difficult language understanding tasks that each model is finetuned and then evaluated on (Wang et al., 2020). The breakdown of each model's performance on these tasks is shown in Table 4. The ADAPTIVE CURRICULUM (VARIABLE LR) achieved the highest BoolQ accuracy (69.54) and scored higher than the ADAPTIVE CURRICULUM on all seven tasks. It also performed competitively on QQP (72.06), although the BASELINE OFFICIAL led with 72.55. The BASELINE OFFICIAL also achieved the highest scores on RTE (57.55) and MNLI (46.62), and shared the highest WSC (65.38) and MRPC (71.57) scores with the BASELINE SYNTHETIC. The FIXED CURRICULUM achieved the highest MultiRC score (63.78). Overall, we found that no single condition led across all tasks, but the BASELINE OFFICIAL achieved the highest aggregate SuperGLUE score (63.71).

5 Discussion

The results reveal different strengths across the five conditions. The BASELINE OFFICIAL achieves the highest BLiMP and aggregate SuperGLUE scores, while the ADAPTIVE CURRICULUM (VARIABLE LR) achieves the best entity tracking despite lower BLiMP accuracy. The BASELINE SYNTHETIC performs best on EWoK and COMPS, whereas the FIXED CURRICULUM leads on BLiMP Supplement and MultiRC. A possible contributor to the lower BLiMP scores is the shift from natural to synthetic training text. Teacher-generated text may reflect the vocabulary and syntactic preferences of `meta-llama/Llama-3.2-1B-Instruct` (Grattafiori et al., 2024) and `Qwen/Qwen2.5-1.5B-Instruct` (Yang et al., 2024). However, the synthetic baseline also outperforms all three curriculum conditions on BLiMP, suggesting that the use of synthetic text alone does not explain the full performance gap.

The ADAPTIVE CURRICULUM (VARIABLE LR) achieves an entity tracking score of 36.07, exceeding the challenge organizers' GPT-2 Strict-Small reference score (23.58) and approaching the corresponding base teacher models evaluated on the same benchmark (`Llama-3.2-1B`: 38.55, `Qwen2.5-1.5B`: 39.87), despite training on much less data. However, the standard ADAPTIVE CURRICULUM scores much lower on this task, suggesting that the gain may also depend on the learning rate schedule. The SuperGLUE results show a more limited advantage that the variable learning rate variant leads on BoolQ (69.54), narrowly above the BASELINE OFFICIAL (69.48), while the

latter outperforms both adaptive variants on MNLI and MRPC and retains the highest aggregate score. Together, these results suggest that the benefits of adaptive training may depend on the tasks and learning rate scheduling, with an advantage appearing on entity tracking in one variant.

On the overall macroaverage, the curriculum learning models failed to surpass either baseline model. As we discuss in Section 6, the curriculum used in this work was not strongly grounded in linguistic or child development literature, which may have led to this shortcoming. Additionally, the ADAPTIVE CURRICULUM models performed worse than the FIXED CURRICULUM overall, suggesting the feedback phase wasn't as beneficial as we had hoped. The post-hoc ratings of the student-generated sentences reported in Section 3.4 showed that the teacher models frequently rated the sentences as "incoherent" (over 95% for Llama). A few cherry-picked examples include "I can't believe how much I'm looking forward to going on an exciting adventure!" and "Well, you talked about it yesterday and now he's excited.", which are grammatically correct except for a missing comma in the second sentence. This suggests that the teacher models may not have been able to provide quality corrections during the feedback phase, limiting its effectiveness.

6 Conclusions

We investigated five pretraining strategies for a 58M-parameter student under a 10M word training budget: an official baseline, a synthetic baseline, a fixed developmental curriculum, an adaptive feedback curriculum, and the adaptive curriculum with a variable learning rate. In the adaptive curriculum, teacher models generate, corrective training data in response to the student's own productions. The baseline retained the strongest grammatical knowledge as measured by BLiMP, while the adaptive curriculum with the variable LR achieved substantially stronger entity tracking, nearly matching the teacher models on that benchmark despite being trained on far less data. These findings suggest that with proper scheduling correction-based feedback during pretraining may strengthen reasoning and referential tracking, even when it does not improve overall grammatical accuracy, and may be most valuable in settings where downstream adaptability matters more than zero-shot grammatical judgment.

While our curriculum models' results showed promise in some specific task areas, the overall evaluations failed to match the curriculum-free baseline models. This shows that our approach was not as effective as we hypothesized. To improve our method in future work, we could isolate the effect of the curriculum from the overall dataset by using large instruction models to order the curated official dataset by increasing complexity. We could also leverage the use of two teacher models by having them take distinct, specialized roles; student productions could be routed to the teacher best suited to correcting a specific error type. Finally, we could explore a wider range of adaptive phase hyperparameters to better balance grammaticality and reasoning performance, or alternate between the foundational and feedback phases within each developmental stage.

Our five models and their intermediate checkpoints are publicly available on the Hugging Face Hub: the OFFICIAL BASELINE,[1] the FIXED CURRICULUM,[2] the ADAPTIVE CURRICULUM VARIABLE LR,[3] the ADAPTIVE CURRICULUM with a learning rate of 3×10^{-4},[4] and the SYNTHETIC BASELINE.[5]

Limitations

This work had a few limitations that should be noted. The hyperparameters used in training were chosen based on prior experience and observations made during the experiments, but they were not rigorously tuned. This includes the learning rate, number of epochs, and batch size, as well as the number of feedback iterations and the split between the foundational and adaptive phases in FEEDBACK CURRICULUM. It is possible that a more thorough grid search over hyperparameters may have resulted in improved model performance.

Additionally, the ADAPTIVE CURRICULUM model used the same teacher models to generate the adaptive feedback as well as to generate the foundational data. This may have resulted in the synthetic data from both phases reflecting teacher-

[1] https://huggingface.co/ksu-help/babylm2026-strict-small-baseline-ep5

[2] https://huggingface.co/ksu-help/babylm2026-strict-small-fixed-ep5

[3] https://huggingface.co/ksu-help/babylm2026-strict-small-adaptive-synth-ep5

[4] https://huggingface.co/ksu-help/babylm2026-strict-small-adaptive-lr3e-4-ep5

[5] https://huggingface.co/ksu-help/babylm2026-strict-small-synthetic-baseline-ep5

specific shortcomings or stylistic biases. It may have been beneficial to use a separate teacher model for each purpose, to allow one to correct any mistakes learned from the other.

Finally, we defined our three developmental stages through roughly equal age divisions and progressions of grammar rules, without strong grounding in prior linguistic or child development literature. The choice of three discrete stages was practical rather than theoretically motivated; a continuous or finer-grained complexity ordering might better reflect how children actually acquire language. Additionally, even young children are exposed to adult-to-adult speech that is considerably more complex than our infant-stage prompts, so removing complex constructions entirely may not accurately reflect naturalistic input. Future work could ground stage boundaries in established language acquisition milestones (Gilkerson et al., 2017), gradually increase the proportion of complex utterances rather than using hard stage transitions, or compare discrete versus continuous complexity schedules directly.

Acknowledgements

The authors generated the ideas, workflow, experimental design, and claims of this work. AI tools were used as follows: OpenAI and Anthropic assisted with code debugging and grammar correction in the manuscript. Grammarly was also used to increase the readability of the paper. All suggestions were reviewed, verified, and edited if necessary by the authors before inclusion and no confidential or personally identifiable data were used.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, Suzhou, China. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Michelle M. Chouinard and Eve V. Clark. 2003. Adult reformulations of child errors as negative evidence. *Journal of Child Language*, 30(3):637–669.

Eve V. Clark. 2018. Conversation and language acquisition: A pragmatic approach. *Language Learning and Development*, 14(3):170–185.

Richard Diehl Martinez, Zébulon Goriely, Hope McGovern, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2023. CLIMB: Curriculum learning for infant-inspired model building. *arXiv preprint arXiv:2311.08886*.

Jill Gilkerson, Jeffrey A. Richards, Steven F. Warren, Judith K. Montgomery, Charles R. Greenwood, D. Kimbrough Oller, John H. L. Hansen, and Terrance D. Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American Journal of Speech-Language Pathology*, 26(2):248–265.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna Ivanova, Carina Kauf, Greta Tuckute, Nancy Kanwisher, Joshua B Tenenbaum, Evelina Fedorenko,

and Jacob Andreas. 2024. EWoK: Elements of world knowledge for evaluation of language model world modeling. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 3835–3855.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Evan Lucas, Dylan Gaines, Tagore Rao Kosireddy, Kevin Li, and Timothy C. Havens. 2024. Using curriculum masking based on child language development to train a large language model with limited training data. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 221–228, Miami, FL, USA. Association for Computational Linguistics.

Jonas Mayer Martins, Ali Hamza Bashir, Muhammad Rehan Khalid, and Lisa Beinborn. 2025. Once upon a time: Interactive learning for storytelling with small language models. In *Proceedings of the First BabyLM Workshop*, Suzhou, China. Association for Computational Linguistics.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2920–2938.

Suchir Salhan, Richard Diehl Martinez, Zébulon Goriely, and Paula Buttery. 2024. Less is more: Pre-training cross-lingual small-scale language models with cognitively-plausible curriculum learning strategies. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, Miami, FL, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Jean-Loup Tastet and Inar Timiryasov. 2024. BabyLlama-2: Ensemble-distilled models consistently outperform teachers with limited data. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, Miami, FL, USA. Association for Computational Linguistics.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby Llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 279–289, Singapore. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2020. Superglue: A stickier benchmark for general-purpose language understanding systems. *Preprint*, arXiv:1905.00537.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

A Fixed Curriculum Prompts

Infant.

```
Generate language for very young children (under about 4 years old), in the style of simple child-directed speech.

Generate several grammatically simple sentences a young child would naturally say or hear.

Style of this level:
- Single-clause sentences, usually subject-verb-object.
- Present tense and basic everyday verbs (eat, run, see, play, sleep, want, have, look, go, like).
- Concrete, familiar nouns. No abstract ideas, no subordinate or relative
```

clauses, no passive voice.
- Sentences may be short or slightly longer (simple "and" coordination is fine). What defines this level is grammatical simplicity, NOT the number of words.

Examples of the target style (note the varied lengths):
- I see you.
- The cat sees the bird.
- Mama gives me milk.
- The dog runs fast.
- I want my ball and my cup.

Use the source passage below only as inspiration for topic and words. Do not copy it, translate it, or try to preserve its meaning.
Output only the new sentences, one per line, no numbering and no bullets.

Source passage:

Toddler.

Generate language for children aged about 4 to 7 years old.

Generate several sentences at this developmental level.

Style of this level:
- Mostly one or two clauses, joined with simple connectives (and, but, because, when, so).
- Descriptive adjectives, prepositional phrases, and a moderate everyday vocabulary.
- Concrete situations a young child understands. Avoid technical or abstract vocabulary.
- Sentences may be short or longer. A short sentence can still show this level (e.g. a simple "because" clause), and a longer one can stay simple. Grammatical style, not length, defines the level.

Examples of the target style (note the varied lengths):
- She cried because she fell.
- The big yellow bus stopped at the corner.
- My little brother is sad because his balloon flew into the tree.
- When it rained, the children ran inside to stay dry.
- He wanted to play outside, but he had to finish his lunch first.

Use the source passage below only as inspiration for topic and words. Do not copy it, translate it, or try to preserve its meaning.
Output only the new sentences, one per line, no numbering and no bullets.

Source passage:

Child.

Generate richer language for children aged about 8 to 12 years old.

Generate several sentences at this developmental level.

Style of this level:
- Subordinate clauses, relative clauses, and varied tenses.
- Discourse connectives (however, therefore, although, despite, while, whereas).
- Abstract ideas and reasoning are allowed.
- Sentences may be short or long. A short sentence can still be syntactically complex (e.g. "He left before she could answer"), and complexity, not length, is what defines this level.

Examples of the target style (note the varied lengths):
- He left before she could answer.
- Despite the long flight, the team arrived in good spirits.
- The library, which had been closed for renovations, reopened with new study spaces.
- Although the experiment had failed twice, the students believed small changes would finally produce clearer results.
- She wondered whether the rumor about the old house was actually true.

Use the source passage below only as inspiration for topic and words. Do not copy it, translate it, or try to preserve its meaning.
Output only the new sentences, one per line, no numbering and no bullets.

Source passage:

B Adaptive Curriculum Prompts

Infant.

A small language model in training produced the sentence below while learning to talk. It may contain grammar mistakes.
Generate 4 to 5 correct sentences that express the same idea properly, in the style of speech to a child under about 4 years old: one clause, present tense, concrete everyday words, no subordinate or relative clauses.
If the sentence is already correct, generate 4 to 5 simple sentences on the same idea at this level.
If it cannot be understood, generate 4 to 5 simple well-formed sentences at this level on a similar everyday topic.
Write each sentence on its own line.

Toddler.

A small language model in training produced the sentence below while learning to talk. It may contain grammar mistakes.
Generate 4 to 5 correct sentences that express the same idea properly, at the level of a child aged about 4 to 7: one or two clauses joined by simple connectives (and, but, because, when, so), everyday vocabulary, concrete situations.

If the sentence is already correct, generate 4 to 5 sentences on the same idea at this level.
If it cannot be understood, generate 4 to 5 well-formed sentences at this level on a similar everyday topic.
Write each sentence on its own line.

Child.

A small language model in training produced the sentence below while learning to write. It may contain grammar mistakes.
Generate 4 to 5 correct sentences that express the same idea properly, at the level of a child aged about 8 to 12: they may use subordinate or relative clauses, varied tenses, and discourse connectives (however, although, because, while), and may express abstract ideas.
If the sentence is already correct, generate 4 to 5 sentences on the same idea at this level.
If it cannot be understood, generate 4 to 5 well-formed sentences at this level on a similar topic.
Write each sentence on its own line.

C Student Sentence Rating Prompt

Decide which of the three categories fits the sentence best: Grammatically correct, Incoherent, or Coherent but with errors.

Examples: "the the baby was very happy." -> Coherent but with errors "They got us a new game." -> Grammatically correct "the when park going that is" -> Incoherent "what are you waiting for?" -> Grammatically correct "it was a on the day." -> Incoherent "He go to school." -> Coherent but with errors "The Great War was parked outside." -> Incoherent "And I'd have trouble remembering what I had planned." -> Grammatically correct "where are the book." -> Coherent but with errors "she gave you the name." -> Grammatically correct "was I alive is my favorite food." -> Incoherent "Well,how fast were with you?" -> Coherent but with errors

Answer with only one of these three: Grammatically correct, Incoherent, or Coherent but with errors. No explanation needed

Sentence: sentence

Miss Universal Grammar: Pretraining BabyLMs with Linguistic and Non-Linguistic Sources

Erika Varis Doggett*
Disney Research Imagineering
Walt Disney Imagineering
erika.varis.doggett@disney.com

Mary Katie Kennedy*
University of Southern California
mkkenned@usc.edu

Abstract

Inspired by linguistic theory and recent work exploring linguistic biasing of neural networks, we propose a "Universal Grammar" (UG) pretraining paradigm for transformer-based neural networks, preparing the network to learn language more effectively, especially in low data volume scenarios. We accomplish this "universal" pretraining with a combination of synthetic language-like sequences created with a context-free grammar, biological sequence sources like protein amino acids, and material sources like crystal structures. Experiments with the BabyLM Challenge datasets show improved language acquisition in English, Dutch, and Chinese with our `MissUniverse` models, compared to not using "UG" pretraining. This pretraining paradigm has interesting implications for both the future of language models, and potentially how we think about human Universal Grammar.

1 Introduction

Children demonstrate a reasonable mastery over their native language around age five (Gleitman and Newport, 1995), a skill that continues to expand into a proficiency over complex grammatical constructions, a wider vocabulary, and even an awareness of what language to use in which social contexts for specific conversational goals (Clark, 2024). All of this is achieved despite children being exposed to no more than 50M words by age five (Warstadt and Bowman, 2024; Frank, 2023), and less than 100M words by age 13 (Gilkerson et al., 2017). In contrast, large language models (LLMs) often need to train on orders of magnitude more data to even broach human proficiency. To further highlight this divide between the language-acquisition efficiency of humans and LLMs, bilingual children—those who acquire two languages from an early age as their two first languages—have an overall reduced input of both languages due to their inputs being split between the two languages; despite this, research has found that bilingual children utilize vocabulary diversity and syntactic complexity as rich as their monolingual counterparts (Nicoladis and Jiang, 2018; Korade et al., 2024).

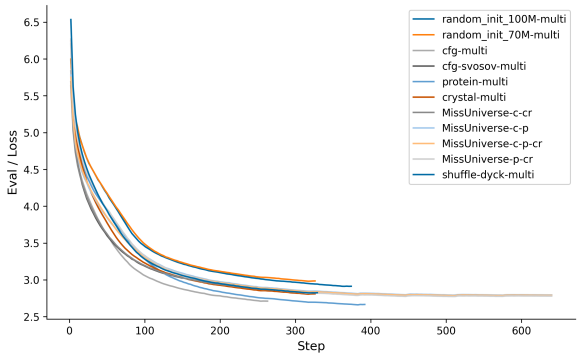


Figure 1: Eval loss curves comparing different data sources to induce a "Universal Grammar" in stage 1 pretraining, for multilingual models.[1]

Observation of children's remarkable language-learning ability led to the notion of the Poverty of the Stimulus (PotS) (Chomsky, 1965, 1981b; Lightfoot, 1989; Crain, 1991). In brief, PotS raises the paradoxical problem that the inputs children receive are not sufficient enough to specify the correct grammar that children ultimately converge upon, rather than the numerous other possible grammatical explanations for the language inputs. A solution to this problem has come to be known as "Universal Grammar" (UG), which has largely been defined variously over the years by Noam Chomsky as the biological grammar possessed by all humans (Roberts, 2017). While this theory has received its criticisms (Evans and Levinson, 2009; Tomasello, 2009; Ambridge and Lieven, 2014), we take its core notions as inspiration for our BabyLM.

*Both authors contributed equally to this work.

[1]Only the competition `MissUniverse` models were run for the full 10 epochs; all others underwent early stopping when loss plateaued.

Proceedings of the Second BabyLM Workshop, pages 702–721
October 29, 2026 ©2026 Association for Computational Linguistics

Building on the tradition of inducing syntactic biases in LLMs (Gessler and Schneider, 2023; Mueller and Linzen, 2023; Hu et al., 2025), we aim to induce an analogous "UG" state for an artificial neural network, inspired by hierarchical fundamentals and Merge operations proposed in human theoretical UG (Chomsky, 1995). To achieve this, we extend the "pre-pretraining" methodology of Hu et al. (2025) and pretrain on data that demonstrates hierarchical structures and geometric folding. Our work diverges from the human-centric limitation of traditional conceptualizations of UG in that our data takes inspiration not only from human language, but also from biological and physical "languages" and their associated "syntax." Recent work has found LLMs develop non-trivial, deep representations of human language through training on unlabeled text (Hewitt and Manning, 2019; Kulmizev et al., 2020; Li et al., 2021a; Liu et al., 2026); the same mechanisms that have enabled its success in human language have also been shown to be applicable in biological processes such as proteins (Rao et al., 2021; Jumper et al., 2021; Lin et al., 2023b) as well as physical materials such as crystal structures (Antunes et al., 2024; Zimmermann et al., 2025; Zhang et al., 2026).

By pretraining on data sources that demonstrate hierarchical syntax, we aim to induce in our model its own "UG", one that is not limited only to human language, but which nevertheless demonstrates structural properties that are analogous to the basic principles of syntax in human language, such as Internal and External Merge (Chomsky, 1995). By focusing on data that demonstrates hierarchical structures wherein long-distance elements are folded into a close proximity or in which certain elements combine only with other elements, we seek to induce the basic fundamentals of syntax as the models' own UG that might enable it to better acquire a variety of human languages.

While our artificial "UG" proposal here is inspired by the theory of human UG, it should by no means be taken as an endorsement of that theory for either humans or ANNs. We merely draw the analogy between a theoretical framework that has been proposed for humans, and a training paradigm for transformer-based ANNs: if human languages all share the same basic linguistic principles, can we prime a neural network to be quicker at acquiring such principles before exposure to natural language ever occurs?

2 Background

2.1 Linguistic Principles

In the field of linguistics, a well-known framework of syntax is generative Minimalist Syntax, which has developed out of the foundational theories of Noam Chomsky (1957; 1981a; 1986; 1965). This constituency-based family of grammatical theories is built on the binary-branching X-bar theory and its core operations of *External Merge* and *Internal Merge* (also known as "Move"). These Merge operations serve to connect elements; External Merge involves the formation of phrases when heads take their arguments and specifiers (e.g., the verb "eat" taking "apples" as its argument to form the verb phrase "eat apples"), while Internal Merge involves the movement of formed phrases to new landing spots in the syntax (e.g., "What did you see them eat ~~what~~?"). After all operations are applied in the course of derivation, the sentence is then linearized when read from left-to-right along the leaf nodes.

2.2 LLMs are capable of discovering syntax

The interpretability field of LLMs has amassed a number of converging studies that indicate models' ability to discover syntax, such as the work of Hewitt and Manning (2019) who found that dependency structures are recoverable from the contextualized representations of sentences while other work, such as Liu et al. (2026), find that LLMs display behaviors reflecting tree-like representations of constituency.

Within the space of multilingual models, several studies have converged upon evidence that multilingual models share an abstract, conceptual representation space; Starace et al. (2023) found linguistic information, such as part-of-speech, to have shared representations across languages, which Brinkmann et al. (2025) further expanded by finding morphosyntactic concepts such as grammatical number, gender, and tense to be shared across languages. Models furthermore have been discovered to display similar behavior in their representations across various layers. Several researchers have found early layers constitute an input space that aligns to either the input language or the semantics; middle layers focus on task-solving and convert the representations into a shared abstract concept space, which often is more aligned to English representations; and late layers finally convert the concepts back into the output language (Zhao et al., 2024; Wendler et al., 2024; Tamo et al., 2026).

Models' ability to converge upon shared, universal grammatical features and to furthermore share abstract representations are indicative of models' abilities to not only extract syntax, but to create their own iteration of a Universal Grammar-like functionality if trained upon enough data.

2.3 Syntactic Induction in LLMs

Previous research in LLMs has sought to induce syntactic biases within models during pretraining (Zhou et al., 2020; Xu et al., 2021; Bai et al., 2021; Li et al., 2021b; Zhang et al., 2022; Gessler and Schneider, 2023; Yang et al., 2026). Given that models are able to extract core grammatical concepts across languages if provided enough unlabeled training data, we seek to discover whether inducing the basic fundamental operations of syntax within models can provide the Universal Grammar tools, and reduce the necessary monolingual and multilingual training data while achieving improved performance. Our work in particular builds off the research of Hu et al. (2025), which found that a stage of pretraining on the context-sensitive, hierarchical formal language of k-Shuffle Dyck conferred an inductive bias that reduced the necessary token budget by 33%.

Using the methodology of Hu et al. (2025), we seek to pretrain our models on a variety of different training data sources that demonstrate hierarchical structures, geometric folding akin to Internal Merge, and mildly context-sensitive grammars in an attempt to induce in our models a UG similar to what is postulated to exist in humans.

3 Data

The inspiration for our data comes from linguistic and non-linguistic grammars, similar to Papadimitriou and Jurafsky (2023) who pretrained on music. Hu et al. (2025) found that among their set of synthetic formal languages, the best data for pre-pretraining was the context-sensitive k-Shuffle Dyck language. This parallels human language, which generally falls within the mildly context-sensitive tier of grammar (Joshi et al., 1990).

However, syntax is found not only in human language, but in nature as well. In this section, we describe the three datasets we use for pretraining to induce a syntactic bias within our models.

3.1 Proteins

Proteins are three-dimensional structures that are the result of a linearized amino acid chain, which will always[2] fold in a *specific* configuration depending upon the chemical bonds that form between the atoms (Anfinsen, 1973). However, searching the space of all possible folding configurations would take an exorbitant time; despite this, proteins are able to fold into their specific shape in mere milliseconds or less (Dill et al., 2007), and this near instantaneous folding of a seemingly innumerable space of possibilities has come to be known as the Levinthal Paradox (Levinthal, 1968).

Early research identified computational linguistics as a means to solve this problem. Dill et al. (2007) developed their Zipping and Assembly Mechanism by Dynamic Programming (ZAMDP), inspired by the computational linguistic methods of CKY algorithm and chart parsing, to solve local problems in order to achieve a global structure for protein and RNA secondary structures. They further observe that basic biopolymer structures, such as β hairpins in protein structures or the stem loops of secondary RNA structures, demonstrate simplistic context-free grammars with recursively nested links. More complex structures, such as protein configurations like helix bundles and β sheets along with RNA secondary pseudoknots, demonstrate crossing dependencies like that observed in Dutch. In this regard, proteins are mildly context-sensitive, similar to human language.

Recently, the advent of LLMs has allowed the field to advance in leaps and bounds, with AlphaFold v2 winning the 2024 Nobel Prize for Chemistry for solving the Levinthal Paradox. Countless scores of LLMs have demonstrated remarkable potential for models to assist with protein structures (Jumper et al., 2021; Baek et al., 2021; Rao et al., 2021; Vig et al., 2021; Wang et al., 2022; Lin et al., 2023b; Fang et al., 2023). Even more notably, the protein language model ESM2 (Lin et al., 2023b) masks out 15% of amino acids in a protein's amino acid chain, and trains the model to predict the missing acid, which requires the model to create complex, contextualized representations of the protein, similar to how LLMs develop rich representations of sentences(Brinkmann et al., 2025; Hewitt and Manning, 2019; Nikolaev and Padó, 2023; Kamath et al., 2024). These well-contextualized representations of the linearized chain of amino acids provide the necessary information for a secondary model, ESMFold, to efficiently predict three dimen-

[2]Prions are an exception to this as they are misfolded proteins that induce misfolding in adjacent proteins. For discussion, see Ezzat and Espay (2023).

sional protein structures.

Dataset Because ESM2 and ESMFold (Lin et al., 2023b) demonstrate that linearized amino acid chains of proteins encodes the information necessary to predict complex protein folding, we use the Huggingface protein dataset released by the ESM2 team (Lin et al., 2023a). This protein dataset, built off of the UniRef dataset (Suzek et al., 2015), is composed of clustered sequences according to length and similarity, increasing variation while limiting redundancy. For training budgets, we subsampled the data to 1M sequences or less and filtered to sequences that could fit within a 1024 token context window, resulting in a total of 966,428 sequences.

3.2 Crystal Structure

The biomedical field is not the only area to leverage LLMs' remarkable abilities for syntax discovery. The materials science field has also demonstrated LLMs' potential within the realm of crystals and chemistry (Jablonka et al., 2023; Zimmermann et al., 2025; Khastagir et al., 2025; Zhang et al., 2026). Among this research includes the exploration of LLMs as a means to identify new, three dimensional crystalline materials—structures which are composed of a repeating minimal unit of atoms in a three dimensional space—with various chemical properties for use in batteries, semiconductors, solar cells, and more (Khastagir et al., 2025). Unlike many protein models, many crystal generation models utilize an autoregressive (AR) task to generate plausible novel crystalline structures (Antunes et al., 2024; Khastagir et al., 2025).

Because crystals represent three dimensional structures that are formed according to a chemical syntax with possibly infinitely repeating forms, reminiscent of human language's ability for recursion, we select crystals as another dataset for pretraining to inject syntax.

Dataset Antunes et al. (2024) trained their crystal generative model CrystaLLM model on Crystallographic Information Files (CIFs), which contains structural information about the material, including a tabular list of atom's element and their xyz coordinates. We use the dataset from CrystaLLM,[3] which is composed of data from the Materials Project (Jain et al., 2013), OQMD (James E. Saal, 2013), and NOMAD (Scheidgen et al., 2023). We used the linearized list of atoms, and a standard BPE tokenizer trained on a subsample of the data.[4]

3.3 Synthetic Language

While Hu et al. (2025) used formal languages for pretraining, we were inspired by simple context-free grammars (CFGs) for creating language-like sequences, and added high-level distributional patterns to mimic the mild context-sensitive patterns found in natural language, like vocabulary frequency distributions, morphological patterns, and semantic clustering. We outline the pertinent features in the following sub-sections. Using the full system, we generate 1,000,000 sequences with a mean length of 20.5 words (ranging from 7 to 26 words each).

Grammar rules The context-free grammar rules are SVO (Subject-Verb-Object) with head-initial tree branching, and includes clause, inflection, and tense syntactic phrase options; preposition phrases; adverbs; adjectives; determiner+noun phrases; and sentential-level conjunctions. The tree depth is limited to 12 levels, for compute purposes. We also did some exploration into including an alternate SOV word order, outlined in section 5.2.

Zipfian distribution The base vocabulary size is set to 1,000 nonce words with English phonotactics. About 60% of the words were monosyllabic, and 40% bisyllabic.

The words were assigned to the different syntactic categories, and Zipfian frequencies assigned to them, with boosts depending on how often the CFG rules use the category (see Table 5 in the Appendix). Thus, we create a plausible distribution of higher and lower frequency words within the sequences.

Morphological connections Within our base lexical vocabulary (adjectives, adverbs, nouns, and verbs) we took 30% of each category to transform with morphological affixes (again, English plausible phonotactically). The morphological items received a frequency assignment of 10% of the target

[3] https://github.com/lantunes/CrystaLLM#obtaining-the-training-data

[4] Early tests with the xyz coordinates showed poor performance, so we omitted the coordinates. However, we recognize that CrystaLLM and other previous work used customized tokenizers with the xyz coordinate information. Future work can explore whether coordinates and other spatial information can be beneficially worked into our UG pretraining through additional preprocessing, tokenization, or other such measures.

frequency for the lexical category, so the morphological rule is present but does not overwhelm the vocabulary. This expansion added about 500 words to the base vocabulary.

Agglutination While languages like English and Chinese largely rely on strict word order and minimal morphological affixation, other languages are highly agglutinative. To mimic agglutination, we introduced a post-lexical fusion effect.

We create a small set of templates for items that can be fused together, like verb-object and determiner-noun, with a corresponding rate of application (e.g., 15%). For example, words like "the" "cat" might be concatenated with an optional linking vowel such as "o", resulting in the newly fused word "thecat" or "theocat".

Semantic clustering To artificially create semantic clustering effects, we assign 4–5 arbitrary semantic fields to our lexical categories. An example of semantic fields for adjectives would be physical (e.g., size/color), evaluative (e.g., good/bad), emotional (e.g., sad/happy), temporal, and quantitative (e.g., amount).

We assign a compatibility matrix to the different semantic fields to identify which ones "go together" (e.g., object nouns with physical adjectives), and assign probability weights of high (0.9), medium (0.7–0.8), or default (0.3) for the different semantic field combinations.

At sequence generation, when a word is chosen its semantic field becomes an active context, and the frequency for the next word is multiplied by the semantic weights of its matrix field assignments, as illustrated in 1.

$$W = f_{zipf} + (f_{zipf} * w_{sem}) \quad (1)$$

Thus, if a word would otherwise receive a frequency score of 0.5 but it falls within a high compatibility semantic field, it receives a final frequency weight of 0.5 (base Zipfian) + (0.5 base Zipfian * 0.9 compatibility weight) = 0.95, whereas one that falls within a low compatibility semantic field receives a final weight of 0.5 + (0.5 * 0.3) = 0.65.

CFG feature ablation Ablation experiments with the different CFG features—morphological affixation, agglutination, semantic clustering—showed that a CFG using all of our linguistically-inspired features had better loss curves when training on subsequent BabyLM data than CFG using only one or a subset of features (see Figure 2).

We generated 1,000,000 sequences of 20.5 average word length using this system.

4 Method

4.1 Stages

Following Hu et al. (2025), we approached training in two stages: stage 1, in which we train on the non-human language data sources in order to induce a "UG" state, and stage 2, which trains on the BabyLM datasets for natural language learning.

We performed experiments on the multilingual BabyLM challenge sets (English, Dutch, and Chinese), with results reported in section 5, along with some testing on the BabyLM strict track (100M token) dataset and strict-small (10M), which are reported in the Appendix.

Our challenge track models take the `MissUniverse` moniker, while all others represent experiment ablations and explorations.

4.2 Tokenizers

For the sake of simplicity in these experiments, we used a custom BPE tokenizer that is trained up front before beginning stage 1 or stage 2 training. The tokenizer is trained from the stage 1 data (e.g., CFG, proteins, crystals, or a mix thereof) and a subset of stage 2 data (i.e., BabyLM data) so that the same tokenization and vocabulary embeddings can be used for both stages.[5]

For English and Dutch, we use whitespace separation and lowercase normalization, and for Chinese we use NFC unicode normalization and byte fallback for the BPE tokenizer. Our vocabulary size is approximately 8,000 tokens.

We also include a test with a custom shared tokenizer, trained on all data sources (CFG, proteins, crystals, English, Dutch, and Chinese)[6], and increased the vocabulary size to 32,000. The results for this shared tokenizer test are available in the Appendix.

4.3 Models

Our models implement GPT-2 (Radford et al., 2019) as a base architecture as it fits well on a single

[5]Future research may consider experimenting with tokenizer adaptation between stages so that the network could be trained with a stage 1 tokenizer for non-language data before being adapted to a stage 2 tokenizer for natural language data.

[6]We used ratios according to our most multilingual test case for this tokenizer: approximately 30% allocated to CFG, proteins, and crystals (in a 1:1:0.5 ratio themselves, by sequence count) and 70% allocated to a 1:1:1 mix of English, Dutch, and Chinese.

GPU, allowing for a high degree of experimentation with low computational costs.[7] We performed tests on both large (approximately 0.7B param) and small (124M param) GPT-2 model sizes,[8] the latter of which is reported in the Appendix.

Following Hu et al. (2025), we used a max input length of 1024 to take advantage of potential long distance dependencies in protein sequences. Since the individual CFG and crystal sequences were often much shorter, we packed multiple "sentences" into the input. We found this improved results over training on too many padding tokens.[9]

4.4 Task

We trained our models on a standard autoregressive task, focusing on the effects of the stage 1 data pretraining. We selected the autoregressive architecture as our underlying base because its unidirectional nature mimics how humans process language inputs incrementally, building expectations of the sentence's syntactic structure as we process the linearized outputs of a speaker or text (Marslen-Wilson, 1975; Konieczny, 2000; van Gompel and Pickering, 2007). Furthermore, GPT2 specifically has been found to be a better predictor of online human language processing (Lee et al., 2024).[10]

We leave experimentation with masked language modeling tasks to future research.

Evaluation We primarily evaluate our UG pretraining method on a multilingual mixture of English, Dutch, and Chinese datasets with the provided BabyLM evaluation pipelines. When relevant, we compare our models to vanilla GPT2 baselines that trained from random initialization solely on natural language, without interrupted staged training.

5 Results

We report the by-language results of our MissUniverse models in Table 1,[11] and compare our models to two baselines: a vanilla GPT2 model trained on the full 100M BabyLM token budget (`random init 100M`) and a vanilla GPT2 model trained on an amount of natural language data equivalent to the roughly 70M tokens the MissUniverse models (`random init 70M`).

From Table 1, we can observe that few instances of our MissUniverse models outperform the 100M baseline overall, with the exceptions of our CFG-protein and protein-crystal mix for Chinese evals. That being said, several of our MissUniverse models outperform the 70M baseline, which trained on an equivalent amount of natural language data. When examining the overall multilingual zero-shot average (see Table 2), only `MissUniverse-c-p` outperforms the equivalent 70M baseline while still falling short of 100M tokens of natural language training.

We might conclude from this that only the CFG-protein mix confers a linguistic bias that exceeds what can be learned from 70M tokens alone, and that this linguistic boost does not surpass what could have been learned from utilizing the 30M tokens on natural language. However, two key observations stand out. First, when the overall score is calculated using zero-shot and fine-tuning results (Table 2), all MissUniverse models exceed both baselines with the exception of `MissUniverse-c-p`, which still outperforms the 70M baseline. Second, we observe that in Table 1, many of our models outperform the baselines when the benchmark is *not* evaluating grammatical knowledge. In Table 2, we split evaluations into grammatical (BLiMP and MultiBLiMP, when relevant) and non-grammatical (HellaSwag, Winogrande, GlobalPIQA, XStoryCloze, and XCOMPS, when relevant). Upon doing so, we find that stage 1 pretraining by and large confers a notable benefit when the evaluation is *not* testing the model's grammatical ability, with our models exceeding even the 100M baseline for Chinese (`p-cr`) and English (`c-p-cr` and `c-p`).

[7] Initial experiments suggest that our approach of pretraining on non-human language data—including synthetic CFGs, protein sequences, and crystal CIF sequences—should extend to other small language models like the Phi family of models (Javaheripi et al., 2023); however, due to resource constraints, we focus our experimentation on the GPT-2 architecture.

[8] While the aim was for 1B param and 124M param sizes, due to implementing a more limited vocabulary size in order to train better on our pretraining data mixes, our true "large" model size was 706.7M parameters. Likewise, the 124M parameter model is truly 98M parameters. For reference purposes in this paper, we use "0.7B param" for our large GPT-2 model size, and "124M param" for our small GPT-2 model size.

[9] When stage 1 data sources were mixed in a single model, packing was heterogeneous.

[10] While humans receive and produce language incrementally, research suggests that they build hierarchical expectations during processing and utilize hierarchical planning during production (Lee et al., 2013). Future work could consider exploring how unidirectional and bidirectional training objectives or a combination of the two affect syntactic bias induction.

[11] Note that `random init 70M-multi` is averaged from two runs.

Multilingual Model (NL Evals)	Avg (NL)	BLiMP	MultiBLiMP	HellaSwag	Winogrande	XStoryCloze(NL)	XCOMPS	GlobalPIQA	FineTune
random init 100M-multi	**53.76**†	73.75†	87.77†	26.75†	49.88†	47.60	51.41†	39.14†	37.79†
random init 70M-multi	**53.14**	72.34	86.25	26.53	49.59	47.99⋆	51.19	38.10	36.63
MissUniverse-c-p-cr	**52.94**	71.61	85.16	26.48	50.04†⋆	48.37†⋆	50.77	38.12†	37.18†
MissUniverse-c-p	**53.05**	72.44†	85.20	26.65†	49.63†	50.39†⋆	50.95	36.11	36.85†
MissUniverse-p-cr	**52.92**	72.09	84.94	26.29	50.21†⋆	49.15†⋆	50.12	37.61	38.27†⋆
MissUniverse-c-cr	**52.94**	71.37	85.33	26.65†	50.04†⋆	48.99†⋆	51.50†⋆	36.68	38.09†⋆
Multilingual Model (ZH Evals)	**Avg (ZH)**	**BLiMP**		**HellaSwag**	**Winogrande**	**XStoryCloze**	**XCOMPS(Zho)**	**GlobalPIQA**	**Finetune**
random init 100M-multi	**47.47**†	74.27†		26.76†	49.38	46.67	52.56	35.15†	39.99
random init 70M-multi	**47.45**	72.32		26.74	51.07⋆	47.33⋆	52.58⋆	34.64	40.61⋆
MissUniverse-c-p-cr	**47.19**	72.69†		26.64	48.06	48.84†⋆	52.70†⋆	34.18	40.61†⋆
MissUniverse-c-p	**47.67**†⋆	73.83†		26.85†⋆	49.63⋆	47.91†⋆	51.18	36.65†⋆	39.86
MissUniverse-p-cr	**47.47**†⋆	71.77		26.92†⋆	49.79⋆	48.84†⋆	52.33	35.15†⋆	40.51⋆
MissUniverse-c-cr	**46.85**	72.76†		26.89†⋆	49.96⋆	47.52†⋆	51.25	32.74	40.06⋆
Multilingual Model (EN Evals)	**Avg (EN)**	**BLiMP**	**MultiBLiMP**	**HellaSwag**	**Winogrande**	**XStoryCloze**		**GlobalPIQA**	**Finetune**
random init 100M-multi	**52.48**†	65.30†	83.64†	26.72	51.36†	49.77†		38.08†	41.04†
random init 70M-multi	**51.44**	64.46	81.82	26.84⋆	51.36⋆	49.54		34.62	40.60
MissUniverse-c-p-cr	**51.66**†	64.33	79.61	26.88†⋆	49.22	49.85†⋆		40.08†⋆	42.64†⋆
MissUniverse-c-p	**52.09**†	64.21	81.17	26.77†⋆	51.20	49.69†		39.52†⋆	41.94†⋆
MissUniverse-p-cr	**51.33**	64.74†	80.00	26.57	51.28	49.23		36.15†	42.98†⋆
MissUniverse-c-cr	**52.01**†	64.17	82.73†	26.79⋆	49.79	49.46		39.11†⋆	43.05†⋆

Table 1: Multilingual zero-shot evaluation results for Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom). Average reports the zero-shot average per language. Best performers are underlined, and those that beat or tie with `random init (70M)` are marked with a † while those that beat or tie with `random init (100M)` are marked with ⋆.

We postulate that this may be due to stage 1 acquiring some of the basic operations of syntax, thus freeing up some of the model's resources in stage 2 for world-knowledge acquisition, though this seems to come at the cost of more language-specific syntactic acquisition. It is possible that the lagging grammatical knowledge is due to our models needing to learn not just three grammars, but five or even six grammars: the grammar of our CFG, of proteins, and of crystal structures, in addition to the three natural languages. To test this, we conduct two follow-up experiments: we reduce the number of grammars the model is exposed to in stage 1 training (Section 5.1), and we reduce the number of languages in stage 2 down to a single, monolingual condition (Section 5.2).

5.1 Single-Source Stage 1

To reduce the number of grammars the models are exposed to, we limit the number of stage 1 inputs to a single non-language source. We additionally include a new CFG that includes both SVO and SOV word order in order to see if that improves the models' performance on Dutch, which has varied word-order.

As Table 3 demonstrates, this method did not drastically impact scores. English scores remain relatively unchanged, with several models beating the 70M baseline but not the 100M. Chinese and Dutch show slight changes: previously no model beat any baseline for Dutch, while `protein` and `shuffle-dyck` are able to exceed the 70M baseline. For Chinese, previously two models (CFG-protein and protein-crystal) outperformed both baselines, whereas for single-source only `shuffle-dyck` is able to tie with the 70M. Overall, we do not find reducing the number of input sources for stage 1 to simplify the system enough for our multilingual models to excel over the baselines for zero-shot results.[12]

5.2 Monolingual Stage 2

As a second follow-up exploration, we reduce the number of languages the model trains on. Rather than focusing on multilingual models as our MissUniverse models did, we instead turn our attention to the effects of stage 1 training effects on *monolingual* models. For baseline comparisons, we trained a vanilla GPT2 baseline on 100M tokens of a single language: English, Chinese, Dutch. We then initiated stage 1 training on a data mix of CFG, proteins, and crystals, as well as single-source inputs, including shuffle-dyck. Because Dutch demonstrates SOV and has some measure of flexible word-ordering, we additionally included single-source CFGs that were SVO, SOV, and a mix of the two.

When we compare the results of Table 4 to those in Table 1 and Table 3, we find a large increase in the number of models that exceed a model that was trained on 100M of natural language data. Previously, only the CFG-protein and CFG-crystal

[12]In this follow-up experiment, we did not fine-tune the new single-source models due to compute restraints. Future research may consider exploring the differences between multi-source and single-source stage 1 training on fine-tune evaluations.

Model	Multilingual Avgs				NL Avgs		ZH Avgs		EN Avgs	
	Overall	Zero-shot	Gram	Other	Gram	Other	Gram	Other	Gram	Other
random init 100M-multi	**44.72**†	**51.37**†	76.95†	42.23†	80.76†	42.96†	74.27†	42.10	74.47†	41.48†
random init 70M-multi	**44.31**	**50.81**	75.44	42.01	79.30	42.68	72.32	42.47⋆	73.14	40.59
MissUniverse-c-p-cr	**44.76**†⋆	**50.72**	74.68	42.16†	78.39	42.76†	72.69†	42.08	71.97	41.51†⋆
MissUniverse-c-p	**44.56**†	**51.05**†	75.37	42.37†⋆	78.82	42.75†	73.83†	42.44†	72.69	41.80†⋆
MissUniverse-p-cr	**44.99**†⋆	**50.70**	74.71	42.12†	78.52	42.68†	71.77	42.61†⋆	72.37	40.81†
MissUniverse-c-cr	**44.90**†⋆	**50.72**	75.25	41.95	78.35	42.77†	72.67†	41.67	73.45†	41.29†

Table 2: Multilingual averages, with the Overall including fine-tune evals in addition to zero-shots. Zero-shot evals are further sub-divided into grammatical benchmarks (BLiMP and MultiBLiMP when available) and non-grammatical "other" benchmarks (HellaSwag, Winogrande, XstoryCloze, XCOMPS when available, and GlobalPIQA). Best-performing models are underlined; models that outperform or tie with the 70M baseline are denoted with † while those that outperform the 100M baseline are marked with ⋆.

Multiling Model	Multiling Avg	NL Avg	ZH Avg	EN Avg
rand init 100M	**51.37**†	53.76†	47.47†	52.48†
rand init 70M	**50.81**	53.14	47.45	51.44
CFG (SVO)	**50.14**	52.40	46.60	51.05†
CFG (SVO-SOV)	**50.70**	52.21	47.43	52.20†
protein	**50.91**†	53.19†	47.42	51.75†
crystal	**50.54**	52.55	46.38	52.35†
shuffle-dyck	**50.83**†	53.52†	47.45†	51.08

Table 3: Single source multilingual zero-shot evaluation results for Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom). Best models are *underlined*. Those that beat or tie with the random init baselines are marked with † for random init (70M) and ⋆ for random init (100M).

Monolingual Model	ZH Avg	EN Avg	NL Avg
rand init 100M	48.89	52.12	55.32
SVO-p-cr	48.85	51.80	56.14⋆
SVO-p	48.72	52.00	54.75
p-cr	47.55	52.03	56.14⋆
SVO-cr	49.66⋆	52.50⋆	55.71⋆
CFG (SVO)	49.29⋆	52.30⋆	54.08
CFG (SOV)			54.42
CFG (SVO-SOV)			55.77⋆
protein	48.16	52.62⋆	54.42
crystal	49.11⋆	51.98	55.80⋆
shuffle-dyck	49.29⋆	51.95	56.20⋆

Table 4: Monolingual zero-shot evaluation results, trained on Chinese (ZH), English (EN), or Dutch (NL) in stage 2. Best models are *underlined*, and those that beat or tie with the random init baseline are marked with a ⋆.

MissUniverses outpaced their 100M baseline and only for Chinese. However, if the model's required training is reduced from learning three languages to learning merely one, the benefits of the syntactic induction begin to outpace the benefits of training on an extra 30M tokens of natural language data.

By reducing the models' natural language acquisition demands, the benefits of multi-source stage 1 training become more prevalent, particularly for Dutch, which exceeds the baseline for all but the SVO-protein mix. Interestingly, we find the SVO-cr mix to benefit all three languages evaluated in BabyLM.

We postulate the strength of SVO-cr across Chinese, English, and Dutch is due in part to the linguistically-informed nature of the CFG, especially for these languages that demonstrate SVO order. It is possible that the CFG is closer in nature to the syntax demonstrated in the natural language data, thus conferring a greater benefit. Evidence of this hypothesis comes from the SVO, SOV, and mixed SVO-SOV comparisons in Dutch, a language with flexible word-order (Dryer, 2013). By training on a CFG that has both SVO and SOV word-order, the Dutch zero-shot average increases by roughly 3% compared to either word-orders alone.

In sum, we find that when the language requirements of stage 2 are reduced, the benefits of our UG-induction method through stage 1 training emerge more clearly. Future research may consider exploring whether the benefits of stage 1 UG-induction emerges in bilingual models, or if the induction is best limited to monolingual models when the token budget is capped at 100M tokens.

6 Discussion

For multilingual models, our "UG" pretraining confers a benefit on non-grammar-focused zero-shot evals but not grammar ones, perhaps still suffering from the 30M loss of natural language data. When the model is given additional training time via fine-tuning (a "stage 3" of training), our MissUniverse models catch up and surpass the baselines, even 100M. When "stage 3" isn't available (i.e., zero-shots), reducing the number of input sources in stage 1 is not enough to surpass what would have

been gained by just giving the model an extra 30M tokens of natural language. However, if the model is required to only acquire one language, then again we see the benefits of stage 1 training.

6.1 Universal Grammar

While we refer to a Universal Grammar style biasing throughout this work, we do not imply that our CFG, crystals, and protein mixes are actually representative of the UG in humans, if one exists. Instead, this work explores what a UG-like bias could look like in artificial neural networks using synthetic language-like sequences and biological data sources that demonstrate properties similar to Merge and Move (e.g., protein folding). With this biasing, we achieve high performance in non-grammatical tasks, monolingual settings, and fine-tuning on downstream tasks.

However, the success of non-language sources for network biasing raises some interesting questions regarding the human brain bias for language. Perhaps the theorized human Universal Grammar may also be better described as a primed state for generalized downstream learning, akin to our stage 1 pre-trained models, rather than an explicit framework of language patterns.[13]

6.2 Future Steps

There are several avenues for future looking research. One, exploring what about each stage 1 source confers which benefits. We already postulate that the CFG may confer more benefits when its word order is aligned to the target language, but further investigation is needed. Likewise, which grammatical operations did the stage 1 sources help with, and did it hurt? Why does protein perform well for Dutch but poorly for Chinese?

In continued regards to the biological sources, it is unclear whether the performance boost often seen by adding crystals to proteins is due to additional structural complexity, or whether the crystals merely had a regularization effect.

Other aspects of language modeling and acquisition, like tokenizer adaptation from stage 1 to stage 2, incorporating a masked language modeling task, or dynamically changing task objectives to more closely mimic human cognitive development (Thelen and Smith, 1994), we similarly believe are promising future avenues for investigation.

[13]After all, if these patterns occur in other biological sequences then perhaps UG is less about language than it is about truly universal patterns.

7 Conclusions

It has already been demonstrated in the literature that LLMs share grammatical representations across multiple languages (Starace et al., 2023; Wendler et al., 2024; Zhao et al., 2024; Brinkmann et al., 2025). And Dhaliwal et al. (2026) find that adding additional languages to training improves model performance, particularly for low-resource languages. We hypothesize that by training on our data mixes, which are not derived from any human language but which never the less demonstrate structured data with basic syntactic foundations that are shared with human languages, we prime our models to a non-language-specific "UG"-like state that enables the models to better perform on a variety of languages, despite the reduced natural language input, and learn downstream (fine-tuning) tasks better.

Our results align with previous literature finding success in biasing neural networks towards linguistic patterns with careful pretraining of the network (McCoy et al., 2020; Mueller and Linzen, 2023; Papadimitriou and Jurafsky, 2023; Lindemann et al., 2024; Hu et al., 2025; Petty et al., 2025; Aryabumi et al., 2025; Kim et al., 2024).

Despite seeing less natural language data, our results pretraining on different sources of syntax demonstrate just how "universal" a UG really can be, and suggest interesting future research at the intersection of language, networks, and the natural world.

Limitations

As much as we promote here a "Universal Grammar" pretraining, the synthetically generated CFG we created is still somewhat English-centric, with an SVO head-right grammatical structure and using English phonotactic nonce words. While we begin early explorations into word order alternations, future research in this area should still address a wider variety of grammatical structures to more closely match existing human linguistic typology.

Our tests were limited to the languages presented, and it is possible that upon further testing, our results may not hold cross-linguistically with a wider set. Initial tests with Farsi (see Appendix section E) merit further investigation regarding word order influence.

We also were not able to test every possible biological source sequence fully. Virus genome sequences for example, may be an area of potential

study, or may not prove beneficial despite similar syntactic properties in genome sequences.

The results we present here are limited to autoregressive language modeling, and may or may not hold with a masked language modeling task.

Likewise, compute resources and timing constraints limited our ability to test full ablations. Some variations may change our existing comparative rankings, and as discussed briefly in the Appendix, our results here may not hold equally well with smaller (124M parameter) models. We have also not had the chance to fully test models larger than 0.7B parameters, similarly limiting the extent of our findings. We hope to test this method on larger models in the future.

References

Ben Ambridge and Elena V. M. Lieven. 2014. Child language acquisition: Why universal grammar doesn't help. *Linguistic Society of America*, 90(3):53–90.

Christian B. Anfinsen. 1973. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230.

Luis M. Antunes, Keith T. Butler, and Ricardo Grau-Crespo. 2024. Crystal structure generation with autoregressive large language modeling. *Nature Communications*, 15.

Viraat Aryabumi, Yixuan Su, Raymond Ma, Adrien Morisot, Ivan Zhang, Acyr Locatelli, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. 2025. To code or not to code? exploring impact of code in pre-training. In *The Thirteenth International Conference on Learning Representations*.

Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy DeGiovanni, Jose H. Pereira, Andria V. Rodrigues, Alberdina A. van Dijk, Ana C. Ebrecht, and 13 others. 2021. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876.

Fan Bai, Alan Ritter, and Wei Xu. 2021. Pre-train or annotate? domain adaptation with a constrained budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5002–5015, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jannik Brinkmann, Chris Wendler, Christian Bartelt, and Aaron Mueller. 2025. Large language models share representations of latent grammatical concepts across typologically diverse languages. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6131–6150, Albuquerque, New Mexico. Association for Computational Linguistics.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press.

Noam Chomsky. 1981a. *Lectures on government and binding*. Foris Publications.

Noam Chomsky. 1981b. *Rules and Representations*. Basil Blackwell.

Noam Chomsky. 1986. *Knowledge of Language: Its Nature, Origin, and Use*. Praeger.

Noam Chomsky. 1995. *The Minimalist Program*. MIT Press, Cambridge, MA.

Eve V. Clark. 2024. *First Language Acquisition*, 4th edition. Cambridge University Press.

Stephen Crain. 1991. Language acquisition in the absence of experience. *Behavioral and Brain Sciences*, pages 597–612.

Mehak Dhaliwal, Shashwat Chaurasia, Yao Qin, Dezhi Hong, and Thomas Butler. 2026. English is not all you need: Systematically exploring the role of multilinguality in llm post-training. *Preprint*, arXiv:2604.13286.

Ken A. Dill, Adam Lucas, Julia Hockenmaier, Liang Huang, David Chiang, and Aravind K. Joshi. 2007. Computational linguistics: A new tool for exploring biopolymer structures and statistical mechanics. *Polymer*, 48(15):4289–4300.

Matthew S. Dryer. 2013. Order of subject, object and verb (v2020.4). In Matthew S. Dryer and Martin Haspelmath, editors, *The World Atlas of Language Structures Online*. Zenodo.

Nick Evans and Stephen C. Levinson. 2009. The myth of language universals: Language diversity and its importance for cognitive science. *Behavioural and Brain Sciences*, 32:429–492.

Kariem Ezzat and Alberto J. Espay. 2023. Chapter 2 - the allure and pitfalls of the prion-like aggregation in neurodegeneration. In Alberto J. Espay, editor, *Precision Medicine in Neurodegenerative Disorders, Part II*, volume 193 of *Handbook of Clinical Neurology*, pages 17–22. Elsevier.

Xiaomin Fang, Fan Wang, Lihang Liu, Jingzhou He, Dayong Lin, Yingfei Xiang, Kunrui Zhu, Xiaonan Zhang, Hua Wu, Hui Li, and Le Song. 2023. A method for multiple-sequence-alignment-free protein structure prediction using a protein language model. *Nature Machine Intelligence*, 5:1087–1096.

Michael C. Frank. 2023. Bridging the data gap between children and large language models. *Trends in Cognitive Sciences*, 27(11):990–992.

Luke Gessler and Nathan Schneider. 2023. Syntactic inductive bias in transformer language models: Especially helpful for low-resource languages? In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 238–253, Singapore. Association for Computational Linguistics.

Jill Gilkerson, Jeffrey A. Richards, Judith K. Montgomery Steven F. Warren, Charles R. Greenwood, D. Kimbrough Oller, John HL Hansen, and Terrance D. Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American Journal of Speech-Language Pathology*, 26(2):248–265.

Lila R. Gleitman and Elissa L. Newport. 1995. *An Invitation to Cognitive Science*, 2nd edition, volume 1, chapter The Invention of Language by Children: Environmental and Biological Influences on the Acquisition of Language. MIT Press.

Alain Goriely. 2024. Eighty-six billion and counting: do we know the number of neurons in the human brain? *Brain*, (3):689–691.

Daniil Gurgurov, Tanja Baeumel, Josef van Genabith, and Simon Ostermann. 2026. Sparse subnetwork enhancement for underrepresented languages in large language models. *Preprint*, arXiv:2510.13580.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. Between circuits and Chomsky: Pre-pretraining on formal languages imparts linguistic biases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9691–9709, Vienna, Austria. Association for Computational Linguistics.

Kevin Maik Jablonka, Qianxiang Ai, Alexander Al-Feghali, Shruti Badhwar, Joshua D. Bocarsly, Andres M. Bran, Stefan Bringuier, L. Catherine Brinson, Kamal Choudhary, Defne Circi, Sam Cox, Wibe A. de Jong, Matthew L. Evans, Nicolas Gastellu, Jerome Genzling, María Victoria Gil, Ankur K. Gupta, Zhi Hong, Alishba Imran, and 34 others. 2023. 14 examples of how llms can transform materials science and chemistry: a reflection on a large language model hackathon. *Digital Discovery*, 2(5):1233–1250.

Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. 2013. The materials project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1).

Muratahan Aykol Bryce Meredig C. Wolverton James E. Saal, Scott Kirklin. 2013. Materials design and discovery with high-throughput density functional theory: The open quantum materials database (oqmd). *JOM*, 65:1501–1509.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, and 1 others. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Aravind K Joshi, K. Vijay-Shanker, and David Weird. 1990. *Foundational Issues in Natural Language Processing*, chapter The Convergence of Midly Context-Sensitive Gramamr Formalisms. MIT Press, Cambridge, MA, USA.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2025. Babybabellm: A multilingual benchmark of developmentally plausible training data. *Preprint*, arXiv:2510.10159.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, and 15 others. 2021. Applying and improving alphafold at casp14. *Proteins*, 89(12):1711–1721.

Gaurav Kamath, Sebastian Schuster, Sowmya Vajjala, and Siva Reddy. 2024. Scope ambiguities in large language models. *Transactions of the Association for Computational Linguistics*, 12:738–754.

Subhojyoti Khastagir, Kishalay Das, Pawan Goyal, Seung-Cheol Lee, Satadeep Bhattacharjee, and Niloy Ganguly. 2025. Llm meets diffusion: A hybrid framework for crystal material generation. In *Proceedings of the 39th Conference on Neural Informatio Processing Systems (NeurIPS)*.

Najoung Kim, Sebastian Schuster, and Shubham Toshniwal. 2024. Code pretraining improves entity tracking abilities of language models. *Preprint*, arXiv:2405.21068.

Lars Konieczny. 2000. Localitya nd parsing complexity. *Journal of Psycholinguistic Research*, 29:627–645.

Chloe Korade, Elena Nicoladis, and Monique Charest. 2024. Monolingual and bilingual children's production of complex syntactic structures. *Journal of Child Language*, 51(2):454–469.

Mark Kretschmann. [link].

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Eun-Kyoung Rosa Lee, Sathvik Nair, and Naomi H. Feldman. 2024. A psycholinguistic evaluation of language models' sensitivity to argument roles. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3262–3274, Miami, Florida, USA. Association for Computational Linguistics.

Eun-Kyung Lee, Sarah Brown-Schmidt, and Duane G. Watson. 2013. Ways of looking ahead: Hierarchical planning in language production. *Cognition*, 129(3):544–562.

Cyrus Levinthal. 1968. Are there pathways for protein folding? *J. Chim. Phys.*, 65:44–45.

Bingzhi Li, Guillaume Wisniewski, and Benoit Crabbé. 2021a. Are Transformers a modern version of ELIZA? Observations on French object verb agreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4599–4610, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. 2021b. Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 645–653, Online. Association for Computational Linguistics.

David Lightfoot. 1989. The child's trigger experience: Degree-0 learnability. *Behavioral and Brain Sciences*, 12:321–334.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. 2023a. Esm-2 uniref pretraining data. `https://huggingface.co/datasets/nvidia/esm2_uniref_pretraining_data`.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. 2023b. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130.

Matthias Lindemann, Alexander Koller, and Ivan Titov. 2024. Strengthening structural inductive biases by pre-training to perform syntactic transformations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11558–11573, Miami, Florida, USA. Association for Computational Linguistics.

Wei Liu, Ming Xiang, and Nai Ding. 2026. Active use of latent tree-structured sentence representation in humans and large language models. *Nature Human Behavior*, 10:303–316.

William D. Marslen-Wilson. 1975. Sentence perception as an interactive parallel process. *Science*, 189(4198):226–228.

Andrea Matarazzo and Riccardo Torlone. 2025. A survey on large language models with some insights on their capabilities and limitations. *Preprint*, arXiv:2501.04040.

R. Thomas McCoy, Erin Grant, Paul Smolensky, Thomas L. Griffiths, and Tal Linzen. 2020. Universal linguistic inductive biases via meta-learning. *Preprint*, arXiv:2006.16324.

Aaron Mueller and Tal Linzen. 2023. How to plant trees in language models: Data and architectural effects on the emergence of syntactic inductive biases. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11237–11252, Toronto, Canada. Association for Computational Linguistics.

Elena Nicoladis and Zixia Jiang. 2018. Language and cognitive predictors of lexical selection in storytelling for monolingual and sequential bilingual children. *Journal of Cognition and Development*, 19(4).

Dmitry Nikolaev and Sebastian Padó. 2023. Investigating semantic subspaces of transformer sentence embeddings through linear structural probing. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 142–154, Singapore. Association for Computational Linguistics.

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8402–8413, Singapore. Association for Computational Linguistics.

Jackson Petty, Sjoerd van Steenkiste, and Tal Linzen. 2025. How does code pretraining affect language model task performance? *Transactions on Machine Learning Research*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. 2021. Transformer protein language models are unsupervised structure learners. In *International Conference on Learning Representations*.

Ian Roberts. 2017. *The Oxford Handbook of Universal Grammar*, chapter Introduction. Oxford University Press.

Markus Scheidgen, Alvin Noe Ladines Lauri Himanen 1*, David Sikter, Mohammad Nakhaee, Ádám Fekete, Theodore Chang, Amir Golparvar, José A. Márquez, Sandor Brockhauser, Sebastian Brückner, Luca M. Ghiringhelli, Felix Dietrich, Daniel Lehmberg, Thea Denell, Andrea Albino, Hampus Näsström, Sherjeel Shabih, Florian Dobener, Markus Kühbach, and 9 others. 2023. Nomad: A distributed web-based platform for managing materials science research data. *Journal of Open Source Software*, 8(90).

Giulio Starace, Konstantinos Papakostas, Rochelle Choenni, Apostolos Panagiotopoulos, Matteo Rosati, Alina Leidinger, and Ekaterina Shutova. 2023. Probing LLMs for joint encoding of linguistic categories. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7158–7179, Singapore. Association for Computational Linguistics.

Baris E. Suzek, Yuqi Wang, Hongzhan Huang, Peter B. McGarvey, Cathy H. Wu, and the UniProt Consortium. 2015. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932.

J. Ben Tamo, Daniel Carlander-Reuterfelt, Jonathan Rubin, Oleg Poliannikov, Dezhi Hong, and Mingxian Wang. 2026. Linguamap: Which layers of LLMs speak your language and how to tune them? In *The Fourteenth International Conference on Learning Representations*.

Esther Thelen and Linda B Smith. 1994. *A dynamic systems approach to the development of cognition and action*. MIT Press.

Michael Tomasello. 2009. Ug is dead. *Behavioral and Brain Sciences*, 32:470–471.

Roger P. G. van Gompel and Martin J. Pickering. 2007. *The Oxford Handbook of Psycholinguistics*, chapter Syntactic Parsing. Oxford University Press.

Jesse Vig, Ali Madani, Lav R. Varshney, Caiming Xiong, richard socher, and Nazneen Rajani. 2021. {BERT}ology meets biology: Interpreting attention in protein language models. In *International Conference on Learning Representations*.

Teng Wang, Zhenqi He, Wing-Yin Yu, Xiaojin Fu, and Xiongwei Han. 2025. Large language models are good multi-lingual learners : When LLMs meet cross-lingual prompts. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4442–4456, Abu Dhabi, UAE. Association for Computational Linguistics.

Wenkai Wang, Zhenling Peng, and Jianyi Yang. 2022. Single-sequence protein structure prediction using supervised transformer protein language models. *Nature Computational Science*, 2:804–814.

Alex Warstadt and Samuel R. Bowman. 2024. What artificial neural networks can tell us about human language acquisition. *Preprint*, arXiv:2208.07998.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Agata Wolna, Aaron Wright, Colton Casto, Samuel Hutchinson, Benjamin Lipkin, and Evelina Fedorenko. 2026. The extended language network: Language-responsive brain areas whose contributions to language remain to be discovered. *Journal of Neuroscience*.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-enhanced pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online. Association for Computational Linguistics.

Xiulin Yang, Arianna Bisazza, Nathan Schneider, and Ethan Gotlieb Wilcox. 2026. A unified assessment of the poverty of the stimulus argument for neural language models. *Preprint*, arXiv:2602.09992.

Lei Zhang, Zhaorui Liu, Ben Ni, and Quan Wang. 2026. Large language models (llms) for materials design. *Advanced Functional Materials*, 36(30):e25897.

Shuai Zhang, Wang Lijie, Xinyan Xiao, and Hua Wu. 2022. Syntax-guided contrastive learning for pre-trained language model. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2430–2440, Dublin, Ireland. Association for Computational Linguistics.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024. How do large language models handle multilingualism? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461, Online. Association for Computational Linguistics.

Yoel Zimmermann, Adib Bazgir, Zartashia Afzal, Fariha Agbere, Qianxiang Ai, Nawaf Alampara, Alexander Al-Feghali, Mehrad Ansari, Dmytro Antypov, Amro

Aswad, Jiaru Bai, Viktoriia Baibakova, Devi Dutta Biswajeet, Erik Bitzek, Joshua D. Bocarsly, Anna Borisova, Andres M Bran, L. Catherine Brinson, Marcel Moran Calderon, and 125 others. 2025. Reflections from the 2024 large language model (llm) hackathon for applications in materials science and chemistry. *Preprint*, arXiv:2411.15221.

A Context-Free Grammar Details

Tables 5, 6, 7, and 8 provide additional details for the creation of the context-free grammar.

Note, the compatibility matrices and semantic fields outlined do not represent real word meanings or assignments, but imagined groupings. All words are assigned randomly to different syntactic categories and imaginary semantic groups that may be loosely represented by the categories presented in these tables.

Category	Frequency boost
Determiners	10.0x
Nouns	8.0x
Tense/Inflection	6.0x
Verbs	5.0x
Conjunctions	3.0x
Complementizers	2.0x
Prepositions	1.5x
Adjectives/Adverbs	1.5x

Table 5: CFG syntactic categories and their frequency multipliers.

B Ablations

B.1 CFG feature ablations

Eval loss curves on CFG feature ablations showed most efficient training using all developed CFG features (Figure 2).

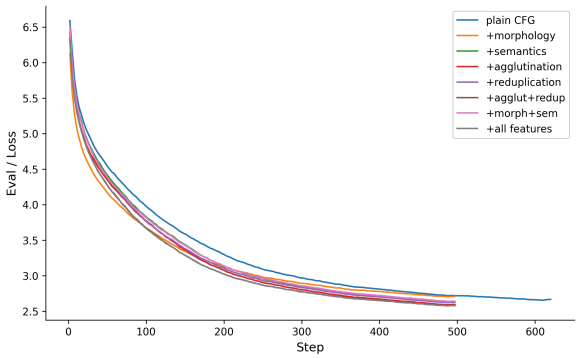


Figure 2: Eval loss curve ablations for CFG features, for stage 2 training with English BabyLM data.

B.2 Data mixing ablations

We also experimented with different proportions of CFG, protein, and crystal mixes. We present a subset of those results for comparison in Appendix F, Table 10.

C BabyLM English: Strict

Like Hu et al. (2025), our pretraining paradigm achieved lower loss faster than natural language training for our 0.7B strict (100M) models (see Figure 1). In particular, our stage 1 UG pretraining paradigm achieved a noticeable improvement in loss curves compared to three baselines: a random initialization baseline (i.e., skipping stage 1 pretraining and only training directly on BabyLM data for the full 1B total tokens seen), training on BabyLM natural language in both stage 1 and stage 2, and the Shuffle-Dyck pretraining language used in Hu et al. (2025), demonstrating the benefits of these rich sources of syntax for UG.

Mixing proteins and crystals in a 1:1 ratio resulted in good performance on the English BabyLM eval pipeline, as did CFG and proteins (Table 10). Crystal data did not appear to differ greatly from `Shuffle-Dyck` as a solo source for stage 1 training. When mixed in a 2:2:1 ratio with CFG and proteins, it performed well with our 0.7B model, but still not as high as either proteins+crystals or CFG+proteins.

D Small Models (124M params)

D.1 Small model size

Our early experiments were on the largest models we could run with our compute resources—the nearly 0.7B param size of GPT2—and showed high separation between the stage 1 pretraining data conditions. However, we also ran tests on the same 100M dataset size in a smaller 124M parameter GPT2 model. In an attempt to match posted BabyLM GPT2 2025 baselines, we also dropped the sequence packing stage 2 training.[14]

In the smaller model, the CFG and protein data sources did not appear to hold the same performance gain from our specialized data mixes (see Figure 3) and Table 10 in Appendix F). Two-stage training on natural language (BabyLM data) in both stages performed the best for the Strict dataset

[14]The pulled config for the BabyLM GPT2 baseline indicated a max sequence length of 1024. As such, we kept the long max sequence in place for these 124M tests.

	motion	cognition	creation	interaction	change	perception
living	0.9	0.8	-	0.9	-	0.9
objects	0.6	-	0.8	-	-	-
places	0.7	-	-	-	-	-
abstract	-	0.9	-	-	0.7	-
substances	-	-	-	-	0.8	-

Table 6: CFG noun-verb compatibility matrix for semantic clustering

	physical	evaluative	emotional
living	0.8	-	0.7
objects	0.9	0.8	
places	0.8	-	
abstract	-	0.9	0.8

Table 7: CFG noun-adjective compatibility matrix for semantic clustering

Pattern	Application Rate
V D N	0.15
P D N	0.12
D N	0.3
T V	0.21

Table 8: CFG agglutination template. Linking morphemes are [”, ”, ”, ’a’, ’o’, ’i’], with blanks representing bare concatenation.

size,[15] followed by Shuffle-Dyck stage 1 training, then our `CFG+protein+crystal` mix. Interestingly, in the smaller size, `CFG+protein+crystal` did better than only `CFG+protein`, which performed better in the larger model size.

Likewise, the 124M model with the Strict-Small dataset for stage 2 training showed slightly higher performance with `CFG+protein+crystal` compared to `CFG+protein`.[16]

D.2 Strict-Small

For the Strict-Small test with a model size of 124M parameters, we also pulled the stage 1 training down to 500 steps (approximately 16.5M tokens)[17].

We compared stage 1 training mixes of `CFG`, `proteins`, `shuffle dyck`, `CFG+protein`, `CFG+protein+crystals`, and `random init` (i.e., no stage 1 training).

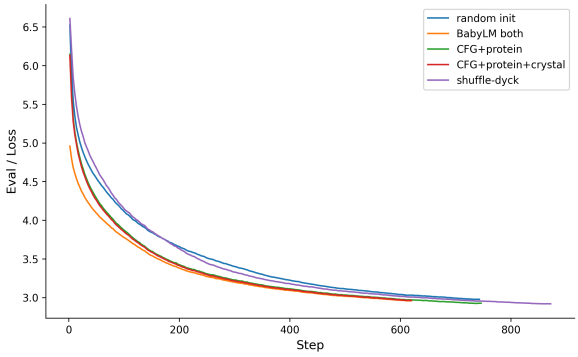


Figure 3: Eval loss curves comparing stage 1 data mixes for the 124M param model size for strict track (100M).

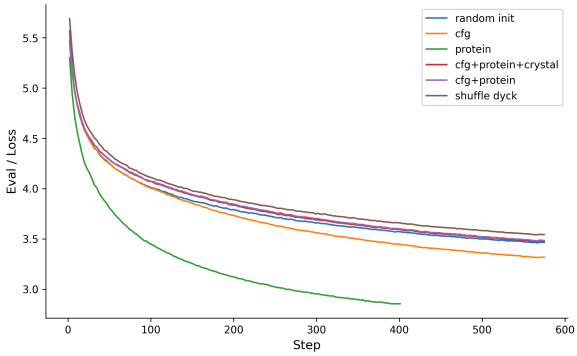


Figure 4: Eval loss curves comparing stage 1 data mixes for the 124M model size on the strict-small 10M dataset.

For this small model and dataset size, stage 1 training only on proteins performed dramatically better in the eval loss curve during training (see Figure 4), but not noticeably better in eval results (Table 10 in Appendix F). We saw somewhat similar behavior in ablations with our full 0.7B model on the larger Strict dataset, where the loss curves looked good for `protein only` but were not borne out in eval results (though the difference in loss curves was not so stark in the larger model ablation tests). From this we emphasize the need to look

[15] Our 124M model two-stage training with BabyLM data did still adhere to the 100M token budget.

[16] We do not include comparisons with BabyLM in both stages for the Strict-Small exploration because we did not limit stage 1 training to only 3M tokens, thus BabyLM in both stages would have just been additional training beyond the 10M English token budget.

[17] Since our stage 1 training is already over the Strict-Small challenge token budget, we did not submit this version to the formal challenge

closely at downstream eval in addition to rapid loss improvement.

As a further note on these Strict-Small tests, in keeping with our goal of creating a theoretical "UG" network state before language training begins, we did not limit stage 1 training to 30% of a total 10M token budget. Instead, stage 2 only was considered the language acquisition test for the small network, and used the entire 10M budget. This setup more closely mimics stage 1 as analogous to a "hardcoded" or "biological" bias towards syntactic structures.

D.3 Discussion

Our 124M models performed several points lower than our 0.7B parameter models for the strict track. One explanation for improved performance of 0.7B param model compared to 124M param model when trained on our UG-inducing datamixes is the complexity of the data and the models' ability to extract said linguistic data. Matarazzo and Torlone (2025) report the Scaling Law of LLMs, noting that as models increase in parameter size, so do their linguistic capabilities as the greater number of parameters enables the model to better capture and model the linguistic relations within the data. As such, larger models often achieve greater linguistic performance than their smaller counterpart models (Matarazzo and Torlone, 2025), a finding backed by smaller models performing worse than their larger counterparts on multilingual tasks (Wang et al., 2025; Gurgurov et al., 2026).

In light of this, we postulate that the larger number of parameters in the 0.7B sized GPT2 model better enables the model to take advantage of the shared UG principles in the data mixes in the first stage, then focus on vocabulary and knowledge acquisition in stage 2.

The higher success with larger models vs. smaller ones show we still have progress to be made with regards to network size and shape, considering that current LLMs have 1 trillion or more parameters (Kretschmann). Compare this to the human brain, which has between 61B and 99B neurons (Goriely, 2024), of which, only 5% of gray matter is used in language processing (Wolna et al., 2026).

E Additional Languages

We began investigating additional languages from the BabyBabel datasets (Jumelet et al., 2025), choosing Farsi[18] as it has a canonically SOV word order and fits nicely with the word order exploration. However, as seen in Table 9, it does not follow expected patterns with an SOV CFG.

Monolingual Model	ZH Avg	EN Avg	NL Avg	FA Avg
rand init 100M	48.89	52.12	55.32	50.11
SVO-p-cr	48.85	51.80	56.14⋆	49.10
SVO-p	48.72	52.00	54.75	49.56
p-cr	47.55	52.03	56.14⋆	49.82
SVO-cr	49.66⋆	52.50⋆	55.71⋆	49.76
CFG (SVO)	49.29⋆	52.30⋆	54.08	49.47
CFG (SOV)			54.42	48.49
CFG (SVO-SOV)			55.77⋆	48.77
protein	48.16	52.62⋆	54.42	48.93
crystal	49.11⋆	51.98	55.80⋆	50.34⋆
shuffle-dyck	49.29⋆	51.95	56.20⋆	

Table 9: Zero-shot evaluation results for monolingual models trained on Chinese (ZH), English (EN), Dutch (NL), or Persian/Farsi (FA) in stage 2. Best models are *underlined*, and those that beat or tie with the random init baseline are marked with a ⋆.

While Dutch shows a performance boost from a CFG word order well aligned with its own distribution, Farsi poses a challenge to this explanation. It is the SVO word-order that outperforms the other two CFGs, despite Farsi being canonically SOV. Perhaps the child-directed language included in the dataset includes instances of SVO order, though this requires a thorough investigation of the relative prevalence of each word-order in the Farsi BabyBabel data to answer.

F Full Tables

[18] We would like to acknowledge and thank Zahra Mirrazi for advising us on the syntax of Farsi in various domains.

Model *(0.7B params Strict)*	**Text Avg**	**BLiMP**	**BLiMP Supp.**	**EWoK**	**Entity Tracking**	**Comps**
CFG only	**50.172**	67.07	61.64⋆	51.02	18.48⋆	52.65
protein only	**49.806**	65.85	61.84⋆	50.18	18.94⋆	52.22
crystal only	**50.88**⋆	66.92	64.12⋆	52.72⋆	17.48⋆	53.16
CFG+protein	**50.794**	66.63	63.45⋆	52.43⋆	18.26⋆	53.2
protein+crystal	**51.376**	67.91	64.85	52.5⋆	18.12⋆	53.5
CFG+crystal	**51.024**⋆	68.02	62.95⋆	52.07⋆	18.39⋆	53.69
CFG2+protein1+crystal1	**49.174**	64.04	60.83⋆	50.4	17.33⋆	53.27
CFG2+protein2+crystal1	**50.814**	65.97	62.35⋆	52.88	19.39	53.48
BabyLM both	**48.188**	63.19	57.93	50.4	18.97	50.45
random init	**50.844**	71.06	60.3	52.04	16.29	54.53
Model *(124M params Strict)*	**Text Avg**	**BLiMP**	**BLiMP Supp.**	**EWoK**	**Entity Tracking**	**Comps**
CFG+protein	**49.234**⋆	63.9	57.34⋆	52.55	17.96⋆	54.42
CFG+protein+crystal	**49.27**⋆	63.09	58.53⋆	52.11	18.35⋆	54.27
shuffle dyck	**48.56**	64.26	56.75⋆	51.11	16.25	54.43
BabyLM both	**50.444**	64.66	59.96	53.41	19.45	54.74
random init	**49.038**	65.38	54.16	52.88	17.85	54.92
Model *(124M params Strict-small)*	**Text Avg**	**BLiMP**	**BLiMP Supp.**	**EWoK**	**Entity Tracking**	**Comps**
CFG+protein	**47.1**	61.3	57.97⋆	49.4	17.11	49.72
CFG+protein+crystal	**47.84**	62.28	57.97⋆	51.34⋆	17.59	50.02⋆
shuffle dyck	**46.794**	62.85	53.14	51.37	16.74	49.87
random init	**47.808**	63.94	57.08	50.2	17.89	49.93
CFG only	**47.232**	60.94	58.62	49.73	16.8	50.07⋆
crystal only	**47.23**	62.6	56.91	50.09	16.73	49.82
protein only	**46.512**	60.45	54.06	49.72	17.81	50.52

Table 10: Data mix eval results (0.7B param model + 100M strict track top; 124M param model + 100M strict track middle; 124M param model + 10M strict-small track bottom). The `random init` comparison ran over the full English data (100M or 10M, depending). For the 0.7B param strict models, two versions of the `cfg+protein+crystal` mix are included: one with a ratio of 2:1:1, and the other with 2:2:1. All other instances of `cfg+protein+crystal` are with the 2:2:1 ratio. Best models are underlined, and those that outperformed `random init` are marked with a ⋆.

Multilingual Model (NL Evals)	**Avg**	**POS(NL)**	**MNLI(NL)**		**SIB200(NL)**	**Belebele(NL)**	**ARC(NL)**	**TruthfulQA(NL)**	**BMLama(NL)**	**INCLUDE(NL)**
random init 100M-multi	**37.79**†	92.04†	43.86		56.00	22.73†	27.71†	16.07	18.05	25.89†
random init 70M-multi	**36.63**	91.82	43.92†		56.50⋆	21.59	23.75	18.75⋆	19.78⋆	16.96
MissUniverse-c-p-cr	**37.18**†	91.26	45.95†⋆		52.50	21.02	25.83†	16.96⋆	12.67	31.25†⋆
MissUniverse-c-p	**36.85**†	91.84†	43.92†⋆		48.00	20.45	25.83†	17.86⋆	17.47	29.46†⋆
MissUniverse-p-cr	**38.27**†⋆	92.50†⋆	40.48		64.50†⋆	19.89	26.88†	22.32†⋆	13.74	25.89†⋆
MissUniverse-c-cr	**38.09**†⋆	92.25†⋆	42.85		62.00†⋆	20.45	24.79†	20.54†⋆	16.80	25.00†
Multilingual Model (ZH Evals)	**Avg**	**POS(ZH)**	**MNLI(ZH)**	**XNLI(ZH)**	**SIB200(ZH)**	**Belebele(ZH)**	**ARC(ZH)**	**TruthfulQA(ZH)**	**BMLama(ZH)**	**INCLUDE(ZH)**
random init 100M-multi	**39.99**	88.65†	45.78†	42.75	76.50†	17.61	28.96†	22.32	13.25†	24.11
random init 70M-multi	**40.61**⋆	88.20	44.26	43.90⋆	74.00	23.86⋆	26.46	27.68⋆	11.26	25.89⋆
MissUniverse-c-p-cr	**40.61**⋆	87.88	45.38†	44.70†⋆	75.00†	23.30⋆	27.08†	20.54	13.08†	28.57†⋆
MissUniverse-c-p	**39.86**	88.71†⋆	46.06†⋆	43.40⋆	66.00	25.57†⋆	24.17	25.00⋆	13.99†⋆	25.89†⋆
MissUniverse-p-cr	**40.51**⋆	89.02†⋆	46.96†⋆	47.75†⋆	72.00	26.14†⋆	22.71	27.68†⋆	17.14†⋆	15.18
MissUniverse-c-cr	**40.06**⋆	88.94†⋆	48.48†⋆	42.95⋆	70.00	21.02⋆	23.94	28.57†⋆	13.41†⋆	23.21
Multilingual Model (EN Evals)	**Avg**	**POS(EN)**	**MNLI(EN)**	**XNLI(EN)**	**SIB200(EN)**	**Belebele(EN)**	**ARC(EN)**	**TruthfulQA(EN)**	**BMLama(EN)**	
random init 100M-multi	**41.04**†	90.17	46.57†	45.40	59.50†	21.02	26.25†	21.43	17.96	
random init 70M-multi	**40.60**	90.34⋆	44.31	45.45⋆	55.50	21.59⋆	25.42	18.75	23.43⋆	
MissUniverse-c-p-cr	**42.64**†⋆	89.31	44.09	45.80†⋆	62.50†⋆	21.02⋆	26.88†⋆	38.39†⋆	13.16	
MissUniverse-c-p	**41.94**†⋆	89.95	46.57†⋆	44.45	62.50†⋆	24.43†⋆	24.58	24.11†⋆	18.96†⋆	
MissUniverse-p-cr	**42.98**†⋆	90.32⋆	44.99†⋆	52.50†⋆	66.00†⋆	21.02⋆	21.46	27.68†⋆	19.87†⋆	
MissUniverse-c-cr	**43.05**†⋆	90.14	44.65†	46.15†⋆	72.00†⋆	21.02⋆	26.25†⋆	26.79†⋆	17.38	

Table 11: Multilingual fine-tune evaluation results for Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom). Averages are calculated for the MissUniverse models, as well as two single-source stage 1 models that outperformed three of the MissUniverse models). Best performers are underlined.

Model	Multilingual Avgs			NL Avgs		ZH Avgs		EN Avgs	
	Fine-tune	Synt-Sem	Other	Synt-Sem	Other	Synt-Sem	Other	Synt-Sem	Other
random init 100M-multi	**39.62**†	61.90†	29.14†	67.95†	27.74†	59.06†	30.46	60.71†	29.23†
random init 70M-multi	**39.33**	61.53	28.89	67.87	26.22	58.79	31.52⋆	60.03	28.94
MissUniverse-c-p-cr-multi	**40.16**†⋆	61.70†	29.98†⋆	68.61†⋆	26.70†	59.32†⋆	31.26⋆	59.73	32.39†⋆
MissUniverse-c-p-multi	**39.56**†	61.86†	29.07†	67.88†	26.51†	59.39†⋆	30.10	60.32†	30.92†⋆
MissUniverse-p-cr-multi	**44.99**†⋆	63.06†⋆	30.01†⋆	66.49	28.87†⋆	61.24†⋆	30.14	62.60†⋆	31.21†⋆
MissUniverse-c-cr-multi	**40.39**†⋆	62.05†⋆	30.19†⋆	67.55	28.26†⋆	60.12†⋆	30.03	60.31†	32.69†⋆

Table 12: Fine-tune evals sub-divided into syntax-semantics benchmarks (POS, MNLI, and XNLI, when relevant) and world-knowledge "other" benchmarks (SIB200, Belebele, ARC, TruthfulQA, BMLama, and INCLUDE, when relevant). Best-performing models are underlined; models that outperform or tie with the 70M baseline are denoted with † while those that outperform the 100M baseline are marked with ⋆.

Multilingual Model (NL Evals)	Avg	BLiMP(NL)	MultiBLiMP(NL)	HellaSwag(NL)	Winogrande(NL)	XStoryCloze(NL)	XCOMPS(NL)	GlobalPIQA(NL)
random init 100M-multi	**53.76**†	73.75†	87.77†	26.75†	49.88†	47.60	51.41†	39.14†
random init 70M-multi	**53.14**	72.34	86.25	26.53	49.59	47.99⋆	51.19	38.10
protein-multi	**53.19**†	72.06	84.81	26.46	49.22	48.22†⋆	51.48†⋆	40.09†⋆
shuffle-dyck-multi	**53.52**†	70.97	85.63	26.21	49.05	50.15†⋆	51.51†⋆	41.14†⋆
CFG (SOV & SVO)-multi	**52.21**	68.98	85.59	26.23	50.29†⋆	49.54†⋆	50.14	34.70
crystal-multi	**52.55**	71.94	84.86	26.48	49.30	48.68†⋆	50.90	35.70
CFG (SVO)-multi	**52.40**	69.47	83.35	26.30	48.60	48.68†⋆	50.81	39.61†⋆
Multilingual Model (ZH Evals)	**Avg**	**BLiMP(ZH)**		**HellaSwag(ZH)**	**Winogrande(ZH)**	**XStoryCloze(ZH)**	**XCOMPS(ZH)**	**GlobalPIQA(ZH)**
random init 100M-multi	**47.47**†	74.27†		26.76†	49.38	46.67	52.56	35.15†
random init 70M-multi	**47.45**	72.32		26.74	51.07⋆	47.33⋆	52.58⋆	34.64
protein-multi	**47.42**	71.86		26.58	51.28†⋆	46.59	52.57⋆	35.65†⋆
shuffle-dyck-multi	**47.45**†	70.73		26.83†⋆	50.95⋆	48.30†⋆	51.71	36.17†⋆
CFG (SOV & SVO)-multi	**47.43**	72.24		26.87†⋆	49.71⋆	47.45†⋆	52.61†⋆	35.70†⋆
crystal-multi	**46.38**	72.48†		26.80†⋆	50.62⋆	47.21⋆	51.95	29.21
CFG (SVO)-multi	**46.60**	69.32		26.52	49.71⋆	46.98⋆	51.88	35.17†⋆
Multilingual Model (EN Evals)	**Avg**	**BLiMP(EN)**	**MultiBLiMP(EN)**	**HellaSwag(EN)**	**Winogrande(EN)**	**XStoryCloze(EN)**		**GlobalPIQA(EN)**
random init 100M-multi	**52.48**†	65.30†	83.64†	26.72	51.36†	49.77†		38.08†
random init 70M-multi	**51.44**	64.46	81.82	26.84⋆	51.36⋆	49.54		34.62
protein-multi	**51.75**†	66.27†⋆	81.04	26.90†⋆	50.29	49.85†⋆		36.14†
shuffle-dyck-multi	**51.08**	64.96†	80.13	26.88†⋆	48.06	49.38		37.07†
CFG (SOV & SVO)-multi	**52.20**†	64.50†	80.00	26.98†⋆	51.28	49.38		41.05†⋆
crystal-multi	**52.35**†	64.36	82.08†	27.16†⋆	50.70	49.23		40.55†⋆
CFG (SVO)-multi	**51.05**	63.68	79.35	26.51	51.61†⋆	50.00†⋆		35.12†

Table 13: Multilingual zero-shot evaluation results for Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom) for single-source stage 1 training. Models appear in order of highest multilingual zeroshot average (top within each language) to lowest. Single-source stage 1 models that outperform all other models in this table and those in Table 1 are underlined. Those that beat or tie with the random init baselines (see Table 1 for random init values) are marked with † for `random init (70M)` and ⋆ for `random init (100M)`.

Multilingual Model (NL Evals)	Avg	BLiMP(NL)	MultiBLiMP(NL)	HellaSwag(NL)	Winogrande(NL)	XStoryCloze(NL)	XCOMPS(NL)	GlobalPIQA(NL)	Piqa-par(NL)	Piqa-nonpar(NL)
sharedtok CFG-multi	**0.5305**†	0.7272	0.8683	0.2600	0.4979⋆	0.4729	0.5155†	0.3718†	0.2136†	0.53†⋆
sharedtok protein-multi	**0.5371**†	0.7262	0.8726†	0.2601	0.4864	0.4853†⋆	0.5179†	0.4112†⋆	0.2524†	0.57†⋆
sharedtok crystal-multi	**0.5334**†	0.7525	0.8764†	0.2605	0.4938	0.4807⋆	0.5227†⋆	0.3474†	0.1748	0.52†
sharedtok shuffle-dyck-multi	**0.5275**	0.7315	0.8687	0.2573	0.5095†⋆	0.4752	0.5182†	0.3323†	0.1845	0.48
sharedtok random init 70M-multi	**0.5277**	0.7351	0.8713	0.2644	0.5012	0.4814	0.5134	0.3273	0.1845	0.47
sharedtok random init 100M-multi	**0.5386**†	0.7362	0.8820	0.2618	0.4971	0.4783	0.5189	0.3959	0.2718	0.52
sharedtok MissUniverse-c-p-cr	**0.5299**	0.7293	0.8593	0.2609	0.4889	0.4745	0.5243†⋆	0.3721†	0.1942†	0.55†⋆
Multilingual Model (ZH Evals)	**Avg**	**BLiMP(ZH)**		**HellaSwag(ZH)**	**Winogrande(ZH)**	**XStoryCloze(ZH)**	**XCOMPS(ZH)**	**GlobalPIQA(ZH)**	**Piqa-par(ZH)**	**Piqa-nonpar(ZH)**
sharedtok CFG-multi	**0.4678**	0.7092		0.2654†⋆	0.4897⋆	0.4721	0.5284†	0.3418†⋆	0.2136⋆	0.47†⋆
sharedtok protein-multi	**0.4786**†⋆	0.7516†⋆		0.2671†⋆	0.5021⋆	0.4652	0.5244†	0.3612†⋆	0.2524†⋆	0.47†⋆
sharedtok crystal-multi	**0.4750**†⋆	0.7225		0.2698†⋆	0.4955⋆	0.4628	0.5331†	0.3662†⋆	0.2524†⋆	0.48†⋆
sharedtok shuffle-dyck-multi	**0.4615**	0.6936		0.2666†⋆	0.4913⋆	0.4667	0.5188	0.3318†⋆	0.2136⋆	0.45†⋆
sharedtok random init 70M-multi	**0.4713**	0.7231		0.2659	0.5070	0.4776	0.5226	0.3314	0.2427	0.42
sharedtok random init 100M-multi	**0.4721**	0.7445		0.2651	0.4889	0.4807	0.5415	0.3120	0.2039	0.42
sharedtok MissUniverse-c-p-cr	**0.4703**	0.7297†		0.2651	0.4823	0.4737	0.5189	0.3521†⋆	0.1942	0.51†⋆
Multilingual Model (EN Evals)	**Avg**	**BLiMP(EN)**	**MultiBLiMP(EN)**	**HellaSwag(EN)**	**Winogrande(EN)**	**XStoryCloze(EN)**		**GlobalPIQA(EN)**	**Piqa-par(EN)**	**Piqa-nonpar(EN)**
sharedtok CFG-multi	**0.5185**	0.6573	0.8208	0.2664†⋆	0.5004†	0.4892		0.3771	0.1942	0.56†⋆
sharedtok protein-multi	**0.5217**	0.6692†	0.8221	0.2629	0.4889	0.4961⋆		0.3911⋆	0.2621	0.52⋆
sharedtok crystal-multi	**0.5207**	0.6474	0.8247	0.2677†⋆	0.5054†⋆	0.4977⋆		0.3814	0.2427	0.52⋆
sharedtok shuffle-dyck-multi	**0.5167**	0.6387	0.8065	0.2647†⋆	0.4979†	0.4961⋆		0.3961⋆	0.2621	0.53⋆
sharedtok random init 70M-multi	**0.5260**	0.6633	0.8299	0.2646	0.4930	0.5039		0.4011	0.2621	0.54
sharedtok random init 100M-multi	**0.5270**	0.6711	0.8390	0.2644	0.5037	0.4930		0.3908	0.2816	0.50
sharedtok MissUniverse-c-p-cr	**0.5159**	0.6486	0.8272	0.2651†⋆	0.4938†	0.4992⋆		0.3614	0.2427	0.48

Table 14: Shared tokenizer evaluation results for multilingual zero-shot evals in Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom). Averages are calculated from the listed evals. Best performers are underlined, and those that beat `random init` are marked with a ⋆. Note: these results do not include the finetuning eval tasks.

Multilingual Model (ZH Evals)	Avg	BLiMP(ZH)		HellaSwag(ZH)	Winogrande(ZH)	XStoryCloze(ZH)	XCOMPS(ZH)	GlobalPIQA(ZH)	Piqa-par(ZH)	Piqa-nonpar(ZH)
random init 70M-ZH	**0.4678**	0.7172		0.2630	0.4897⋆	0.4768⋆	0.5228	0.3373	0.1845	0.49⋆
random init 100M-ZH	**0.4703**†	0.7241†		0.2671†	0.4798	0.4737	0.5348†	0.3420†	0.2039†	0.48
CFG (SOV & SVO) 70M-ZH	**0.4725**†⋆	0.6982		0.2690†⋆	0.4946†⋆	0.4930†⋆	0.5193	0.3611†⋆	0.2621†⋆	0.46
CFG (SOV & SVO) 100M-ZH	**0.4758**†⋆	0.7284†⋆		0.2723†⋆	0.5078†⋆	0.4760†	0.5186	0.3517†⋆	0.2233†⋆	0.48⋆
Multilingual Model (EN Evals)	**Avg**	**BLiMP(EN)**	**MultiBLimP(EN)**	**HellaSwag(EN)**	**Winogrande(EN)**	**XStoryCloze(EN)**		**GlobalPIQA(EN)**	**Piqa-par(EN)**	**Piqa-nonpar(EN)**
rand init 70M-EN	**0.5074**⋆	0.6142	0.8013⋆	0.2702⋆	0.5062⋆	0.4814		0.3711⋆	0.2621⋆	0.48⋆
rand init 100M-EN	**0.5068**	0.6342†	0.7987	0.2684	0.4979	0.4853†		0.3562	0.2524	0.46
CFG (SOV & SVO) 70M-EN	**0.5063**	0.6345†⋆	0.8013†⋆	0.2655	0.4889	0.4915†⋆		0.3562⋆	0.2524⋆	0.46⋆
CFG (SOV & SVO) 100M-EN	**0.5112**†⋆	0.6429†⋆	0.8130†⋆	0.2696⋆	0.4979⋆	0.4985†⋆		0.3455	0.3010†⋆	0.39
Multilingual Model (NL Evals)	**Avg**	**BLiMP(NL)**	**MultiBLiMP(NL)**	**HellaSwag(NL)**	**Winogrande(NL)**	**XStoryCloze(NL)**	**XCOMPS(NL)**	**GlobalPIQA(NL)**	**Piqa-par(NL)**	**Piqa-nonpar(NL)**
random init 70M-NL	**0.5297**	0.7129⋆	0.8366	0.2644	0.4938	0.4899	0.5040	0.4061	0.2621⋆	0.55
random init 100M-NL	**0.5376**†	0.7078	0.8494†	0.2661†	0.5161†	0.4923†	0.5155†	0.4161†	0.2621†	0.57†
CFG (SOV & SVO) 70M-NL	**0.5264**	0.7004	0.8250	0.2640	0.5103†	0.4876	0.5015	0.3961	0.2621†⋆	0.53
CFG (SOV & SVO) 100M-NL	**0.5282**	0.7116⋆	0.8460†	0.2679†⋆	0.4782	0.4992†⋆	0.4989	0.3954	0.3107†⋆	0.48
Multilingual Model (FA Evals)	**Avg**		**MultiBLiMP(FA)**	**HellaSwag(FA)**	**Winogrande(FA)**	**XStoryCloze(FA)**	**XCOMPS(FA)**	**GlobalPIQA(FA)**	**Piqa-par(FA)**	**Piqa-nonpar(FA)**
random init 70M-FA	**0.4755**⋆		0.6753	0.2642	0.4901	0.5170⋆	0.5207⋆	0.3859⋆	0.2718⋆	0.50⋆
random init 100M-FA	**0.4734**		0.6792†	0.2669†	0.5033†	0.5139	0.5207†	0.3567	0.2233	0.49
CFG (SOV & SVO) 70M-FA	**0.4705**		0.6784†	0.2637	0.4942	0.5178	0.5232†⋆	0.3459	0.2718†⋆	0.42
CFG (SOV & SVO) 100M-FA	**0.4806**†⋆		0.6976†⋆	0.2663†	0.5058†⋆	0.5209⋆	0.5070	0.3858⋆	0.2816†⋆	0.49†⋆

Table 15: Multilingual zero-shot evaluation results for Dutch (NL, top), Chinese (ZH), English (EN), and additional evals for Persian (FA, bottom) using SOV-SVO stage 1 mix and 70M tokens of natural language as well as 100M to accommodate for the fourth language. Best performers are underlined, and those that beat `random init (70M)` are marked with a † while `random init (100M)` winners are noted with ⋆. Note: these results do not include the finetuning eval tasks.

Model (Monolingual ZH)	Avg	BLiMP(ZH)		HellaSwag(ZH)	Winogrande(ZH)	XStoryCloze(ZH)	XCOMPS(ZH)	GlobalPIQA(ZH)	Piqa-par(ZH)	Piqa-nonpar(ZH)
random init 100M-ZH	**0.4889**	0.7765		0.2749	0.4988	0.4853	0.5423	0.3557	0.2913⋆	0.42
CFG+protein+crystal-ZH	**0.4885**	0.7712		0.2749⋆	0.4806	0.5309⋆	0.4915	0.3818⋆	0.2136	0.55⋆
CFG+protein-ZH	**0.4872**	0.7756		0.2759⋆	0.4674	0.4814	0.5520⋆	0.3707⋆	0.2913⋆	0.45⋆
protein+crystal-ZH	**0.4755**	0.7880⋆		0.2721	0.4930	0.4915⋆	0.5506⋆	0.3667⋆	0.2233	0.51⋆
CFG+crystal-ZH	**0.4966**⋆	0.7948⋆		0.2716	0.4922	0.4930⋆	0.5416	0.3862⋆	0.2524	0.52⋆
CFG-ZH	**0.4929**⋆	0.7557		0.2759⋆	0.5128⋆	0.4892⋆	0.5528⋆	0.3714⋆	0.2427	0.50⋆
protein-ZH	**0.4816**	0.7777⋆		0.2694	0.4918	0.4807	0.5434⋆	0.3268	0.2136	0.44⋆
crystal-ZH	**0.4911**⋆	0.7781⋆		0.2729	0.5012⋆	0.4884	0.5443⋆	0.3614⋆	0.2427	0.48⋆
shuffle-dyck-ZH	**0.4929**⋆	0.7951⋆		0.2695	0.4926	0.4845	0.5496⋆	0.3659⋆	0.2718	0.46⋆
Model (Monolingual EN)	**Avg**	**BLiMP(EN)**	**MultiBLimP(EN)**	**HellaSwag(EN)**	**Winogrande(EN)**	**XStoryCloze(EN)**		**GlobalPIQA(EN)**	**Piqa-par(EN)**	**Piqa-nonpar(EN)**
rand init 100M-EN	**0.5212**	0.6598	0.8351	0.2672	0.5144	0.5046		0.3462	0.2524	0.44
CFG+protein+crystal-EN	**0.5180**	0.6665⋆	0.8247	0.2642	0.4889	0.5031		0.3608⋆	0.2816⋆	0.44⋆
CFG+protein-EN	**0.5200**	0.6610⋆	0.8364⋆	0.2643	0.4880	0.5093⋆		0.3612⋆	0.2524⋆	0.47⋆
protein+crystal-EN	**0.5203**	0.6643⋆	0.8312	0.2638	0.4872	0.5039		0.3712⋆	0.2524⋆	0.49⋆
CFG+crystal-EN	**0.5250**⋆	0.6752⋆	0.8299	0.2642	0.4988	0.5062⋆		0.3759⋆	0.2718⋆	0.48⋆
CFG-EN	**0.5230**⋆	0.6590	0.8390⋆	0.2681⋆	0.5128	0.4930		0.3661⋆	0.2621⋆	0.47⋆
protein-EN	**0.5262**⋆	0.6654⋆	0.8325	0.2676⋆	0.4979	0.5031		0.3907⋆	0.2913⋆	0.49⋆
crystal-EN	**0.5198**	0.6554	0.8351⋆	0.2636	0.5128	0.5054⋆		0.3464⋆	0.2427	0.45⋆
shuffle-dyck-EN	**0.5195**	0.6619⋆	0.8208	0.2636	0.5004	0.5138⋆		0.3562⋆	0.2524⋆	0.46⋆
Model (Monolingual NL)	**Avg**	**BLiMP(NL)**	**MultiBLiMP(NL)**	**HellaSwag(NL)**	**Winogrande(NL)**	**XStoryCloze(NL)**	**XCOMPS(NL)**	**GlobalPIQA(NL)**	**Piqa-par(NL)**	**Piqa-nonpar(NL)**
random init 100M-NL	**0.5532**	0.7711	0.9091	0.2674	0.5161	0.4930	0.5187	0.3970	0.2039	0.59
CFG+protein+crystal-NL	**0.5614**⋆	0.7977⋆	0.8945	0.2666	0.4839	0.5101⋆	0.5317⋆	0.4452⋆	0.3204⋆	0.57
CFG+protein-NL	**0.5474**	0.7727⋆	0.8953	0.2672	0.4856	0.4954⋆	0.5342⋆	0.3811	0.2621⋆	0.5
protein+crystal-NL	**0.5614**⋆	0.7786⋆	0.9163⋆	0.2651	0.4872	0.4992⋆	0.5374⋆	0.4458⋆	0.2816⋆	0.61⋆
CFG+crystal-NL	0.5571⋆	0.7818⋆	0.9232⋆	0.2686⋆	0.4856	0.4985⋆	0.5108	0.4309⋆	0.2718⋆	0.59⋆
CFG-NL	**0.5408**	0.7371	0.9005	0.2665	0.5021	0.4822	0.5155	0.3818	0.2136⋆	0.55
CFG (SOV)-NL	**0.5442**	0.7631	0.9099⋆	0.2679⋆	0.4889	0.4930⋆	0.5153	0.3714	0.2427⋆	0.50
CFG (SOV & SVO)-NL	**0.5577**⋆	0.7945⋆	0.9245⋆	0.2712⋆	0.4740	0.5023⋆	0.5210⋆	0.4164⋆	0.2427⋆	0.59⋆
protein-NL	**0.5442**	0.7416	0.8846	0.2682⋆	0.4930	0.4830	0.5137	0.4254⋆	0.3107⋆	0.54
crystal-NL	**0.5580**⋆	0.7832⋆	0.9215⋆	0.2672	0.5054	0.4907	0.5309⋆	0.4068⋆	0.2136⋆	0.60⋆
shuffle-dyck-NL	**0.5620**⋆	0.7842⋆	0.9224⋆	0.2676⋆	0.5103	0.4938⋆	0.5101	0.4457⋆	0.2913⋆	0.60⋆
Model (Monolingual FA)	**Avg**		**MultiBLiMP(FA)**	**HellaSwag(FA)**	**Winogrande(FA)**	**XStoryCloze(FA)**	**XCOMPS(FA)**	**GlobalPIQA(FA)**	**Piqa-par(FA)**	**Piqa-nonpar(FA)**
random init 100M-FA	**0.5011**		0.7634	0.2691	0.5198	0.5070	0.5455	0.4017	0.2233	0.58⋆
CFG+protein+crystal-FA	**0.4910**		0.7716⋆	0.2661	0.5049	0.5054	0.5259	0.3720	0.2039	0.54
CFG+protein-FA	**0.4956**		0.7720⋆	0.2650	0.4934	0.5147⋆	0.5370	0.3914	0.2427⋆	0.54
protein+crystal-FA	**0.4982**		0.7744⋆	0.2680	0.5082	0.4954	0.5369	0.4061⋆	0.2621⋆	0.55
CFG+crystal-FA	**0.4976**		0.7611	0.2679	0.4810	0.5132⋆	0.5367	0.4259⋆	0.2718⋆	0.58⋆
CFG-FA	**0.4947**		0.7822⋆	0.2707⋆	0.4852	0.5163⋆	0.5370	0.3767	0.2233⋆	0.53
CFG (SOV)-FA	**0.4849**		0.7630	0.2655	0.4860	0.4985	0.5347	0.3618	0.2136	0.51
CFG (SOV & SVO)-FA	**0.4877**		0.7466	0.2666	0.4909	0.5101⋆	0.5301	0.3816	0.2331⋆	0.53
protein-FA	**0.4893**		0.7262	0.2697⋆	0.5247⋆	0.5147⋆	0.5289	0.3718	0.2136	0.53
crystal-FA	**0.5034**⋆		0.7779⋆	0.2690	0.5148	0.5054	0.5427	0.4109⋆	0.2718⋆	0.55

Table 16: Zero-shot evaluation results for monolingual models trained on Chinese (ZH), English (EN), Dutch (NL), or Persian/Farsi (FA) in stage 2. Best models are *underlined*, and those that beat or tie with the random init baseline are marked with a ⋆.

Model (Monolingual NL)	Avg	BLiMP(NL)	MultiBLiMP(NL)	HellaSwag(NL)	Winogrande(NL)	XStoryCloze(NL)	XCOMPS(NL)	GlobalPIQA(NL)	Piqa-par(NL)	Piqa-nonpar(NL)
sharedtok random init 70M-NL	**55.17**	78.92⋆	92.11	26.77⋆	49.55⋆	49.23⋆	51.41	38.17	22.33	54.00
sharedtok random init 100M-NL	**55.37**†	78.46	92.49†	26.60	48.97	48.07	52.92†	40.11†	26.21†	54.00
sharedtok CFG+protein+crystal-NL	**55.34**†	78.32	92.02	26.58	48.89⋆	49.15⋆	52.21†	40.18†⋆	21.36	59.00†⋆
sharedtok CFG-NL	**55.80**†⋆	78.47⋆	92.45†	26.61⋆	49.46⋆	48.92⋆	53.51†⋆	41.17†⋆	22.33†	60.00†⋆
sharedtok protein-NL	**55.18**†	78.74⋆	92.28†	26.51	48.97⋆	49.69†⋆	51.86†	38.23†	18.45	58.00†⋆
sharedtok crystal-NL	**54.92**	78.28	92.75†⋆	26.38	49.96†⋆	48.61⋆	51.75†	36.68	21.36	52.00
sharedtok shuffle-dyck-NL	**55.02**	78.76⋆	92.11†	26.36	49.96†⋆	49.54†⋆	52.16†	36.24	17.48	55.00†⋆
Model (Monolingual ZH)	**Avg**	**BLiMP(ZH)**		**HellaSwag(ZH)**	**Winogrande(ZH)**	**XStoryCloze(ZH)**	**XCOMPS(ZH)**	**GlobalPIQA(ZH)**	**Piqa-par(ZH)**	**Piqa-nonpar(ZH)**
sharedtok random init 70M-ZH	**48.26**	77.14		26.96	47.65	48.07	54.56⋆	35.20⋆	20.39	50.00⋆
sharedtok random init 100M-ZH	**48.52**†	78.13†		27.17†	48.89†	48.76†	53.52	34.62	25.24†	44.00
sharedtok CFG+protein+crystal-ZH	**47.52**	76.95		27.04†	47.57	47.29	52.60	33.68	21.36†	46.00⋆
sharedtok CFG-ZH	**48.65**†⋆	77.29†		27.01	49.96†⋆	47.76	54.28⋆	35.62†⋆	25.24†⋆	46.00⋆
sharedtok protein-ZH	**48.45**†	78.29†⋆		27.08†	47.57	48.14†	53.96⋆	35.68†⋆	21.36†	50.00†⋆
sharedtok crystal-ZH	**48.94**†⋆	77.23†		27.04†	49.96†⋆	48.99†⋆	53.78	36.65†⋆	23.30†	50.00†⋆
sharedtok shuffle-dyck-ZH	**48.26**†	78.19†⋆		27.38†⋆	47.57	47.45	54.29⋆	34.68⋆	21.36†	48.00⋆
Model (Monolingual)	**Avg**	**BLiMP(EN)**	**MultiBLimP(EN)**	**HellaSwag(EN)**	**Winogrande(EN)**	**XStoryCloze(EN)**		**GlobalPIQA(EN)**	**Piqa-par(EN)**	**Piqa-nonpar(EN)**
sharedtok rand init 70M-EN	**52.83**	68.29	84.55⋆	26.47	49.55	50.54⋆		37.61	26.21	49.00
sharedtok rand init 100M-EN	**53.38**†	69.08†	83.90	26.75†	50.62†	49.85		40.08†	28.16†	52.00†
sharedtok CFG+protein+crystal-EN	**52.48**†	68.02	83.77	26.40	50.70†⋆	48.37		37.61†	26.21†	49.00†
sharedtok CFG-EN	**51.33**	66.04	83.51	26.54†	51.85†⋆	49.30		30.73	18.45	43.00
sharedtok protein-EN	**52.16**	67.45	81.56	26.28	49.88†	49.69		38.11†	26.21†	50.00†
sharedtok crystal-EN	**52.81**	66.94	84.16⋆	26.15	49.63†	50.39⋆		39.61†	26.21†	53.00†⋆
sharedtok shuffle-dyck-EN	**53.12**†	66.91	86.10†⋆	26.87†⋆	50.62†⋆	50.15⋆		38.08†	28.16†⋆	48.00

Table 17: Shared tokenizer monolingual zero-shot evaluation results for Dutch (NL, top), Chinese (ZH), English (EN). Best performers are underlined, and those that beat `random init (70M)` are marked with a † while those that beat `random init (100M)` are marked with a ⋆.

Model *(NL Evals)*	**Avg**	**ARC(NL)**	**Belebele(NL)**	**BMLama(NL)**	**MNLI(NL)**	**Sib200(NL)**	**TruthfulQA(NL)**	**Include(NL)**	**POS(NL)**
MissUniverse-p-cr	**38.27**	26.88	19.89	13.74	40.48	64.50	22.32	25.89	92.50
MissUniverse-c-cr	**38.09**	24.79	20.45	16.80	42.85	62.00	20.54	25.00	92.25
MissUniverse-c-p-cr	**37.18**	25.83	21.02	12.67	45.95	52.5	16.96	31.25	91.26
MissUniverse-c-p	**36.85**	25.83	20.45	17.47	43.92	48.00	17.86	29.46	91.84
Model *(ZH Evals)*	**Avg**	**ARC(ZH)**	**Belebele(ZH)**	**BMLama(ZH)**	**MNLI(ZH)**	**Sib200(ZH)**	**TruthfulQA(ZH)**	**Include(ZH)**	**POS(ZH)**
MissUniverse-p-cr	**40.51**	22.71	26.14	17.14	46.96	72.00	27.68	15.18	89.02
MissUniverse-c-cr	**39.70**	23.96	21.02	13.41	48.48	70.00	28.57	23.21	88.94
MissUniverse-c-p-cr	**40.61**	27.08	23.30	13.08	45.38	75.00	20.54	28.57	87.88
MissUniverse-c-p	**39.86**	24.17	25.57	13.99	46.06	66.00	25.00	25.89	88.71
Model *(EN Evals)*	**Avg**	**ARC(EN)**	**Belebele(EN)**	**BMLama(EN)**	**MNLI(EN)**	**Sib200(EN)**	**TruthfulQA(EN)**		**POS(EN)**
MissUniverse-p-cr	**42.98**	21.46	21.02	19.87	44.99	66.00	27.68		90.32
MissUniverse-c-cr	**42.10**	26.25	21.02	17.38	44.65	72.00	26.79		90.14
MissUniverse-c-p-cr	**42.64**	26.88	21.02	13.16	44.09	62.50	38.39		89.31
MissUniverse-c-p	**41.94**	24.58	24.43	18.96	46.57	62.50	24.11		89.95

Table 18: Multilingual finetuning evaluation results for Dutch (NL, top), Chinese (ZH, middle), and English (EN, bottom). Averages are calculated from the listed evals. Best performers are underlined, and those that beat `random init` are marked with a ⋆.

From Puzzle-Solver to Language Model: Recurrent Inductive Bias, Data Scale, and Benchmark Validity in BabyLM

Patrick Haller **Filipe Laitenberger** **Jonas Golde** **Alan Akbik**
Humboldt-Universität zu Berlin
patrick.haller.1@hu-berlin.de

Abstract

In the small-data regime of the BabyLM Challenge, a model's architecture still shapes what it learns. We ask how much an inductive bias toward recurrence helps, and whether current benchmarks can detect that help. Our model adapts the Hierarchical Reasoning Model, a small recurrent puzzle-solver that reuses one shared block many times per token instead of stacking more layers. We make three changes to turn it into a language model: a sequence-mixing layer that can carry order-dependent state, a gate on one of the two connections between its modules, and a tunable number of recurrent cycles. The result matches a parameter-matched GPT-2, and we use it to study three questions. **Q1:** recurrence buys depth without adding parameters. On a controlled state-tracking task, running more cycles of the same weights tracks longer sequences, even when the extra cycles are added only at inference. **Q2:** the advantage is largest where data is scarcest. HRM-rec leads GPT-2 on BLiMP at 10M words, ties at 100M, and, trained to convergence, still leads at 3B tokens (out of budget, where data and tokenizer also change). **Q3:** the benchmark meant to reward state-tracking cannot see it. A null model that never reads the puzzle outscores every trained model on `entity_tracking`, because 28.5% of its items reveal the answer in the option list and every trained model is at chance on the rest. Gains on that task measure how well a model has learned the frequency of one answer. Because the suite misses a capability we can demonstrate directly, we recommend pairing benchmarks with controlled architectural probes.

1 Introduction

The BabyLM Challenge (Warstadt et al., 2023; Charpentier et al., 2025) asks how far language models can be pushed under a developmentally plausible data budget. Many entries focus on maximizing benchmark scores. We use the setting differently: to measure how much architectural inductive bias helps at small data scale and whether current benchmarks can detect it. We adapt a recurrent puzzle-solver into a competitive language model, then use it to study three questions about inductive bias, data scale, and benchmark validity.

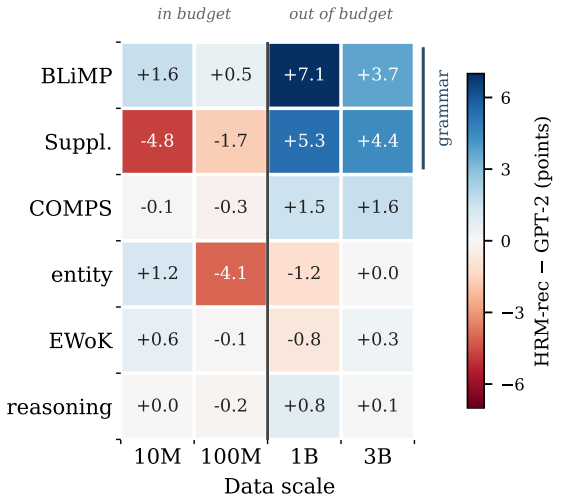


Figure 1: Per-task gap between HRM-rec and GPT-2 (points, HRM-rec minus GPT-2) on the zero-shot tasks across data scale, with both models trained to convergence (parameter-matched; caveats in §4). The durable advantage is on BLiMP. The reasoning-flavored tasks (entity, EWoK, reasoning suite) sit near zero.

Our starting point is the Hierarchical Reasoning Model (Wang et al., 2025), HRM for short, a 27M-parameter network built for symbolic puzzles such as Sudoku. Instead of stacking layers, HRM applies one small shared block many times per token, so its depth comes from recurrence rather than from parameters. It has two copies of that block. A fast module runs several times in a row, then a slow module reads the result and runs once, and this outer loop repeats. We call one pass through the loop a cycle and write L_{cyc} for the number of fast passes per outer step. Only the final pass receives a gradient. §2 gives the details.

Proceedings of the Second BabyLM Workshop, pages 722–734
October 29, 2026 ©2026 Association for Computational Linguistics

HRM was built for puzzles, and, like Wang et al. (2026), we adapt it to language. We make three changes (§3). We replace its attention layers with a delta-rule mixer, a sequence-mixing layer that can carry order-dependent state (Yang et al., 2024, 2025). We gate the hand-off from the slow module to the fast one, and only that direction. And we tune the number of recurrent cycles. Together these raise BLiMP from 60.0 to 68.6 at 10M words and bring the model level with a parameter-matched GPT-2 (18 layers, ∼140M). The directional gate is the main architectural detail: gating only the slow→fast hand-off adds 2.5 BLiMP points over gating both directions (§5.1). We treat HRM-rec's extra compute per token as part of the inductive bias under study. We then use the model to study three questions:

1. **Q1: recurrence as free depth.** Tracking state across a sequence of order-dependent operations needs a depth that grows with the sequence. On a controlled state-tracking task, unrolling the recurrence for more cycles extends the length over which HRM-rec tracks state, at no parameter cost. Because the cycle count is set at inference, the same weights track longer sequences when given more cycles. A fixed-depth transformer has no equivalent setting (§5.2).

2. **Q2: inductive bias across data scale.** We compare against GPT-2 at four data scales, two inside the BabyLM budget (10M and 100M words) and two outside it (1B and 3B tokens, where data and tokenizer also change). The BLiMP advantage peaks at 10M, becomes a tie at 100M, and, once both models are trained to convergence, persists out to 3B tokens (§5.3, Fig. 1).

3. **Q3: architecture as a benchmark audit.** `entity_tracking` is the one BabyLM task that targets state-tracking. We show that most of its score comes from items whose option list reveals the answer, and that every model is at chance on the rest. Gains on it reflect how well a model has learned the frequency of one answer, which we call calibration (§5.4). This extends earlier critiques of the BabyLM evaluation pipeline (Păpuşoi and Nisioi, 2025; Hu et al., 2024), and we pair the audit with the controlled probe of Q1 as a general validity check.

Together these place the recurrent bias where data is scarcest, and show that its clearest benefit (BLiMP) is not the one the state-tracking benchmark reports. This runs against the expectation from the bitter lesson (Sutton, 2019) that generic architectures win. We also report a negative control: mixing synthetic reasoning traces (PleIAs and AI Alliance, 2025) into a fixed word budget lowers BLiMP and improves nothing else (§5.6). Code, configs, and the scaling harness are released.

2 Background and Related Work

Recurrence and weight sharing. Recurrence and weight sharing (Lan et al., 2020; Dehghani et al., 2019) trade parameters for compute. Adaptive computation (Graves, 2016; Banino et al., 2021), gated recurrence (Parisotto et al., 2020), and looped transformers (Giannou et al., 2023; Schwarzschild et al., 2021) all iterate a shared block. The looped-transformer work shows that recurrent depth drives length extrapolation, which is the closest prior result to our finding that more cycles extend the sequence length over which the model tracks state (§5.2). Within BabyLM, gated-linear RNNs match or beat transformers at 10–100M words (Haller et al., 2024), as do linear-attention mixers (Haller et al., 2025) and linguistically gated RNNs (Chesi et al., 2024). We add a hierarchical-recurrent model, a mechanistic account of how recurrence extends state-tracking, and evidence on how its BLiMP advantage varies with data scale.

The Hierarchical Reasoning Model. Distinct from that RNN lineage, HRM (Wang et al., 2025) is a ∼27M-parameter recurrent reasoner that solves algorithmic puzzles (Sudoku, maze navigation, ARC) on which much larger chain-of-thought transformers stumble. It couples two weight-shared transformer modules at different timescales: a fast module L with hidden state z_L and a slow module H with hidden state z_H. In each outer cycle the fast module updates L_{cyc} times against a fixed z_H. The slow module then reads the settled fast state and updates once. There are H_{cyc} such outer cycles per token, so the unrolled network is many times deeper than its layer count. HRM trains with a one-step gradient: only the final inner step is backpropagated through, and a warmup grows this backpropagated depth over training. The original modules use softmax attention and pass state to each other by plain addition in both directions. We

revisit both choices at BabyLM scale (§3). Reproductions have questioned how much of HRM's puzzle results comes from the H/L hierarchy rather than from its refinement loop and task augmentation, so we isolate the hierarchy with an ablation (§5.1) and a synthetic probe (§5.2).

Delta-rule mixers and state tracking. A sequence mixer is the layer that moves information between positions, whether softmax attention or a linear-time recurrent alternative. Diagonal state-space models and vanilla linear attention provably cannot track non-commutative state, that is, quantities whose value depends on the order of operations, such as the result of a sequence of swaps (Gu and Dao, 2024; Dao and Gu, 2024; Merrill et al., 2024; Liu et al., 2023). Delta-rule mixers do not have this limitation. DeltaNet (Yang et al., 2024) uses a per-step state transition that is a generalized Householder reflection, and its gated variant adds a decay term (Yang et al., 2025). Allowing negative eigenvalues in that transition unlocks parity and simple permutations (Grazzi et al., 2025). That line of work characterizes what a single mixing layer can represent when trained directly on the formal task. We use these mixers inside the H/L modules via flash-linear-attention (Yang and Zhang, 2024) and show that recurrent depth also extends tracking length, on top of a fixed mixer.

Inductive bias versus scale. The bitter lesson (Sutton, 2019) and scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022) characterize the large-data limit where generic architectures win. At the small-data end the most reliable BabyLM wins have been architectural (ELC-BERT (Charpentier and Samuel, 2023), GPT-BERT (Charpentier and Samuel, 2024)), while curriculum and data-ordering tricks have repeatedly underdelivered (Warstadt et al., 2023). Our negative SYNTH control (§5.6) fits that pattern. Our question is orthogonal to leaderboard position: how the value of an architectural prior changes with budget, and whether the evaluation protocol can measure it.

Evaluation validity. Benchmarks can reward spurious signal (Gururangan et al., 2018; Bowman and Dahl, 2021). Partial-input baselines answer multiple-choice questions without reading the question (Balepur et al., 2024), and scoring options by raw summed log-probability favors short options (Brown et al., 2020; Gao, 2021). Both are directly relevant to the BabyLM zero-shot tasks, which compare log-probabilities of candidate strings (Warstadt et al., 2020; Kim and Schuster, 2023; Gao et al., 2024). At developmental scale, Păpuşoi and Nisioi (2025) show that a model emitting identical logits for every input scores maximally, not at chance, on BLiMP, and Hu et al. (2024) trace inflated EWoK scores to a fixed label assigned on probability ties (since randomized). We extend these critiques to the previously unaudited `entity_tracking` with a puzzle-blind null model and a score decomposition that exposes a choices-only giveaway (§5.4).

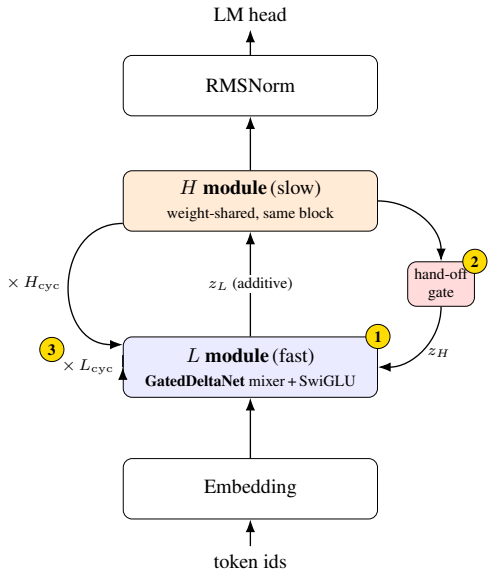


Figure 2: HRM-rec. Two weight-shared transformer stacks, a fast module L and a slow module H: L iterates L_{cyc} times against a fixed H state, then H updates once, for H_{cyc} outer cycles under a truncated one-step gradient. Circled numbers mark our three modifications to HRM: ① a **GatedDeltaNet** mixer replaces softmax attention; ② a *directional* slow→fast **hand-off gate** (red) replaces the symmetric additive injection (the fast→slow hand-off z_L stays additive); ③ the fast-cycle **recurrence depth** L_{cyc} becomes a tunable setting.

3 Model: HRM-rec

HRM-rec uses a GPT-2-small-sized block (12 layers, d=768, 12 heads), split 6+6 across H and L (Figure 2). Token embeddings enter the fast module L, the two modules exchange state as described in §2, and the slow module's final state feeds the language-model head. We make three changes to turn the puzzle-solver into a language model, described below with their rationale. §5.1 reports the evidence.

- **GatedDeltaNet mixer:** we replace softmax

attention in the H/L modules with GatedDeltaNet. The rationale is the state-tracking expressivity reviewed in §2: delta-rule transitions can represent non-commutative state that diagonal mixers cannot.

- **Directional hand-off gate:** HRM's modules exchange state by plain addition, $z \leftarrow z + \mathrm{inj}$, in both directions. We replace one of these additions with a learned hand-off gate. Writing x for the receiving module's current state and y for the incoming state from the other module, the gate returns a convex combination

$$\mathrm{gate}(x, y) = (1 - g) \odot x + g \odot y, \quad (1)$$

where $g \in [0, 1]^d$ is computed from $[x; y]$ by the GRU-style gate of Parisotto et al. (2020). Its parameters ($\sim$3.5M) are included in the reported count. The gate is directional: only the slow→fast hand-off is gated, so within each inner step the two modules update as

$$z_L \leftarrow L\big(\mathrm{gate}(z_L, z_H)\big), z_H \leftarrow H\big(z_H + z_L\big), \quad (2)$$

with L, H the fast and slow stacks. Keeping the fast→slow direction additive matters: gating it instead hurts (§5.1).

- **Recurrence depth:** in the unmodified HRM, changing the number of fast-module cycles L_{cyc} does not change BLiMP. With the mixer and gate in place it does. L_{cyc} is set at training time and can be changed at inference. §5.1 quantifies the gain and §5.2 explains its mechanism.

4 Experimental Setup

Data and tracks. We adopt the 2025/2026 BabyLM setup (Warstadt et al., 2023; Charpentier et al., 2025): the **Strict** and **Strict-Small** tracks cap training data at **100M** and **10M** words, and own data may be substituted within the budget. The corpus is a detoxified mix of transcribed speech and accessible text (CHILDES, BNC, Gutenberg, OpenSubtitles, etc.); we use it with the official 16,384-token BPE tokenizer at the developmental scales.

Models. Both models share the d=768, 12-head width of §3. The GPT-2 baseline is a standard causal transformer, sized to parameter-match HRM-rec at 18 layers (140.9M parameters). HRM-rec is the final configuration selected by the ablations (§5.1, Table 1): GatedDeltaNet mixer (head dim 128), slow→fast hand-off gate, H_{cyc}=2 and L_{cyc}=5, at 140M parameters. The two models share width, tokenizer, data, and training recipe, so any difference isolates the recurrent bias. For the same reason we compare against a standard causal-LM baseline rather than the strongest leaderboard entry. Higher-scoring BabyLM systems such as GPT-BERT (Charpentier and Samuel, 2024) change the training objective (masked + causal), which would confound the architectural question. Our claims are about the effect of the bias, not about leaderboard rank.

Training. Both models train in bf16 with a cosine learning-rate schedule and warmup, using the optimizer of their reference implementation (AdamW for GPT-2; `adam_atan2` with a weight EMA of 0.999 for HRM-rec, following HRM). Every in-budget run trains for ten epochs over its corpus (100M words seen at Strict-Small, 1B at Strict), our convention rather than a track rule. Sequence length is 256, with an effective batch of 16 at 10M words and 64 at 100M, chosen by a small sweep. For HRM-rec, the depth that receives gradient grows over the first $\sim$20% of steps (the HRM warmup of §2). Full hyperparameters are in App. A.

Out-of-budget scaling. The scaling study (§5.3) departs from the BabyLM budget. It uses a 32,768-token BPE tokenizer trained on FineWeb-Edu (Penedo et al., 2024) and SYNTH (PleIAs and AI Alliance, 2025), sequence length 1024, and a token budget (1B or 3B) rather than an epoch count. Natural runs use FineWeb-Edu, and mixed runs interleave 50% SYNTH at the word level. Both models train with a matched learning rate and a full cosine anneal to convergence.

Evaluation. Models are assessed zero-shot with the official BabyLM pipeline, which scores each item by whether the model assigns higher probability to the correct member of a minimal pair or to the correct continuation. The tasks are **BLiMP** (Warstadt et al., 2020) and **BLiMP-Supplement** (grammar), **COMPS** (Misra et al., 2023) (conceptual property knowledge), **EWoK** (physical and social world knowledge), and **Entity Tracking** (Kim and Schuster, 2023) (discourse state). WUG and Reading are absent from our pipeline version. At the out-of-budget scales we add a reason-

Step	BLiMP
HRM (softmax attn., additive inj.)	60.0 ± 0.5
+ GatedDeltaNet mixer	62.2 ± 0.3
+ gated injection (both directions)	63.6 ± 0.3
+ directional gate (slow→fast)	66.1 ± 0.9
alt: gate fast→slow only	62.7 ± 1.1
+ *tuned learning rate (hyperparameter)*	67.2 ± 0.2
+ recurrence depth (L_{cyc}=5)	$\mathbf{68.6 \pm 0.7}$
−*hierarchy: flat recurrent stack*	55.8 ± 0.3

Table 1: Architecture ablation on Strict-Small (10M words). The cumulative path (single indent) runs from unmodified HRM (60.0) to the final model (68.6), each step separated by non-overlapping error bars. The gating step is *directional*: slow→fast (66.1) beats both-directions (63.6), while the reverse hand-off alone (*alt*, off-path) *hurts* (62.7). The learning-rate step (italics) is a hyperparameter change. The last row removes the H/L hierarchy from the final model at matched parameters and compute (a single recurrent GatedDeltaNet stack), collapsing BLiMP to 55.8.

ing suite (ARC-easy and ARC-challenge, OpenBookQA, HellaSwag, PIQA, CommonsenseQA, WinoGrande) scored by length-normalized accuracy in the LM Evaluation Harness (Gao et al., 2024). The pipeline also fine-tunes each model on subsampled (Super)GLUE (BoolQ, MNLI, QQP, MRPC, MultiRC, RTE, WSC; §5.5).

Comparability. We not that the models are parameter-matched (∼140M each in budget), but HRM-rec spends roughly $H_{\text{cyc}} \times L_{\text{cyc}}$ more compute per token, so a compute-matched comparison would be less favorable to it. *Distribution shift:* the out-of-budget points change data and tokenizer along with scale, so they show only whether the in-budget advantage persists under a different data distribution. All in-budget results (10M/100M) models were trained over three seeds. We report mean and standard deviation. Out-of-budget runs are single-seed for cost.

5 Results

5.1 Architecture Ablation

We ask what each of the three changes of §3 contributes.

Setup. All ablations run on the 10M-word Strict-Small track (BLiMP) and follow a cumulative modification path from the unmodified HRM.

Delta-rule mixing. Replacing softmax attention in the H/L modules with GatedDeltaNet gives +2.2 BLiMP.

Directional gating. Gating both directions with the hand-off gate of Eq. 1 helps (+1.3), but the two directions behave differently (Table 1, *alt* row). Gating the slow→fast hand-off helps, while gating the fast→slow hand-off hurts. Removing the harmful gate therefore yields +2.5 BLiMP over the both-directions design. The fast module benefits from admitting high-level state selectively, while the slow module does better integrating the fast module's output unfiltered. Monea et al. (2026) describe a similar division of labor, in which a state-holding component and a predicting component are kept separate, and the directional gate is a recurrent instance of it.

Recurrence depth. With the mixer and gate in place, the number of fast-module cycles matters. Raising L_{cyc} from 3 to 4 to 5 lifts BLiMP from 67.2 to 67.3 to 68.6. In the unmodified HRM it made no difference. §5.2 explains why.

Removing the hierarchy. The path above adds modifications within the H/L hierarchy. Reproductions of HRM question whether the hierarchy itself matters (§2), so we also train a control without it: a single GatedDeltaNet stack of the same 12 layers, iterated the same number of times with the input added back at each iteration, in the style of the Universal Transformer. This control has the same parameters and the same effective depth as the final model. It reaches only 55.8 ± 0.3 BLiMP, below even the unmodified hierarchical HRM (60.0) and 12.8 points under the final model.[1] It also fits the training data worse throughout training (train perplexity ∼22 vs ∼10), so the flat stack already falls behind during optimization.

5.2 Q1: Recurrence as Free Depth

Q1 asks whether unrolling the weight-shared recurrence buys effective depth: longer state tracking at the parameter cost of a single layer. State-tracking is the natural test for a recurrent model. Following a sequence of order-dependent operations requires a composition depth that grows with the sequence length. HRM's outer loop supplies that depth by applying its shared block again and again. A fixed-depth transformer has to fit the same work into a fixed number of layers.

Setup. The task is to follow five objects being shuffled. Each input token is one permutation of

[1] Stable across learning rates $\{1.5, 3, 6\} \times 10^{-4}$: 55.5–56.7.

model	$L^\star$
bare GatedDeltaNet	24 ± 0
HRM-rec, cyc $(2, 3)$	87 ± 1
HRM-rec, cyc $(2, 5)$	111 ± 6

Table 2: State-tracking on S_5 ($|S_5|$=120): $L^\star$ (train 128, test 512), mean±std over 3 seeds. Cycle counts are $(H_{\text{cyc}}, L_{\text{cyc}})$. With the mixer held fixed, each increase in cycle count extends $L^\star$, at low seed variance.

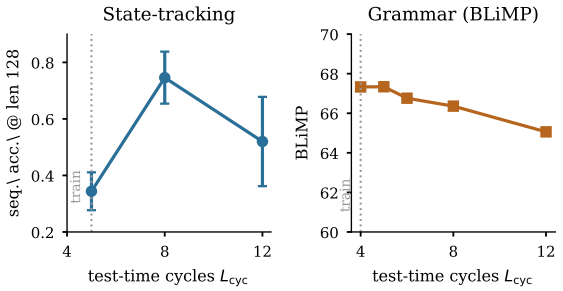


Figure 3: Changing the number of recurrence cycles at test time, on the same weights. *Left*: the state-tracking probe, trained at L_{cyc}=5 (dotted), re-evaluated at more cycles. Sequence accuracy at length 128 (3-seed mean±sd) rises to an optimum and then falls. *Right*: a 10M-word LM trained at L_{cyc}=4, evaluated on BLiMP at L_{cyc}=4/5/6/8/12, scores 67.3/67.3/66.8/66.4/65.1. Extra test-time cycles improve state-tracking and leave BLiMP unchanged.

five positions (a generator of the permutation group S_5), and at every position the model predicts the arrangement that results from applying all permutations so far, in order. Because permutations do not commute, the model has to carry an exact ordered state through the sequence, the bookkeeping a diagonal mixer cannot do. Models train on sequences of length 128 and are tested on sequences of length 512. We report $L^\star$, the longest prefix tracked at floored cumulative accuracy ≥ 0.5: how far into a test sequence the model keeps the running product correct before accuracy decays to chance. Results are means over three seeds. All models use the same GatedDeltaNet mixer, so any gain is attributable to the recurrence. Cycle counts are written as $(H_{\text{cyc}}, L_{\text{cyc}})$.

Results. With the mixer fixed (Table 2), each increase in cycle count extends $L^\star$: 24 for the bare mixer, 87 at cycles $(2, 3)$, and 111 at $(2, 5)$, with non-overlapping seed ranges. Each extra cycle applies the mixer's state transition once more before the prediction, composing one further permutation into the carried state. A deeper stack would do the same work with additional layers. This explains why L_{cyc} affects BLiMP once the mixer is in place (§5.1).

Changing cycles at test time. Larger L_{cyc} allows us to increase the compute after training during inference. We re-evaluate the probe, trained at L_{cyc}=5, at more cycles without retraining (Fig. 3, left). Sequence accuracy at the training length rises from 0.34 to 0.75 at L_{cyc}=8, and length 192, unsolved at the trained depth, reaches 0.24, in all three seeds. Past a length-dependent optimum, further cycles hurt.

State tracking improvements to not convert to the BabyLM grammar benchmarks. A 10M-word LM trained at L_{cyc}=4 holds BLiMP flat as inference cycles rise from 4 to 5 and declines slowly beyond that (Fig. 3, right). We read the decline as a train/test depth mismatch, since the LM was optimized at one depth, and not as evidence about grammar in either direction. Extra inference compute pays off only where the mechanism applies, on state-tracking length, and a fixed-depth transformer such as GPT-2 has no equivalent setting.

This result also separates Q1 from Q2. If the state-tracking the probe measures drove BLiMP, the cycles that extend $L^\star$ should raise BLiMP too, and they do not. We conjecture that both reward effective depth, but why unrolled depth helps BLiMP specifically remains open. Geiping et al. (2025) report the same split in a larger LM, where added recurrent depth helps some capabilities and not others. Whether the formal state-tracking isolated here transfers to natural-language phenomena that plausibly need it, such as long-distance agreement or coreference, is untested here and, to our knowledge, open.

5.3 Q2: The Recurrent Advantage Across Data Scale

Q2 asks how the recurrent advantage changes with data scale.

Setup. We compare HRM-rec (L_{cyc}=5) against GPT-2 at matched parameters across four data scales: in budget (10M/100M words, official corpora and tokenizer) and out of budget (1B/3B tokens, FineWeb-Edu with the 32k tokenizer; §4). Evaluation uses the BabyLM zero-shot suite and, at the larger scales, the normalized-accuracy reasoning suite.

scale	model	BLiMP	gap
10M words	GPT-2	67.0 ± 0.3	
	HRM-rec	$\mathbf{68.6 \pm 0.7}$	$+1.6$
100M words	GPT-2	75.3 ± 1.3	
	HRM-rec	75.8 ± 0.8	$+0.5$
1B tokens	GPT-2	71.3	
	HRM-rec	**78.4**	$+7.1$
3B tokens	GPT-2	76.0	
	HRM-rec	**79.7**	$+3.7$

Table 3: BLiMP accuracy by data scale. *Top two rows* (10M/100M words): BabyLM corpus, 16k tokenizer, 3 seeds. *Bottom two* (1B/3B tokens): out-of-budget, FineWeb-Edu with a 32k tokenizer, trained to convergence (matched LR, full cosine anneal), one representative seed per model (three-seed means for 1B are given in the text). Comparability caveats in §4.

Results. Table 3 traces the BLiMP gap across scale. HRM-rec leads at 10M words ($+1.6$), ties at 100M ($+0.5$), and, once both models are trained to convergence, stays ahead at 1B and 3B tokens. The table lists one seed per model at the out-of-budget points, matching Fig. 1. At 1B, where we ran three seeds per model, the means are HRM-rec 76.6 vs GPT-2 73.5 ($+3.1$) with non-overlapping seed ranges, a smaller gap than the single seed's $+7.1$ but in the same direction.

The advantage is specific to BLiMP. BLiMP tests core morphosyntax, whereas BLiMP-Supplement is dominated by lexical-semantic and pragmatic phenomena (hypernymy, question–answer congruence, turn taking) where extra compositional depth has no obvious role. GPT-2 leads Supplement at both in-budget scales (60.9 vs 56.1 at 10M, 62.2 vs 60.5 at 100M). Supplement is also the noisier set (seed std 1.1 vs 0.7 for BLiMP).

Within every budget this setting can spend, scale does not reverse the recurrent advantage. Any crossover is deferred past that range. Nor does the advantage reduce to added capacity. The directional gate removes parameters yet gains $+2.5$ BLiMP (§5.1). Raising L_{cyc} from 3 to 5 adds no parameters and gains $+1.4$. And at matched parameters the gap shrinks with data, from $+1.6$ at 10M to $+0.5$ at 100M, and at 100M the two models are close on all five in-budget tasks (App. C). The recurrent design improves sample efficiency, and its BLiMP advantage is largest where data is scarcest.

5.4 Q3: What `entity_tracking` Measures

The probe (§5.2) showed directly that recurrence buys state-tracking. `entity_tracking` is the only suite task that claims to measure this ability. . We measure whether in a language modeling setup the same improvements are transferable.

Setup. `entity_tracking` (Kim and Schuster, 2023) describes objects being placed in and moved between numbered boxes, then asks which objects a given box holds. Each item offers a list of candidate answers, and the pipeline scores each candidate string by its log-probability under the model. We build a puzzle-blind null model that never reads the box-state description: if `nothing` is among the options it answers `nothing`, otherwise it guesses uniformly. On the 3,152 items the BabyLM pipeline ships, `nothing` appears only as the correct answer and never as a distractor. So on the 28.5% of items whose answer is `nothing`, its presence in the option list gives the answer away, a choices-only artifact in the sense of Balepur et al. (2024). On the other 71.5%, distractors are token-length-matched to the correct answer by construction. We audit the subset the BabyLM pipeline distributes and do not re-check the original release of Kim and Schuster (2023).

Results. The null model scores an expected 43.3 under the official pipeline, which breaks ties at random, above every trained model in this study. The number follows from the split: the rule is correct on all 28.5% giveaway items and at the 20% chance rate on the rest, for $0.285 \times 100 + 0.715 \times 20 \approx 43$. The best of our runs reaches 41.2. On the length-matched items every trained model, whether GPT-2 or HRM-rec, 16k or 32k tokenizer, at any training stage, near the 20% chance floor (18–25%, Table 4). Nearly all of the benchmark's signal, and all of its run-to-run variance, sits in the giveaway subset. As a result, HRM-rec tracks state better on the probe (§5.2) yet scores worse on the giveaway items (43.0 vs 77.5).

Effect of training. We also compare two GPT-2 checkpoints. One is undertrained: its learning rate was not annealed and its validation perplexity was still falling. It scores 12.8 on `entity_tracking`. The same model trained to convergence scores ∼31. The whole difference is on the giveaway items. The converged model answers `nothing` on 77% of them and the undertrained checkpoint on almost none, while both are

scale	model	"nothing" (29%)	matched (71%)
100M words	GPT-2	77.5	18.1
	HRM-rec	43.0	22.6
3B tokens	GPT-2	46.1	21.9
	HRM-rec	35.0	24.6
null model (no puzzle)		**100.0**	20.0

Table 4: `entity_tracking` accuracy split into the 29% giveaway items (gold is `nothing`, never a distractor) and the 71% whose distractors are length-matched to the gold (harness-faithful scoring). Every model is near the 20% chance floor on the matched items; all score and all cross-model variance come from the giveaway column. HRM-rec, which the probe (§5.2) shows tracks state better, is worse at the giveaway (43 vs 77), so the benchmark ranks it below GPT-2.

at chance on the length-matched 71% (Table 4). Training therefore teaches the model how often `nothing` is the answer. It does not teach it to read the box contents. We refer to this as calibration.

Other checks. With both models trained to convergence, HRM-rec and GPT-2 score the same on `entity_tracking` (30.9 at 3B), and GPT-2 is slightly ahead on the reasoning suite (33.8 vs 32.7, App. E). The reasoning suite contains no state-tracking task and both models are near chance on it, so we do not draw conclusions from it. The scoring rule adds a second problem. The pipeline ranks options by summed token log-probability, which favors short options (§2), and `nothing` is the shortest. Length normalization (Gao, 2021) removes this bias, but `nothing` still appears only as a correct answer, and surface-form effects can remain after normalization (Holtzman et al., 2021). Normalizing the scores is therefore not enough to fix the task.

5.5 Downstream Fine-Tuning

We fine-tune both models to check whether the recurrent architecture hurts downstream performance. The BabyLM pipeline fine-tunes on subsampled (Super)GLUE, and we run it on the 100M-word models. A linear classifier is attached to the final-token representation and the full model is fine-tuned with the pipeline defaults. For HRM-rec we expose the final hidden state of the recurrent backbone through a thin encoder interface. HRM-rec averages 69.5 and GPT-2 68.1 (per-task scores in App. B). The two models are within one point of each other on five of the seven tasks, and each number comes from a single run, so we do not consider the difference reliable. HRM-rec fine-tunes about as well as a standard transformer.

5.6 Synthetic Reasoning Data

HRM was designed for reasoning tasks, so we test whether HRM-rec benefits from training data that contains reasoning traces.

Setup. At a fixed 10M-word budget we replace 25, 50, or 75% of the training data with SYNTH, which consists of query → reasoning → answer traces. One further run trains a 50% mixture to convergence on 3B tokens (§4).

Results. BLiMP decreases as the SYNTH fraction increases (App. D), and no other task improves, whether we use the full reasoning traces or only the question–answer pairs. An earlier run showed an improvement on the question sub-tasks, but it did not replicate. The 3B-token run gives the same result: the 50% mixture has lower BLiMP and no higher reasoning-suite score than the run without SYNTH. The only score that SYNTH raises consistently is `entity_tracking`, which is the task that §5.4 found to reward calibration.

6 Conclusion

We adapted HRM, a recurrent puzzle-solver, into a language model that matches a parameter-matched GPT-2 at BabyLM scale, and used it to study three questions. First, running more recurrent cycles lets the model track state over longer sequences, even when the cycles are added only at inference. Second, HRM-rec has a BLiMP advantage over GPT-2 that is largest at 10M words, and GPT-2 does not overtake it at any scale we trained, up to 3B tokens. Third, `entity_tracking`, the BabyLM task intended to measure state-tracking, mostly measures whether a model has learned how often `nothing` is the answer.

The first two results do not share a mechanism. Extra test-time cycles improve state-tracking but leave BLiMP unchanged (§5.2), so we cannot yet explain why the recurrent model does better on BLiMP. On the architecture side, we found that gating only the slow→fast connection works better than gating both directions (§5.1).

We draw two recommendations for evaluation from the `entity_tracking` analysis. Options that reveal the answer, such as the `nothing` items, should be removed or reported separately. And

when two models are known to differ in some ability, as ours do on the state-tracking probe, a benchmark for that ability should be checked against them.

Limitations

HRM-rec and GPT-2 have the same number of parameters but not the same compute (§4). HRM-rec spends more compute per token because of its recurrence, and a compute-matched comparison would be less favorable to it. The 3B-token results come from a single seed per model. The out-of-budget models have ∼167M parameters, and at 1–3B tokens several tasks in the reasoning suite are near chance, so the absolute reasoning scores carry little information, although the comparison between the two models is consistent across tasks. Validation perplexity cannot be compared between the runs with and without SYNTH, because their held-out sets differ. Our benchmark audit covers a single task. We expect other multiple-choice tasks scored without length normalization to have similar problems, but we have not tested this.

Acknowledgments

We thank all reviewers for their valuable comments. Alan Akbik and Patrick Haller are supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Emmy Noether grant "Eidetic Representations of Natural Language" (project number 448414230). Alan Akbik is furthermore supported under Germany's Excellence Strategy "Science of Intelligence" (EXC 2002/1, project number 390523135). Jonas Golde is supported by the German Federal Ministry of Economic Affairs and Climate Action (BMWK) as part of the project ENA (KK5148001LB0). The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project "Pre-Training SLMs via Knowledge Distillation". NHR funding is provided by federal and Bavarian state authorities.

References

Nishant Balepur, Abhilasha Ravichander, and Rachel Rudinger. 2024. Artifacts or abduction: How do LLMs answer multiple-choice questions without the question? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Andrea Banino, Jan Balaguer, and Charles Blundell. 2021. Pondernet: Learning to ponder. *Preprint*, arXiv:2107.05407.

Samuel R. Bowman and George E. Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of NAACL-HLT*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every layer counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. Gpt or bert: why not both? *Preprint*, arXiv:2410.24159.

Cristiano Chesi, Veronica Bressan, Matilde Barbini, Achille Fusco, Maria Letizia Piccini Bianchessi, Sofia Neri, Sarah Rossi, and Tommaso Sgrizzi. 2024. Different ways to forget: Linguistic gates in recurrent neural networks. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 106–117, Miami, FL, USA. Association for Computational Linguistics.

Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of ICML*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. *Preprint*, arXiv:1807.03819.

Leo Gao. 2021. Multiple choice normalization in LM evaluation. `https://blog.eleuther.ai/multiple-choice-normalization/`. Accessed 2026.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. The language model evaluation harness.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *Preprint*, arXiv:2502.05171.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. 2023. Looped transformers as programmable computers. In *Proceedings of ICML*.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. *Preprint*, arXiv:1603.08983.

Riccardo Grazzi, Julien Siems, Arber Zela, Jörg K. H. Franke, Frank Hutter, and Massimiliano Pontil. 2025. Unlocking state-tracking in linear RNNs through negative eigenvalues. In *Proceedings of ICLR*.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of NAACL-HLT*.

Patrick Haller, Jonas Golde, and Alan Akbik. 2024. BabyHGRN: Exploring RNNs for sample-efficient language modeling. In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*.

Patrick Haller, Jonas Golde, and Alan Akbik. 2025. Sample-efficient language modeling with linear attention and lightweight enhancements. In *Proceedings of the First BabyLM Workshop*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. *Preprint*, arXiv:2305.02363.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of ICLR*.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2023. Transformers learn shortcuts to automata. In *Proceedings of ICLR*.

William Merrill, Jackson Petty, and Ashish Sabharwal. 2024. The illusion of state in state-space models. In *Proceedings of ICML*.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and inheritance in pre-trained language models. In *Proceedings of EACL*.

Giovanni Monea, Nathan Godey, Kianté Brantley, and Yoav Artzi. 2026. The state-prediction separation hypothesis. *arXiv preprint arXiv:2607.01218*.

Rareş Păpuşoi and Sergiu Nisioi. 2025. A comparison of elementary baselines for BabyLM. In *Proceedings of the First BabyLM Workshop*.

Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant M. Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Heess, and Raia Hadsell. 2020. Stabilizing transformers for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb datasets: Decanting the web for the finest

text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*.

PleIAs and AI Alliance. 2025. SYNTH. Hugging Face dataset, https://huggingface.co/datasets/PleIAs/SYNTH.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. 2021. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Richard S. Sutton. 2019. The bitter lesson. http://www.incompleteideas.net/IncIdeas/BitterLesson.html.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. 2025. Hierarchical reasoning model. *Preprint*, arXiv:2506.21734.

Guan Wang, Changling Liu, Chenyu Wang, Cai Zhou, Yuhao Sun, Yifei Wu, Shuai Zhen, Luca Scimeca, and Yasin Abbasi Yadkori. 2026. Hrm-text: Efficient pretraining beyond scaling. *Preprint*, arXiv:2605.20613.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at CoNLL*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. volume 8, pages 377–392.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025. Gated delta networks: Improving mamba2 with delta rule. In *Proceedings of ICLR*.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. 2024. Parallelizing linear transformers with the delta rule over sequence length. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Songlin Yang and Yu Zhang. 2024. FLA: A triton-based library for hardware-efficient implementations of linear attention. https://github.com/fla-org/flash-linear-attention.

A Reproducibility

Best HRM-rec configuration (10M): GatedDeltaNet mixer (head dim 128), slow→fast handoff gate, H_{cyc}=2, L_{cyc}=5, 6+6 layers, one-step gradient with warmup, `adam_atan2` with weight EMA 0.999, learning rate 3×10^{-4}, sequence length 256, effective batch 16, 10 epochs. The hand-off gate is the GTrXL GRU gate of Parisotto et al. (2020), whose six $d \times d$ projections account for its $\sim$3.5M parameters. Training uses a manual data-parallel loop in bf16, AdamW for GPT-2, and a cosine schedule with warmup for both models. Scaling runs use a 32k byte-level BPE tokenizer, sequence length 1024, a token budget (1B/3B), data streamed and packed into shards, and milestone checkpointing. The state-tracking probe trains on length-128 sequences and evaluates to length 512.

B Fine-tuning per-task scores

Table 5 gives the per-task (Super)GLUE scores behind the average quoted in §5.5, fine-tuned at the 100M-word scale.

task	GPT-2	HRM-rec
BoolQ	69.9	70.0
MNLI	61.0	61.9
QQP	72.8	72.2
MRPC	83.8	85.3
MultiRC	65.4	67.0
RTE	56.8	**62.6**
WSC	67.3	67.3
avg	68.1	**69.5**

Table 5: Fine-tuning on the BabyLM (Super)GLUE subset at the 100M-word scale (accuracy, F1 for MRPC/QQP).

C Developmental-scale results

Table 6 gives the full zero-shot comparison at 10M and 100M words, underlying the phase diagram of Table 3. EWoK is near the 50% chance floor for both models at both scales, consistent with reports that it is barely testable at this data scale (Hu et al., 2024), and shows no architecture effect.

scale	model	BLiMP	Suppl	COMPS	Ent.	EWoK
10M	GPT-2	67.0	60.9	51.5	27.9	50.8
10M	HRM-rec	68.6	56.1	51.4	29.1	51.4
100M	GPT-2	75.3	62.2	57.9	32.7	55.7
100M	HRM-rec	75.8	60.5	57.6	28.6	55.6

Table 6: Zero-shot at the developmental scales. BLiMP/Suppl/COMPS/Entity are 3-seed means, and EWoK is a single run on the fast subset (50% chance). The HRM-rec 10M BLiMP seed std is $\pm$0.7 (Table 1).

D Synthetic-data results

Table 7 gives the per-mixture zero-shot scores behind §5.6: BLiMP falls monotonically with the SYNTH fraction at both scales, while `entity_tracking` is the one score it raises.

scale / model	SYNTH	BLiMP	Suppl	COMPS	Ent.
10M HRM-rec	0%	**68.6**	57.8	51.4	29.1
	25%	67.7	58.0	51.1	32.2
	50%	65.3	56.8	50.8	38.2
	75%	64.2	56.4	49.9	33.6
3B GPT-2	0%	76.0	57.4	57.8	30.9
	50%	73.8	55.0	56.7	35.6
3B HRM-rec	0%	79.7	61.8	59.4	30.9
	50%	78.8	58.9	58.2	32.9

Table 7: Zero-shot results for mixing SYNTH reasoning traces into training.

E Per-task reasoning scores

Table 8 gives the normalized-accuracy breakdown behind the reasoning-suite numbers quoted in §5.4. Several tasks (HellaSwag, CommonsenseQA, WinoGrande) sit at chance for all $\sim$167M models, so the suite average is carried by ARC-easy. GPT-2 leads it at both scales.

scale	**model**	ARC-e	ARC-c	OBQA	HellaSwag	PIQA	CSQA	WinoG.	**avg**
1B	GPT-2 nat	33.5	21.3	26.8	25.1	54.6	19.6	50.8	33.1
1B	GPT-2 mix	31.1	21.1	28.0	25.3	54.1	19.6	49.7	32.7
1B	HRM-rec nat	29.2	22.4	26.8	25.1	53.6	19.6	51.3	32.6
1B	HRM-rec mix	29.4	22.6	26.8	25.0	53.6	19.6	51.2	32.6
3B	GPT-2 nat	37.3	20.6	26.0	26.1	56.5	19.7	50.3	33.8
3B	GPT-2 mix	36.2	21.1	24.6	26.5	54.1	19.6	50.6	33.2
3B	HRM-rec nat	32.0	21.3	27.0	25.2	54.0	19.6	50.0	32.7
3B	HRM-rec mix	31.3	21.5	26.2	25.2	54.1	19.6	49.8	32.5

Table 8: Reasoning suite, per task (% normalized accuracy). Chance is 25% (ARC, OpenBookQA, HellaSwag), $\sim$20% (CommonsenseQA), 50% (WinoGrande).

Language Allocation and Targeted Data Design for a Multilingual BabyLM

Naoki Mori and Seiji Kyo and Kohki Goda
the Graduate School of Informatics
Osaka Metropolitan University, Osaka, Japan
mnao@omu.ac.jp

Abstract

We study language allocation and data design for English–Dutch–Chinese pretraining in the multilingual BabyLM 2026 setting. Across three matched seeds, three fixed mixtures showed no significant difference in the seven-benchmark selection average (TASK-7) after Holm correction. Two within-language contrasts were significant after correction within each language: English favoured 50/25/25 by 2.487 points and Chinese favoured 27.5/27.5/45 by 2.151 points. The mixtures were the three highest-scoring conditions of a single-seed pilot, so that comparison is exploratory. An inverse-score controller changed sampling shares without a detected aggregate improvement in a 25M-word proxy. Across four matched seeds, adding synthetic and parallel-corpus text raised the zero-shot task average (ZS16) from 56.57 to 57.37, the largest language-specific difference being Chinese (+1.47); both remained significant after Holm correction. Four of five Chinese tasks improved in the three seeds with task-level records. That condition also consumed 14.9% more tokens and used its own tokenizer, so it does not isolate content from exposure. Multi-token prediction showed no detected improvement across three seeds. Parameter-count comparisons were single runs; the 213M model had the highest fine-tuning average (FT22, 41.25) and a leaderboard-reported Multilingual Average of 50.68.

1 Introduction

The multilingual track of BabyLM 2026 fixes an English-equivalent training budget across English, Dutch, and Chinese (Choshen et al., 2026). The constraint turns corpus construction into an allocation problem: additional exposure for one language reduces exposure elsewhere. This is a small-model setting in which the transfer–interference trade-off reported for much larger multilingual systems can be examined directly (Conneau et al., 2020; Chang et al., 2024).

We report two sets of experiments under this budget. The first concerns **language allocation** (§4–6): a tokenizer pilot, three fixed language mixtures with matched seeds, and a dynamic reallocation controller. The second concerns **data design and model configuration** (§7): corpus augmentation with synthetic and parallel text, multi-token prediction, and comparisons across parameter counts.

Replicated comparisons use matched seeds and paired tests; single-run results are descriptive. We document data construction, budget accounting, and overlap screening in §7.1.

2 Related work

The budgeted setting. The multilingual track draws on BabyBabelLM, a 45-language collection (Jumelet et al., 2026a), and uses Byte Premium to account for differences in content-matched UTF-8 corpus sizes across languages (Arnett et al., 2024; Choshen et al., 2026). We study how to divide this fixed budget.

What the division is expected to do. Two lines of work predict opposing effects of language allocation. Exposure to one language can benefit others through cross-lingual transfer (Conneau et al., 2020). Conversely, languages compete for fixed model capacity, so shifting exposure toward one can degrade the others (Chang et al., 2024). Those studies use larger models and corpora and vary the language set. We instead fix the language set, architecture, and optimizer and vary the proportions (§5). Our observed differences alone do not distinguish transfer from capacity competition.

Tokenization and objectives as design levers. Unigram segmentation (Kudo, 2018) is an alternative to byte-pair encoding (BPE); controlled comparisons have favored unigram (Bostrom and Dur-

Proceedings of the Second BabyLM Workshop, pages 735–749
October 29, 2026 ©2026 Association for Computational Linguistics

rett, 2020). Fusco et al. (2025) studied linguistically motivated segmentation in BabyLM. Multi-token prediction (MTP) adds auxiliary future-token heads (Gloeckle et al., 2024). Our tokenizer pilot varies both segmentation and vocabulary size, hence parameter count, and is exploratory (§4); our objective comparison detects no improvement (§7.3).

Adaptive mixtures. DoReMi learns fixed domain weights with a separate proxy model and a minimax objective (Xie et al., 2023). Our heuristic updates language probabilities within one run using benchmark accuracy (§6); we claim neither novelty nor a controlled comparison with DoReMi.

3 Experimental setup

3.1 Data and models

Ratios are written as English/Dutch/Chinese. The allocation study uses the official BabyBabelLM data without augmentation (Choshen et al., 2026; Jumelet et al., 2026a). The data-design study adds parallel text (Tiedemann, 2020; Zhang et al., 2020) and offline task-family-targeted synthetic text (§7.1).

All models are GPT-2-style causal decoders (Radford et al., 2019) trained from scratch with AdamW (Loshchilov and Hutter, 2019) (Table 1). Learning rates decay linearly except in the dynamic experiment, which uses one full-run cosine schedule (Appendix C). Except in the tokenizer pilot, byte-level BPE is fitted to each model's corpus; the baseline and augmented conditions therefore use different tokenizer files. Text is concatenated into fixed-length blocks. At a fixed epoch count, larger corpora require more optimizer steps. Context length is the maximum position length; block length is the packed training sequence length.

Setting	Alloc.	Data-des.	Final
Layers	12	12	14
Attention heads	12	12	16
Hidden size	768	768	1024
Context length	512	512	512
Block length	512	128	128
Vocabulary	16,000	32,000	32,000
Parameters	97.737M	110.025M	212.790M
Peak LR	6e−4	4.5e−4	4.5e−4
LR schedule	linear[§]	linear	linear
Warmup	500 steps	2%	2%
Weight decay	0.01	0.01	0.01
Precision	bf16	bf16	bf16
Epochs	10	10	10
Batch (tok)	131,072	16,384	16,384
MTP heads k	—	4[†]	4
MTP aux. weight	—	0.3[†]	0.3

Table 1: Model settings. [†]Multi-token prediction (MTP) is used only in the +MTP comparison, not the augmentation comparison. The MorPiece pilot uses 39,697 vocabulary items and 115.937M parameters. [§]The dynamic experiment uses cosine decay.

3.2 Evaluation metrics

The allocation study uses Task-7, the unweighted mean of seven zero-shot minimal-pair scores (English BLiMP and MultiBLiMP; Dutch BLiMP-NL, MultiBLiMP, and XCOMPS; Chinese ZhoBLiMP and XCOMPS) (Warstadt et al., 2020; Suijkerbuijk et al., 2025; Jumelet et al., 2026b; Liu et al., 2026; He et al., 2025). Lang-3 averages within language, then across the three languages. Table 19 additionally reports All-16, the unweighted mean of 16 task scores: the seven of Task-7 together with HellaSwag, Winogrande, and XStoryCloze in each language. It covers the same task set as ZS16 below and averages it the same way. The English grammatical component of both Task-7 and All-16 is the standard BLiMP release, whereas ZS16 uses the BabyLM-filtered subset, so values are not comparable across the two studies; the allocation study selected on Task-7, not on All-16. All scores are produced with the official BabyLM multilingual evaluation pipeline. The seven benchmarks are equally weighted within a language, while sub-tasks inside a BLiMP-style benchmark are weighted by their number of evaluation items.

The data-design study uses ZS16 and FT22. ZS16 is the unweighted mean of the 16 public zero-shot task scores available for every ablation (English 5, Dutch 6, Chinese 5), and FT22 is the unweighted mean of the 22 non-POS fine-tuning task–language scores available for every ablation. These are descriptive comparison metrics, not the official Multilingual Average. The official score additionally includes GlobalPIQA and POS and averages the complete task set within each language before macro-averaging across English, Dutch, and Chinese; ZS16 is therefore not a three-language equal-weight macro. All scores are reported as percentages.

3.3 Statistical analysis

We used two-sided paired t-tests over matched seeds. For each intervention in the data-design study, the ZS16 task average and three language-specific tests formed one family and were adjusted

with Holm's procedure (Holm, 1979). In the fixed-mixture study, the three pairwise contrasts within each language formed a separate family per language. Confidence intervals are unadjusted 95% t intervals for the paired mean difference. With $n = 4$, normality cannot be checked and the smallest attainable two-sided sign-flip p-value is .125; we therefore report paired-t evidence together with the seed-level differences in Appendix B rather than describing any effect as robust. Conditions differing in more than one factor are not used to attribute causes.

4 Tokenizer pilot

Table 19 compares equal-mixture runs with one training seed each at the full 100M-word budget over ten epochs, using byte-level BPE, a byte-level unigram model, and MorPiece (Fusco et al., 2025). MorPiece is a split-based tokenizer that posits a morpheme boundary wherever the Tolerance Principle applies as a word traverses the lexicon. The mixture pilot of Table 20 instead used a 25M-word budget, so the two tables differ in training data as well as in what they vary: cumulative raw-token exposure over the ten epochs is 9.94×10^8 here against 2.48×10^8 there. We do not attribute the difference in score level to the budget alone. MorPiece has the highest observed TASK-7 and LANG-3 scores and leads on four of seven component tasks. Because there is one seed per condition, we treat the seven tasks as an across-task descriptive comparison rather than as replicates: the mean task difference relative to BPE is 1.337 points and relative to Unigram 1.585 points, but these summaries do not express run-to-run uncertainty. MorPiece also adds 18.199M parameters (18.6%) through the embedding and output layers and changes vocabulary and segmentation together, so this pilot is exploratory.

5 Fixed language mixtures

We train 50/25/25, 40/40/20, and 27.5/27.5/45 with three matched seeds each (Table 2). Each language scores highest at its largest allocation; 40/40/20 has the highest observed TASK-7 mean.

5.1 Choice of mixtures

The three mixtures ranked highest on TASK-7 in a seven-mixture pilot with one seed per mixture, MorPiece, and a 25M-word proxy (Table 20). The symmetric conditions 25/50/25 and 25/25/50 ranked fourth and seventh and were not retained. Selecting on the reported metric makes the aggregate comparison exploratory.

Allocation spans differ: 22.5 points for English, 25 for Chinese, and 15 for Dutch, whose largest share (40) ties English. Thus the contrasts cannot compare allocation sensitivity across languages.

5.2 Aggregate comparisons

All three pairwise TASK-7 intervals include zero (Table 3), and the ordering varies by seed: 50/25/25 is highest for seed 1, 40/40/20 for seeds 2 and 3. At this sample size a difference was not detected; this does not establish equivalence.

5.3 Language-level comparisons

Within each language the three pairwise contrasts form one family; Table 10 gives the seed-level scores and Table 17 all nine contrasts. Two contrasts remain significant after correction. English is 2.487 points higher under 50/25/25 than under 27.5/27.5/45 (95% CI $[2.078, 2.895]$; Holm-adjusted p=.0044). Chinese is 2.151 points higher under 27.5/27.5/45 than under 40/40/20 (95% CI $[1.047, 3.255]$; Holm-adjusted p=.0418). Dutch has its highest observed mean under 40/40/20, but its corrected contrasts are not significant. Relative to 50/25/25, 40/40/20 is 1.642 higher in Dutch, .863 lower in English, and .727 lower in Chinese. As noted in §5.1, the Dutch contrasts cover a 15-point allocation range against 22.5 for English and 25 for Chinese, so these contrasts do not support a comparison of allocation sensitivity across the three languages.

6 Dynamic reallocation

The dynamic experiment uses a 25M-word proxy, BPE 16k, and a 97.737M-parameter decoder. The inverse-score heuristic raises the share of low-scoring languages. Starting from $c_{0,\ell} = 1/3$, it computes every two epochs

$$q_{t,\ell} = \frac{(s_{t,\ell} + \epsilon)^{-1}}{\sum_{j \in \{\text{en,nl,zh}\}} (s_{t,j} + \epsilon)^{-1}}, \qquad \epsilon = 10^{-8}, \tag{1}$$

where $s_{t,\ell}$ is the unweighted mean of that language's benchmark accuracies on the same seven-benchmark suite used to report TASK-7, on the 0–1 scale and without clipping, evaluated zero-shot on the checkpoint at the end of the preceding two epochs (steps 384, 768, 1152, and 1536

Mixture (en/nl/zh)	English	Dutch	Chinese	Task-7	Lang-3
50/25/25	78.884 ± .597	73.054 ± .331	65.352 ± .657	72.519 ± .167	72.430 ± .142
40/40/20	78.021 ± .443	74.696 ± .460	64.626 ± .082	72.769 ± .322	72.448 ± .300
27.5/27.5/45	76.397 ± .668	73.145 ± .285	66.776 ± .513	72.255 ± .432	72.106 ± .457

Table 2: Fixed mixtures: three-seed mean ± sample SD.

Task-7 contrast	Δ	95% CI	p_{Holm}
40/40/20 − 50/25/25	.250	[−.932, 1.431]	.9191
40/40/20 − 27.5/27.5/45	.514	[−.495, 1.523]	.4792
50/25/25 − 27.5/27.5/45	.265	[−1.169, 1.698]	.9191

Table 3: Paired tests on TASK-7 ($n = 3$), one family of three contrasts.

of 1920). The direct condition sets $c_{t+1} = q_t$; the smoothed condition averages the new target with the current shares, $c_{t+1} = .5c_t + .5q_t$, which halves the step size. In both, the result is then projected onto the simplex subject to per-language bounds of $[0.10, 0.60]$. Every one of the 24 updates landed strictly inside those bounds—all post-update shares lie between .311 and .369—so the bounds never became active and the trajectories are those of the unconstrained rule. Language is drawn per 512-token block, and blocks are then sampled uniformly with replacement from that language's pool, so the realized counts vary around the target shares rather than partitioning the data exactly. Each method is run with seeds 1–3. Appendix C gives the per-benchmark scores and the seed-level final shares.

Mean Chinese exposure rose from 83.67M to 89.00M raw tokens under direct updates (87.89M when smoothed). Neither method improved mean Chinese, TASK-7, or LANG-3 scores or reduced the language gap (Table 4). All paired aggregate tests have p_{Holm}=1.0 (fixed minus inverse Δ=.181, 95% CI $[-.883, 1.245]$). Because the same public benchmarks supply feedback and outcomes, this is a diagnostic of adaptation, not held-out generalization.

7 Data design and model configuration

We evaluate corpus augmentation and multi-token prediction separately before reporting the final models.

7.1 Augmentation data construction

What is added. The augmented corpus is the baseline corpus *plus* two sources; no real text is removed or replaced. Table 21 gives the composition. The baseline is sentence-segmented BabyBabelLM text built to per-language targets of 34M/ 33M/ 33M English-equivalent words. Chinese word counts are CJK (Chinese, Japanese, Korean) character counts, used as a segmenter-free proxy; the Budget paragraph below relates this convention to the byte measure the track actually caps.

Synthetic text. Synthetic text was generated offline by **Qwen3.5-4B** (HF model `Qwen/Qwen3.5-4B`, immutable revision `851bf6e806efd8d0a36b00ddf55e13ccb7b8cd0a`; approximately 4B parameters, within the challenge's teacher-size limit). Sampling used temperature 0.8, top-p 0.9, and at most 160 new tokens per sample. Prompts were rendered with the model's chat template with reasoning traces disabled, and any residual `<think>` span was stripped from the output. We kept all 25,000 generated samples per language without length, language, or quality filtering. Prompts were split evenly over four templates that target the task *families* in the evaluation suite, not its items: STORY (narrative continuation), COMPARE (comparatives), COREF (pronoun reference), and FACT (simple declarative statements). Each prompt instantiates one template with one topic drawn from a fixed list of 143 everyday child-directed topics (kitchen, weather, school, the playground, and so on), with a per-language sampling seed (11/22/33 for English/Dutch/Chinese). The templates are given verbatim in Appendix E. Prompts contain a task-family description, a topic, a length instruction, and the target language name; they contain no evaluation instance, label, answer key, minimal pair, or output of the evaluation pipeline. The teacher was used only to generate text; no teacher logits, scores, or rankings were used for training.

Parallel text. Parallel text is drawn from community-produced translation corpora rather than teacher output, and contains some noisy or fragmentary sentences. We use Tatoeba (Tiedemann, 2020) for en–nl and en–zh, taken in full after filtering, and OPUS-100 (Zhang et al.,

Method	Eng	Dut	Chi	Task-7	Lang-3	Gap	Final zh
Fixed	67.564	68.671	58.933	65.573	65.056	9.738	.333
Inverse	67.066	68.686	58.775	65.391	64.842	9.910	.366
Smoothed	67.849	68.440	58.392	65.400	64.894	10.048	.363

Table 4: Dynamic experiment: three-seed means for fixed equal sampling, direct inverse-score updates, and smoothed inverse-score updates. Gap is the maximum minus minimum language mean; Dutch is the maximum and Chinese the minimum in every condition. Final zh is the Chinese sampling share applied over the last two epochs.

2020) to fill the remaining allowance, split equally between the two pairs. Filters keep pairs of 2–60 words per side with a length ratio of at most 3.0 for Tatoeba and 3–40 words with a ratio of at most 2.0 for OPUS-100, and drop pairs whose first 60 characters on both sides duplicate an already-accepted pair. Each retained pair yields two training sequences, one per direction, using the template `L1: X\nL2: Y` with the language names spelled out (`English: …\nDutch: …`) and its reverse. This gives 269,818 pairs and 539,636 sequences (240,952 from Tatoeba, 298,684 from OPUS-100; 319,296 en–nl and 220,340 en–zh). Tatoeba's Chinese side mixes scripts: 13.3% of the en–zh Chinese bytes are on lines containing traditional characters. We charge all Chinese at the simplified premium; charging those lines at the traditional premium instead would change the total by 0.07 MB, or 0.01% of the cap. Examples of both sources are given in Appendix E.

Budget. The track measures the budget in bytes, not in words. In the organizers' data-preparation code the 100M-word tier is the constant 543.00 MB. A language qualifies while its UTF-8 size stays below 543.00 MB × premium. Equivalently, size divided by the premium must stay at or below 543.00 MB.[1] We apply that rule with the published coefficients: 1.0 for English, 1.0516 for Dutch, and 0.93597 for simplified Chinese, the script BabyBabelLM uses. The call for papers lists 0.9894, which the same table gives for traditional Chinese.

Under this rule the baseline corpus is 457.5 MB and the augmented corpus 527.7 MB of English-equivalent bytes, that is 84.3% and 97.2% of the cap (Table 5). Both are within budget, and the augmented corpus leaves little headroom. The parallel file mixes languages within a line; charging it entirely at the English premium and splitting it by language give 527.74 and 527.66 MB, so that choice does not affect the conclusion.

Our construction code additionally tracked a word count, dividing whitespace tokens (English, Dutch) or CJK characters (Chinese) by the same premiums. That quantity is a proxy in a different unit and is not the one the track caps; we report it in Table 5 only because the corpus was assembled against it. It does not track the byte measure across languages, because a premium-adjusted byte count converts to a word count only at that language's bytes per word. Relative to English, the proxy is inflated by a factor of 1.08 for Dutch and 1.62 for Chinese, where a character stands in for a word. Those factors, not the budget, are what the word column reflects.

Because the augmentation is additive and the epoch count is fixed at 10, the augmented condition also receives more optimizer steps: 87,820 against 76,440 at 16,384 tokens per step, that is 1.439B against 1.252B model tokens read, a 14.9% increase. These are BPE tokens including repetition across epochs, a different quantity from the English-equivalent word budget above, and we report them separately for that reason. The two conditions therefore differ in training exposure as well as in content, and each was trained with a tokenizer fitted to its own corpus, so §7.2 compares two data conditions rather than isolating content at matched exposure.

Overlap audit. We screened the distributed English BLiMP, Dutch BLiMP-NL, and Chinese ZhoBLiMP minimal-pair files using normalized exact matching followed by 8-gram candidate retrieval and character 5-gram Jaccard similarity (Appendix I). Across 9,853,776 line–language checks, three training lines matched two of 143,150 evaluation sentences; none was synthetic. The near-duplicate pass found no additional matches, but its recall was not measured. This au-

[1] `src/pad_utils.py` and `resources/byte_coefs_20240233.tsv` in the organizers' `multilingual-babylm` repository.

	Baseline		Augmented	
	MB	eq. MB	MB	eq. MB
English	183.4	183.4	188.1	188.1
Dutch	173.1	164.6	177.3	168.6
Chinese	102.5	109.5	106.9	114.2
Parallel	—	—	56.9	56.9
Total	459.0	**457.5**	529.1	**527.7**
Cap: 543.00 MB (84.3% and 97.2% used)				
Construction-time word proxy, millions:				
English		34.000		37.185
Dutch		33.000		34.929
Chinese		32.792		35.578
Optimizer steps		76,440		87,820
Model tokens read		1.252B		1.439B

Table 5: Budget and exposure. The upper block applies the track's rule: UTF-8 megabytes divided by the language premium, against the 543.00 MB constant the organizers use for the 100M-word tier. The middle block is the construction-time word proxy (whitespace tokens, or CJK characters for Chinese, divided by the same premiums); it is not the capped quantity and is listed for provenance. Model tokens read are BPE tokens including repetition across epochs.

dit covers neither the remaining zero-shot tasks nor fine-tuning or GlobalPIQA, and does not establish absence of overlap.

7.2 Targeted augmentation

We compare the real-text baseline with the augmented condition across four matched seeds at 110.025M parameters (Table 6). The augmented condition scored higher on ZS16 ($56.57 \rightarrow 57.37$). The largest language-specific mean difference is observed for Chinese, the lowest-scoring baseline language. The ZS16 and Chinese differences remain significant after Holm correction across the four reported metrics; the paired tests did not detect differences for English or Dutch.

Task-level records were uniquely recoverable for three seeds (7, 99, 123). Four of five Chinese tasks improved; leave-one-task-out means remained positive ($+1.10$ to $+1.83$; Appendix D). Across languages, 13 of 16 ZS16 tasks improved (Table 16). These are descriptive decompositions, not independent replicates or evidence against selection bias.

The comparison confounds content with 14.9% more training tokens and a refit tokenizer. It also combines synthetic and parallel text. We did not run exposure-matched real-text controls or the 2×2 design needed to separate the added sources.

7.3 Multi-token prediction

The +MTP condition adds $k{=}4$ prediction heads on the shared trunk. Head $j{=}1$ is the ordinary next-token head; heads $j \geq 2$ apply a per-head linear projection and GELU before reusing the output embedding, and each of their losses enters the sum with weight 0.3. The three projections add 1.772M parameters, so the condition trains 111.797M parameters against the 110.025M of Table 1; at the final model's hidden size they add 3.146M, for 215.936M. The projections are unused at evaluation, where the parameter counts are those listed. The number of active heads is ramped from 1 to 4 over the first four epochs, after which all four are active (Gloeckle et al., 2024). Only the $j{=}1$ head is used at evaluation, so the model is evaluated as an ordinary causal decoder. Adding this objective to the augmented corpus changes ZS16 by $+0.097$ points across three matched seeds (95% CI $[-0.617, 0.810]$; $t(2){=}0.583$; Holm-adjusted $p{=}1.0$). No language-specific contrast is significant after correction. This comparison does not detect an improvement; it does not show that multi-token prediction is ineffective in general.

7.4 Single-run comparisons

With one run per parameter count, the 213M model had the highest observed FT22 score (41.25) against 39.81 at 110M, 39.90 at 263M and 40.15 at 338M, with ZS16 between 57.82 and 57.99. Among single-run ablations at 213M, three variants had higher ZS16 and lower FT22 than the reference run and the rest were lower on both. Both sets are in Appendix A. These are descriptive observations: they establish neither a scaling effect nor a relationship between the two summaries that

Metric	Δ	95% CI	$t(3)$	raw p	Holm p
ZS16 (task avg.)	+0.800	[0.321, 1.279]	5.320	.0130	.0389
English	+0.808	[−0.173, 1.788]	2.622	.0789	.1578
Dutch	+0.235	[−0.287, 0.757]	1.433	.2474	.2474
Chinese	+1.473	[1.238, 1.707]	19.996	.00027	.0011

Table 6: Augmented condition minus real-text baseline, four matched seeds. The four metrics form one Holm family; intervals are unadjusted. The two conditions differ in corpus content and in training exposure (Table 5).

holds across interventions. We do not test explanations for the co-occurrence of higher ZS16 with lower FT22.

7.5 Evaluated models

Two models were evaluated at 213M. Our submitted model used the augmented corpus of §7.1 and is listed on the official leaderboard as `babylm-2026-mtl-bigckpt` (ZS16 57.82, FT22 41.25); it obtained a leaderboard-reported Multilingual Average of 50.68. The second model, listed as `babylm-2026-mtl-stage2`, used a modified corpus in which part of the real text was replaced by additional synthetic text at matched language proportions, with the parallel text left intact. It has higher ZS16 (58.40) but lower FT22 (39.52) and a Multilingual Average of 50.14, consistent with the higher ZS16 and lower FT22 observed in three single-run ablations (§7.4). Appendix G lists the training runs behind the seed comparisons and the two evaluated models.

8 Conclusion

Across three fixed mixtures with matched seeds, we detected within-language differences of 2.487 points in English and 2.151 points in Chinese, but no aggregate difference after correction. The mixtures came from a single-seed pilot and cover different allocation ranges, which limits comparisons of sensitivity across languages. The inverse-score controller changed sampling shares without a detected aggregate improvement. Across four matched seeds the augmented condition scored higher on ZS16, most for Chinese, with positive differences on four of five Chinese tasks in the seeds whose task-level records survive. That condition also consumed 14.9% more tokens and used a tokenizer refit on its own corpus, so it does not separate content from exposure. Multi-token prediction showed no detected improvement across three seeds, and the remaining single-run comparisons are descriptive. The 213M model obtained a leaderboard-reported Multilingual Average of 50.68. These results support reporting both aggregate and language-level scores when comparing multilingual training conditions.

Limitations

Seed counts are three or four for the replicated comparisons and one elsewhere; with $n \leq 4$ the intervals are wide and normality cannot be checked. The public evaluation suite was reused for tokenizer, proxy, mixture, and augmentation selection, so the reported values are subject to selection bias. Proxy and full runs share a fixed 500-step warmup whose relative share differs between them.

The three fixed mixtures were selected as the three highest-scoring conditions of a single-seed pilot ranked on Task-7, the same aggregate on which §5.2 then compares them, so that comparison is subject to selection bias. The pilot itself used one seed per mixture at a smaller scale and with a different tokenizer, so its ranking is not a reliable ordering. The retained mixtures vary each language over a different range (§5.1), and the symmetric conditions 25/50/25 and 25/25/50 exist only as single pilot runs.

The augmented corpus is additive, so the augmentation comparison confounds content with a 14.9% increase in training exposure and with a tokenizer refit on each corpus, and it combines synthetic and parallel text without separating them (§7.1). Both corpora are inside the track's byte budget, the augmented one at 97.2% of the cap, so the design leaves little room to add data without displacing some.

The overlap audit covers the distributed English BLiMP, Dutch BLiMP-NL, and Chinese ZhoBLiMP files described in Appendix I, but not the other zero-shot or fine-tuning tasks, or GlobalPIQA. For one of the four augmentation seeds the per-task record could not be uniquely recovered, because its results directory accumulated several evaluation passes. The seed-level language means of Table 11 are unaffected, but the task-level decomposition in Appendix D uses the other three

seeds. Leaderboard values are those the leaderboard reported for these entries; the leaderboard remains open.

Ethical Considerations

The synthetic portion of our corpus was accepted without filtering (§7.1): we did not verify that every sample is in the intended language, factually correct, or free of social bias, and a model trained on it inherits whatever the teacher produced. The teacher is used only to generate text, under the terms of its published model card, and we record its immutable revision so that the generation can be traced. The parallel text is community-contributed and contains noisy sentences. Training and evaluation data are used under the licences of their sources, and the evaluation suites are the ones distributed by the track.

References

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K. Bergen. 2024. When is multilinguality a curse? language modeling for 250 high- and low-resource languages. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4074–4096, Miami, Florida, USA. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Achille Fusco, Maria Letizia Piccini Bianchessi, Tommaso Sgrizzi, Asya Zanollo, and Cristiano Chesi. 2025. Linguistic units as tokens: Intrinsic and extrinsic evaluation with BabyLM. In *Proceedings of the First BabyLM Workshop*, pages 496–507, Suzhou, China. Association for Computational Linguistics.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15706–15734. PMLR.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schütze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81, Vienna, Austria. Association for Computational Linguistics.

Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prévot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in Chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Technical Report*.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Jörg Tiedemann. 2020. The Tatoeba translation challenge – realistic data sets for low resource and multilingual MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1174–1182, Online. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems*, volume 36.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

A Single-run comparisons

Parameters	ZS16	FT22
110M	57.86	39.81
213M	57.82	41.25
263M	57.99	39.90
338M	57.86	40.15

Table 7: One run per parameter count. The 110M row is a single run of the augmented-plus-MTP condition and is not the three-seed mean of §7.

Variant (vs 213M reference run)	ΔZS16	ΔFT22
Hybrid morpheme tokenizer	+0.50	−1.47
Substituted synthetic text	+0.58	−1.73
Hybrid tokenizer + substituted synthetic	+0.42	−2.38
Block length 256	−0.32	−0.74
Block length 512	−0.40	−1.54
Character-level vocabulary	−0.78	−0.92
Additional translation text	−0.22	−2.32

Table 8: Single-run ablations at 213M, reported as differences from the reference run.

B Seed-level results

Mixture	Seed	Task-7	Lang-3
50/25/25	1	72.705	72.591
	2	72.380	72.322
	3	72.473	72.378
40/40/20	1	72.405	72.112
	2	72.883	72.539
	3	73.018	72.692
27.5/27.5/45	1	71.942	71.768
	2	72.748	72.626
	3	72.075	71.924

Table 9: Seed-level aggregate scores, fixed-mixture experiment.

Method	Seed Task-7 scores	Mean	SD
Fixed	65.073 / 65.966 / 65.679	65.573	.456
Inverse	65.361 / 65.416 / 65.397	65.391	.028
Smoothed	65.729 / 65.586 / 64.885	65.400	.452

Table 12: Seed-level Task-7 scores, dynamic experiment.

C Dynamic experiment details

Lang	Benchmark	Fixed	Inverse	Smoothed
en	BLiMP	59.153	59.198	59.204
en	MultiBLiMP	75.974	74.935	76.494
nl	BLiMP-NL	70.702	70.691	70.294
nl	MultiBLiMP	84.299	84.842	84.456
nl	XCOMPS	51.014	50.524	50.571
zh	ZhoBLiMP	66.005	66.197	65.351
zh	XCOMPS	51.862	51.354	51.433

Table 13: Per-benchmark three-seed means for the dynamic experiment. These seven scores are both the control signal and the reported outcome; the language means of Table 4 are their unweighted within-language averages.

Mixture	Seed	English	Dutch	Chinese
50/25/25	1	78.368	73.388	66.017
	2	79.538	72.726	64.703
	3	78.746	73.049	65.338
40/40/20	1	77.616	74.165	64.555
	2	77.952	74.949	64.716
	3	78.494	74.976	64.606
27.5/27.5/45	1	75.716	72.983	66.606
	2	77.051	73.475	67.353
	3	76.423	72.978	66.371

Table 10: Seed-level language means for the fixed-mixture experiment. These are the scores behind Table 2 and the within-language contrasts of §5.3, and the rows of Table 17 that report them.

Condition	Seed	ZS16	English	Dutch	Chinese
Baseline	42	56.81	59.01	59.18	51.77
	7	56.61	58.27	59.12	51.94
	99	56.55	58.61	58.87	51.70
	123	56.32	58.08	58.61	51.81
Augmented	42	57.37	59.12	59.32	53.29
	7	57.66	59.76	59.32	53.56
	99	57.07	59.12	58.78	52.97
	123	57.39	59.20	59.30	53.29
Augmented + MTP	7	57.44	58.98	59.55	53.36
	99	57.41	59.53	59.68	52.56
	123	57.56	59.30	59.80	53.14

Table 11: Seed-level scores for the data-design comparisons. Augmentation compares Augmented with Baseline on four matched seeds (42, 7, 99, 123). The MTP comparison uses Augmented + MTP versus Augmented on three matched seeds (7, 99, 123).

Method	Seed	en	nl	zh
Inverse	1	.3212	.3137	.3651
	2	.3217	.3156	.3627
	3	.3197	.3113	.3690
Smoothed	1	.3242	.3165	.3593
	2	.3209	.3171	.3620
	3	.3182	.3136	.3682

Table 14: Sampling shares applied over the final two epochs. The fixed condition holds $1/3$ throughout. All values lie inside the $[0.10, 0.60]$ bounds, so the projection step was never active.

Each run takes 192 optimizer steps per epoch at an effective batch of 256 sequences, so 10 epochs are 1,920 steps. The block pool holds 19,531 English, 20,539 Dutch, and 9,082 Chinese blocks of 512 tokens, and each two-epoch segment draws 98,304 blocks, for 2.52×10^8 tokens over the run. The pool itself was built at 40/40/20 with the Byte Premium applied to its per-language token caps. The equal start of §6 is therefore a statement about sampling probabilities, not about the pool being balanced, and the controller reweights sampling without reapplying the premium. The trainer is rebuilt at each segment boundary but resumes from the preceding checkpoint, and the optimizer moments, the scheduler position, the global step, and the training RNG state are restored alongside the weights. We checked this against the five stored checkpoints of one run: the optimizer step, the scheduler's `last_epoch`, and `global_step` agree at 384, 768, 1152, 1536, and 1920, with `max_steps` equal to 1920 throughout. Thus one cosine schedule spans the whole run and the run is not five independent two-epoch runs. Restoring the RNG state does not make the data order identical to an uninterrupted run, because each segment draws a fresh sampling plan from a separate generator seeded by the run seed and the segment index.

D Chinese task-level decomposition

Chinese task	Baseline	Augmented	Δ
ZhoBLiMP	78.13	79.43	+1.29
Winogrande-zh	49.77	51.11	+1.35
XStoryCloze-zh	47.29	50.18	+2.89
HellaSwag-zh	27.36	27.34	−0.02
XCOMPS-zh	56.53	58.32	+1.79
Chinese mean	51.82	53.28	+1.46

Table 15: Per-task means over the three matched seeds (7, 99, 123) whose per-task records are uniquely recoverable. Leave-one-task-out Chinese means range from +1.10 to +1.83. These five scores come from the same runs and are not independent replicates.

Lang	Task	Δ
en	BLiMP (BabyLM-filtered)	+3.107
en	MultiBLiMP	+0.867
en	Winogrande	+0.875
en	XStoryCloze	+0.591
en	HellaSwag	−0.256
nl	BLiMP-NL	−0.074
nl	MultiBLiMP	+0.055
nl	Winogrande	+0.550
nl	HellaSwag	+0.239
nl	XCOMPS	+0.566
nl	XStoryCloze	+0.233
zh	ZhoBLiMP	+1.292
zh	Winogrande	+1.347
zh	XStoryCloze	+2.889
zh	HellaSwag	−0.024
zh	XCOMPS	+1.793

Table 16: Augmented minus baseline mean difference on each of the 16 ZS16 tasks, over the three seeds (7, 99, 123) with uniquely recoverable task-level records. Thirteen of the sixteen differences are positive. These are the same runs, not independent replicates.

E Generation prompts and data examples

The four synthetic-text templates, with `{t}` the sampled topic and `{L}` the target language name:

> STORY: *Write a simple everyday story of 4 short sentences about {t}. Use simple words a child knows. Give it a clear, natural ending. Output only the story in {L}.*
>
> COMPARE: *Write 3 simple sentences that compare everyday things related to {t}, using words like bigger, smaller, faster, older, heavier. Output only the sentences in {L}.*
>
> COREF: *Write 3 simple sentences about {t} that use pronouns (he, she, it, they) clearly referring to specific people or things. Output only the sentences in {L}.*
>
> FACT: *Write 4 simple true everyday facts about {t} that a young child would understand. Output only the facts in {L}.*

Every prompt issued produced one accepted sample, so the counts are 6,250 per template per language, 25,000 per language, and 75,000 in total. The parallel data contributes 240,952 sequences from Tatoeba and 298,684 from OPUS-100, split 319,296 en–nl and 220,340 en–zh. The 143 topics are:

> the kitchen, animals, the weather, school, a birthday party, the beach, cooking, gardening, a train trip, the market, sports, music, friendship, the forest, morning routine, a rainy day, the doctor, shopping, a picnic, the library, farm life, the city, a snowy day, helping at home, a lost dog, the playground, baking bread, a family dinner, the zoo, fixing a bike, the ocean, mountains, rivers, insects, birds, flowers, trees, vegetables, fruits, the moon, stars, the sun, seasons, autumn leaves, spring, summer holidays, winter, a new toy, building blocks, drawing pictures, painting, reading books, puzzles, the school bus, the classroom, a math lesson, recess, the science fair, a soccer match, swimming, riding bikes, jumping rope, hide and seek, grandma's house, grandpa's stories, cousins visiting, a baby brother, pets at home, a cat and a mouse, a dog and a bone, a bird and a nest, fish in a pond, rabbits, the bakery, the fire station, the police, a nurse, a farmer, a fisherman, a pilot, the post office, the bank, a restaurant, the supermarket, a toy shop, a camping trip, a boat ride, a plane journey, a road trip, visiting a museum, the circus, a magic show, a puppet show, a parade, fireworks, cleaning the room, washing dishes, folding clothes, setting the table, making the bed, planting seeds, watering plants, picking apples, raking leaves, shoveling snow, a thunderstorm, a foggy morning, a windy day, a hot afternoon, a cold night, sharing toys, making friends, saying sorry,

helping a neighbor, being kind, the alphabet, counting numbers, colors and shapes, days of the week, telling time, a broken toy, a spilled drink, a scraped knee, a lost mitten, a torn shirt, cooking soup, making a sandwich, baking cookies, pouring juice, eating breakfast, a butterfly, a bee, an ant, a spider, a ladybug, a frog, a turtle, a snail, the playground slide, the swings, the sandbox, the seesaw, the merry-go-round.

Generated examples follow, verbatim from the stored generation files. No example was edited for fluency, and the English gloss of the Chinese sample is provided for readers only:

> [en, STORY, *the alphabet*] *Leo held a big book with letters A to Z. He read the story about a cat and a dog. Then he smiled when he saw his name at the end. Leo felt very happy because he could read his own words.*
>
> [nl, COREF, *cooking*] *Hij kookt de pasta. Zij snijdt de tomaten. Het gerecht smaakt lekker.*
>
> [zh, STORY, *seasons*] 春天来了，小草从土里探出头。夏天到了，知了在树上大声唱歌。秋天到了，树叶变成了金黄色。冬天来了，雪花落在屋顶上，世界变得白茫茫的。
> *Chūntiān lái le, xiǎo cǎo cóng tǔ lǐ tàn chū tóu. Xiàtiān dào le, zhīliǎo zài shù shang dàshēng chànggē. Qiūtiān dào le, shùyè biànchéng le jīnhuángsè. Dōngtiān lái le, xuěhuā luò zài wūdǐng shang, shìjiè biàn de báimángmáng de.*
> Gloss: *Spring came and the grass poked its head out of the soil. Summer arrived and the cicadas sang loudly in the trees. In autumn the leaves turned golden. Winter came, snow fell on the roofs, and the world turned white.*

Parallel examples, shown as they appear in the training corpus:

> [Tatoeba, en–nl] `English: Mickey Mouse is a mouse. \n Dutch: Mickey Mouse is een muis.`
>
> [OPUS-100, en–nl] `English: Because you're where? \n Dutch: Omdat je waar bent?`
>
> [Tatoeba, en–zh] `Chinese: 我不信任汤姆而且汤姆也不信任我。\n English: I didn't trust Tom and he didn't trust me, either.` (*Wǒ bù xìnrèn Tāngmǔ érqiě Tāngmǔ yě bù xìnrèn wǒ.*)

F Statistical tests

Table 17 lists every seed-matched test of the data-design study, recomputed in a single pass from the seed-level scores shown in Table 11. Augmentation uses four matched seeds; MTP compares Augmented + MTP with Augmented on seeds 7, 99, and 123. Within each intervention the four metrics form one Holm family. Intervals are unadjusted 95% t intervals for the paired mean difference.

The allocation-study differences, intervals, and Holm-adjusted values retain the precision of the reported analyses in §5.2–6. Raw p values are approximate recomputations from the rounded seed-level scores in Appendix B. These rounded inputs do not always reproduce the last printed digit of the retained statistics. Each language forms its own Holm family of three fixed-mixture contrasts.

G Reproducibility index

Table 18 identifies the training runs behind the data-design results. Corpus hashes show the first 16 hexadecimal digits of SHA-256; tokenizer hashes in Table 18 show the first eight. The two corpora are `f9917fdfa47ac9ef` (baseline) and `c700f38d1e08da64` (augmented). All runs use 10 epochs and 16,384 tokens per optimizer step; step counts differ because the corpora differ in size. The tokenizer column shows that the baseline and augmented conditions do not share a vocabulary.

The submitted 213M model is released at https://huggingface.co/1ghub/babylm-2026-mtl-bigckpt, and the organizers' data-preparation code cited in §7.1 is at https://github.com/babylm-org/multilingual-babylm. Zero-shot and fine-tuning scores were produced with the official BabyLM multilingual evaluation pipeline at commit `6f825c29`. Fine-tuning uses a batch of 64, a learning rate of 5e−5, at most 10 epochs with per-epoch evaluation, early stopping with patience 3, and seed 12. The checkpoint kept is the one selected by that early-stopping criterion, not the last. FT22 averages the resulting 22 non-POS task–language scores.

Intervention	Metric	n	Δ	95% CI	raw p	Holm p
Augmentation	ZS16	4	+0.800	[.32, 1.28]	.0130	.0389
	English	4	+0.808	[−.17, 1.79]	.0789	.1578
	Dutch	4	+0.235	[−.29, .76]	.2474	.2474
	Chinese	4	+1.473	[1.24, 1.71]	.0003	.0011
+MTP	ZS16	3	+0.097	[−.62, .81]	.6188	1.000
	English	3	−0.090	[−1.62, 1.44]	.8242	1.000
	Dutch	3	+0.543	[−.29, 1.38]	.1079	.3450
	Chinese	3	−0.253	[−.60, .09]	.0863	.3450
Fixed mixtures	TASK-7, one family of three contrasts, $n = 3$					
	40/40/20 − 50/25/25		+0.250	[−.93, 1.43]	.46	.9191
	40/40/20 − 27.5/27.5/45		+0.514	[−.50, 1.52]	.16	.4792
	50/25/25 − 27.5/27.5/45		+0.265	[−1.17, 1.70]	.51	.9191
Dynamic	TASK-7, one family of three contrasts, $n = 3$					
	Fixed − inverse		+0.181	[−.88, 1.24]	.54	1.000
	Fixed − smoothed		+0.173	[−1.68, 2.03]	.73	1.000
	Inverse − smoothed		−0.009	[−1.16, 1.14]	.98	1.000
Fixed, English	50/25/25 − 40/40/20	3	+0.863	[−.81, 2.54]	.157	.157
	50/25/25 − 27.5/27.5/45	3	+2.487	[2.08, 2.90]	.0015	.0044
	40/40/20 − 27.5/27.5/45	3	+1.623	[.05, 3.19]	.047	.094
Fixed, Dutch	50/25/25 − 40/40/20	3	−1.642	[−3.54, .26]	.065	.130
	50/25/25 − 27.5/27.5/45	3	−0.091	[−1.57, 1.38]	.815	.815
	40/40/20 − 27.5/27.5/45	3	+1.551	[.53, 2.58]	.023	.069
Fixed, Chinese	50/25/25 − 40/40/20	3	+0.727	[−1.11, 2.56]	.230	.302
	50/25/25 − 27.5/27.5/45	3	−1.424	[−4.12, 1.27]	.151	.302
	40/40/20 − 27.5/27.5/45	3	−2.151	[−3.26, −1.05]	.014	.042

Table 17: All seed-matched tests reported in this paper. Two-sided paired t-tests. Augmentation uses seeds 42, 7, 99, 123; +MTP uses seeds 7, 99, 123, so those two families are not nested. The data-design rows are recomputed from Table 11. Allocation-study differences, intervals, and Holm values are retained as reported in §5.2–6. Their raw p values are approximate checks using the rounded scores in Tables 9, 10, and 12; the last printed digits need not agree exactly. Each language forms its own Holm family of three fixed-mixture contrasts.

H Pilot and corpus details

Tables 19 and 20 report the exploratory pilots; Table 21 records corpus composition. In the mixture pilot, Dutch varies by 1.273 points, against 4.680 for English and 3.255 for Chinese. The Dutch-high mixture did not exceed the equal mixture on Dutch. Its different tokenizer and training scale prevent this observation from explaining the main experiment's Dutch non-detection.

I Overlap audit details

We screened the training corpus against the evaluation sentences of the minimal-pair files distributed with the track: 67 BLiMP paradigms for English (59,875 pairs), 84 BLiMP-NL paradigms for Dutch (8,400 pairs), and 11 ZhoBLiMP paradigms for Chinese (3,300 pairs), or 119,750, 16,800, and 6,600 sentences counting both members of each pair. The Chinese files are a subset of ZhoBLiMP rather than the whole suite, so the Chinese audit is the narrowest of the three. Training lines and evaluation sentences were normalized with Unicode NFKC, case folding, and whitespace collapsing; Chinese was additionally stripped of whitespace. The audit has two passes. The first compares whole normalized strings for equality, so it does not depend on any length threshold. The second flags a line that shares a word 8-gram (English, Dutch) or a character 8-gram (Chinese) with an evaluation sentence and scores each flagged pair by character 5-gram Jaccard similarity, reporting counts at .90, .80, and .70.

Across 9,853,776 line–language checks the first pass returns three matching training lines, corresponding to two distinct evaluation sentences. These are one English line in the real BabyBabelLM text (*The Eye is closing.*) and two Chinese lines in the parallel data that are copies of the same five-character sentence. The second pass returns no additional pair at any threshold. We retained the original evaluation sets. Scores were not recomputed with the corresponding pairs excluded. No synthetic line matched any evaluation sentence. Both reported exact matches are shorter

Condition	Corpus	Tokenizer	Seeds	Steps
Baseline 110M	baseline	`235dfa07`	42, 7, 99, 123	76,440
Augmented 110M	augmented	`bba937b7`	42, 7, 99, 123	87,820
+MTP 110M	augmented	`bba937b7`	7, 99, 123	87,820
Submitted 213M	augmented	`bba937b7`	42	87,820
Second 213M	modified*	`cd59b082`	42	88,660

Table 18: Training runs behind the seed comparisons and the two evaluated models. *This variant replaces part of the real text with additional synthetic text at matched language proportions, keeping the parallel text intact. The table does not cover the parameter-count runs of Table 7 or the single-run ablations of Table 8.

Tokenizer	Vocab.	Params	Task-7	Lang-3	All-16
BPE	16,000	97.737M	71.806	71.540	54.651
Unigram	16,000	97.737M	71.558	71.271	54.606
MorPiece	39,697	115.937M	73.143	72.940	55.843

Table 19: Tokenizer pilot, one training seed per condition, equal language mixture. Vocabulary and total parameter count vary with the tokenizer.

than the 8-gram window, so they are found by the first pass alone; conversely, the near-duplicate counts are conditional on the 8-gram candidate filter, and we did not measure its recall. These counts therefore do not establish the absence of overlap. The audit covers the distributed files named above and not the other ZS16 benchmarks (MultiBLiMP, XCOMPS, Winogrande, XStoryCloze, and HellaSwag), the fine-tuning suites, or GlobalPIQA.

Mixture (en/nl/zh)	English	Dutch	Chinese	Task-7	Lang-3
40/40/20‡	71.470	69.103	58.955	66.880	66.509
27.5/27.5/45‡	68.335	69.327	61.610	66.839	66.424
50/25/25‡	71.340	68.527	59.565	66.770	66.477
25/50/25	70.155	69.330	58.370	66.434	65.952
Equal (1/3 each)	67.960	69.800	59.840	66.429	65.867
30/30/40	68.245	68.927	60.650	66.367	65.941
25/25/50	66.790	68.633	61.625	66.104	65.683

Table 20: Mixture selection pilot: one seed per mixture at the 25M-word proxy with the MorPiece tokenizer, ordered by TASK-7. ‡Promoted to the three-seed experiment of §5. Scores are on the 25M-word proxy scale and are not comparable to Table 19 or Table 2, both of which use the full budget.

Source	Lang	Documents	Raw words	Eq. words
BabyBabelLM	en	3,698,322	34.000M	34.000M
(real)	nl	4,417,520	34.703M	33.000M
	zh	583,662	30.692M	32.792M
Synthetic	en	25,000	0.914M	0.914M
(teacher)	nl	25,000	0.787M	0.748M
	zh	25,000	1.268M	1.355M
Parallel	en	539,636	2.271M	2.271M
(Tatoeba,	nl	—	1.242M	1.181M
OPUS-100)	zh	—	1.340M	1.432M
Baseline total		8,699,504	99.395M	99.792M
Augmented total		9,314,140	107.217M	107.692M

Table 21: Corpus composition. Raw words are whitespace tokens for English and Dutch and CJK characters for Chinese; equivalent words apply the official 2026 Byte Premiums from the organizers' coefficient table (English 1.0, Dutch 1.0516, simplified Chinese 0.93597; the call for papers lists 0.9894, the coefficient given there for traditional Chinese). The parallel rows are 539,636 training sequences built from 269,818 translation pairs, one sequence per direction; their word counts are given once per pair, so per-epoch exposure is twice the listed value. The equivalent-word column follows the construction-time word proxy of Table 5, which is not the quantity the track caps.

Feeding BabyLMs Macaroni: Code-Switching Curricula Cause Cross-Lingual Convergence

Dries Rooryck[*1] **Alex Cai**[*1] **Yonatan Belinkov**[1,2] **David Alvarez-Melis**[1] **Kianté Brantley**[1]

[1]Kempner Institute, Harvard University, Cambridge, MA, USA
[2]Technion – Israel Institute of Technology, Haifa, Israel

dries_rooryck@college.harvard.edu, adzcai@g.harvard.edu

Abstract

Children in multilingual communities often *code-switch*, using multiple languages in a single utterance. Can we induce cross-lingual alignment in language models by training on code-switched text? We pretrain small decoder-only transformers on two 100M-word multilingual corpora: a base corpus formed by mixing the English, Dutch, and Chinese BabyBabelLM datasets, and a corpus generated from it by inserting word- and sentence-level code-switching using an LLM. We find that training on code-switched data aligns the representations of parallel text, particularly across different scripts, and that this alignment persists through training on monolingual documents. Under a learning curriculum that progresses from word-level code-switching, to sentence-level code-switching, to monolingual documents, models trained on code-switched data outperform baselines trained without it on the BabyLM evaluation suite. Our work characterizes code-switching curriculum learning as an effective data augmentation method for multilingual pretraining. We release our code, data, and models at https://github.com/drooryck/multilingual-macaroni.

1 Introduction

Modern large language models (LLMs) are trained on orders of magnitude more language data than a person typically is exposed to in their lifetime. Yet children in any country communicate fluently, often in multiple languages, after hearing under the equivalent of just 100 million English words (Choshen et al., 2026). The BabyLM competition seeks to train computational language models under such developmentally plausible data constraints. This year's multilingual track asks participants to prepare the highest possible performing model across English, Dutch, and Chinese using the above total word budget.

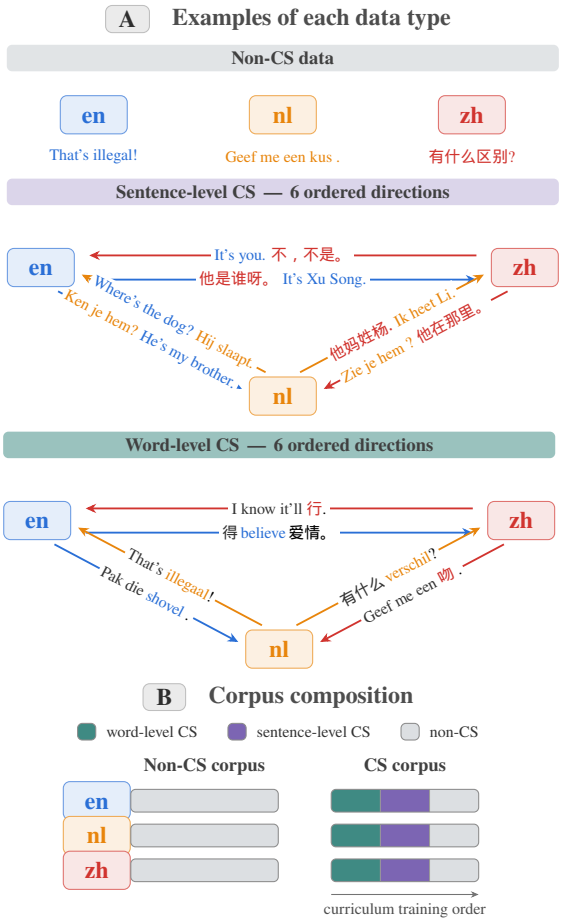


Figure 1: **Composition of our synthetic code-switched corpus.** **A**: We synthesize code-switched (CS) data by prompting an LLM to translate words or sentences from documents in the monolingual competition datasets. Examples shown are from our data. **B**: We train models on two corpora. Both are one-third English, Dutch, and Chinese. The non-CS corpus comprises monolingual source documents, while the CS corpus splits each language's subset equally into word-level CS, sentence-level CS, and monolingual documents, shown left to right in curriculum training order.

We take inspiration from infants raised in multilingual environments, who, within a single utterance, will use words from multiple languages,

[*]Equal contribution.

Proceedings of the Second BabyLM Workshop, pages 750–766
October 29, 2026 ©2026 Association for Computational Linguistics

matrix	emb.	Word	Sent.	Non-CS	Total
English	→NL	5.2	6.0	11.1	33.4
	→ZH	5.9	5.1		
Dutch	→EN	5.8	6.3	11.7	35.1
	→ZH	5.9	5.4		
Chinese	→EN	5.7	5.8	10.4	31.2
	→NL	4.7	4.6		
Total		33.2	33.2	33.2	99.6

Table 1: Composition of the CS corpus by *matrix language* (the source document language) and *embedded language* (the language of the inserted words or translated sentences) in millions of byte-premium-adjusted words (see Appendix A). Cells may not sum exactly to totals due to rounding.

a phenomenon known as **code-switching** (CS) (Volterra and Taeschner, 1978), or, rarely, "macaronic language".[1] While children code-switch naturally during language acquisition, might training neural language models intentionally on CS data help them align their representations of parallel text? This hypothesis has driven numerous schemes for multilingual pretraining (Yang et al., 2020; Li et al., 2024; Wang et al., 2025) and finetuning (Qin et al., 2020; Zheng et al., 2024; Yoo et al., 2025; Wang et al., 2026).

Training on CS data, however, introduces a complication: the trained model is not typically intended to generate CS text, but rather monolingual text in each of multiple languages. Code-switching curriculum learning (Yoo et al., 2025) presents a promising solution: first train on word-level CS, then sentence-level CS, and finally monolingual documents. This ordering imitates the stages of a human learning a new language: first one learns new words, then incorporates whole sentences into their speech, and finally speaks fully in the new language.

Our work adapts code-switching curriculum learning to data-constrained multilingual pretraining. We investigate the causal effects of pretraining on CS data by comparing against baselines trained on the same documents kept entirely in the original language. Our research is organized along the following questions:

1. Does CS training improve downstream performance on the BabyLM evaluation suite?
2. Does CS training align the model's internal representations across languages?
3. Does the model represent words seen embedded in CS contexts differently than words seen only in monolingual contexts?

Our contributions, respectively, are that for models trained on CS data:

1. Performance on the BabyLM evaluation suite improves when trained under a curriculum but not under shuffled ordering (§5);
2. Representations of parallel text align, and under curriculum ordering, the alignment persists through training on monolingual documents (§6);
3. Embedded words align more closely to their translations than words that only appear in monolingual contexts (§7).

We release our models[2], corpora[3], and code[4].

2 Related Work

Developmentally plausible language modeling. The first three iterations of the BabyLM challenge (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025) established data-constrained language acquisition as an interdisciplinary question of interest to machine learning researchers, linguists, and cognitive scientists. Relevant past submissions include a negative finding for cognitively-inspired curriculum learning (Diehl Martinez et al., 2023) and a benchmark for detection of ungrammaticality, including unnatural CS, in second-language acquisition (Gao et al., 2025). Zeng et al. (2026) also study the effect of various synthetic data mixtures on bilingual BabyLMs. Our work adopts the motivation, constraints, and evaluation suite of this year's challenge (Choshen et al., 2026), which introduces a multilingual track based on the BabyBabelLM corpora (Jumelet et al., 2026a).

CS as data augmentation for multilingual training. While attempts to model human CS predate the deep learning era (Chan et al., 2006; Franco and Solorio, 2007), recent works generate CS as a data augmentation strategy for multilingual training. Some works procedurally generate CS data using bilingual dictionaries (Qin et al., 2020; Zhu

[1] Some authors use the terms "code-switching" and "code-mixing" for different phenomena. We follow Yoo et al. (2025) in distinguishing between word-level and sentence-level code-switching. See also https://en.wikipedia.org/wiki/Macaronic_language.

[2] https://huggingface.co/drooryck/multilingual-macaroni-models

[3] https://huggingface.co/datasets/drooryck/multilingual-macaroni-corpus

[4] https://github.com/drooryck/multilingual-macaroni

et al., 2023; Zheng et al., 2024; Feng et al., 2022). Others prompt LLMs, as we do, which exhibit more naturalistic patterns (Kuwanto et al., 2026; Wang et al., 2025). While the above works demonstrate benefits of CS training, Shao et al. (2026) suggest that CS data is less important than parallel data for a model's ability to translate (but that, surprisingly, neither kind of multilingual data is required for other cross-lingual understanding and reasoning tasks). However, their category of CS documents does not match our definitions of word- and sentence-level CS, and ultimately both our works stress the importance of fine-grained lexical alignment for translation. Other work has applied CS for cross-lingual transfer at inference time (Yoo et al., 2026), in chain of thought (Wang et al., 2026), or for instruction tuning (Asano et al., 2026). Our work most closely follows code-switching curriculum learning (Yoo et al., 2025). Whereas code-switching curriculum learning seeks to fine-tune a pretrained model, our study focuses on pretraining in the data-constrained regime and attempts a closer analysis of model representations.

Cross-lingual representation alignment. Recent papers characterize LLMs' ability to generalize across languages despite being trained on mostly monolingual documents. Interpretability studies find shared model components or subspaces across languages (Conneau et al., 2020; Dufter and Schütze, 2020; Chang et al., 2022) or investigate how cross-lingual alignment emerges throughout the training process (Blevins et al., 2022; Wang et al., 2024). Token overlap between languages tends to improve cross-lingual generalization (Pires et al., 2019; Kallini et al., 2025), as does typological similarity (Muller et al., 2023; Longpre et al., 2026). See Hämmerl et al. (2024) for a survey on cross-lingual alignment. Our work contributes a controlled study of how code-switching curriculum learning affects cross-lingual alignment.

3 Synthetic Code-Switched Data Generation

We construct two distinct corpora that both satisfy BabyLM's data budget: one with code-switched (CS) data and the other without (non-CS). Figure 1 illustrates their composition and Figure 5 (Appendix A) gives longer examples. See Appendix A for detailed accounting. We also create four additional corpora as experimental controls.

Monolingual-document (non-CS) corpus. We begin with the English, Dutch, and Chinese BabyBabelLM datasets (Jumelet et al., 2026a). We sample a third of each and combine them to obtain our base non-CS corpus of monolingual documents.

CS corpus. We begin with the English third of the non-CS corpus (33.4M words) and separate it further into thirds: one kept monolingual, one for word-level CS (where English is the matrix language), and one for sentence-level CS (where half of the sentences in each document are translated). The monolingual third is kept as-is. The word-level third is divided into one half for Dutch insertions and one half for Chinese insertions, where we use an instruction-tuned LLM to translate a subset of words to the embedded language. We similarly bisect the sentence-level third and instruct the LLM to translate, in-place, roughly half of the sentences in each document to the embedded language. We then repeat this *mutatis mutandis* for the Dutch and Chinese subsets. See Table 1 for an exact breakdown of the final corpus composition. We verify the quality of generated code-switched text through programmatic checks and through manual verification of sample documents by bilingual English-Dutch and English-Chinese speakers. See Appendix A for our quality verification pipeline and Appendix B for LLM choice and prompts.

Corpora for controls. We construct four corpora to isolate the contribution of various experimental factors.

1. **Word-level CS only**: we revert the sentence-level CS third of the CS corpus to the original monolingual documents.

2. **Sentence-level CS only**: we similarly revert the word-level CS third.

3. **Word salad**: for a third of the original non-CS corpus, we replace roughly 35% of tokens of three or more letters, selected randomly, with words sampled from the embedded language's monolingual data. For Chinese source text, we insert words instead of replacing.

4. **Document translation**: we take a third of documents in the original non-CS corpus and append to them an LLM-generated translation. We then discard a third of these documents to keep the word budget consistent.

Ordering	Corpus	Leaderboard	Fine-tune avg.	Zero-shot avg.	Selected tasks (accuracy, avg. over languages)				
					SIB-200	INCLUDE	Global PIQA	BMLAMA	MultiBLiMP
Shuffled	non-CS	46.44±0.25	42.27±0.48	51.77±0.17	71.85±1.64	29.02±2.61	36.84±1.24	21.60±2.56	86.91±0.72
	CS	46.55±0.47	42.33±0.79	51.95±0.14	**73.33±1.23**	28.96±2.13	35.86±0.69	19.87±1.97	**88.05±0.87**
Curriculum	non-CS	46.36±0.37	42.26±0.74	51.60±0.22	72.33±1.83	27.79±3.33	36.85±0.60	21.11±1.56	86.52±0.46
	CS	**46.72±0.25**	**42.81±0.42**	51.72±0.13	**74.27±1.20**	29.13±1.56	36.66±0.78	20.31±2.42	**87.17±0.50**
Shuffled (controls)	word-level only	46.59±0.24	42.60±0.38	51.68±0.25	73.44±1.70	28.63±2.54	35.79±1.08	20.96±1.80	87.02±0.73
	sentence-level only	46.70±0.29	42.60±0.45	51.96±0.25	72.52±1.48	29.19±1.91	36.25±1.53	21.35±1.35	87.92±0.64
	document translation	46.56±0.38	42.43±0.63	51.83±0.24	72.75±0.94	28.57±3.60	36.20±0.79	20.02±1.97	87.89±0.75
	word salad	46.08±0.34	42.14±0.63	51.07±0.20	72.42±1.90	28.13±2.24	36.39±0.78	18.47±2.69	85.82±0.87

Table 2: Accuracy on the BabyLM evaluation suite (mean ± SD) across eight seeds per condition. We select five tasks: the two language-averaged tasks with the largest positive gap between the curriculum CS and non-CS models (SIB-200 $+1.94$, INCLUDE $+1.34$), the two with the most negative (BMLAMA -0.80, Global PIQA -0.20), and MultiBLiMP as a grammatical reference task. Bold cells mark statistical significance on a two-sided paired t-test between the CS and non-CS models (uncorrected for multiple hypotheses). Full per-task results are in Table 3 (Appendix C).

See Appendix A for verbatim examples from the word salad and document translation controls.

4 Experimental Setup

Our main experiments vary the training corpus (CS vs non-CS) and the data ordering (shuffled vs a three-stage curriculum). Including our four control corpora yields 8 distinct training setups in total. For each setup, we train 8 random seeds, which affect the model initialization and document order.

Architecture and tokenizer. All models use the GPT-2-small architecture from the BabyLM multilingual baseline[5] (12 layers, hidden size 768, 12 heads, context length 1024). We train a byte-level BPE tokenizer of vocabulary size $16,384$, matching the baseline implementation, on the monolingual-document corpus.

No padding. The BabyBabelLM corpora contain many short documents, so to avoid excessive padding tokens, we preprocess by concatenating all documents (separated by end-of-text tokens) and splitting according to the model's context length.

Optimizer. Following the competition baselines, we train models using Adam (Kingma and Ba, 2015) without weight decay, a batch size of 16, and a learning rate schedule that starts at 0, warms up for 1% of total training steps to 5×10^{-5}, and cosine-decays to zero.

Curriculum learning. When training on the CS corpus, we compare between a learning curriculum and the default ordering (10 epochs on the shuffled CS corpus). The curriculum consists of training first for 10 epochs over the word-level CS third of our CS corpus, then 10 epochs on the sentence-level CS third, and finally 10 epochs on the non-CS third, resetting the learning rate schedule in each stage. As for the baselines trained on the non-CS corpus, we also apply these two orderings and their respective learning rate schedules, maintaining the document IDs but using the original monolingual documents instead of the synthesized CS versions. This results in four (corpus, ordering) pairs: CS/shuffled, CS/curriculum, non-CS/shuffled, non-CS/curriculum. The models trained on our control corpora all follow shuffled ordering.

Evaluation. We evaluate models on the BabyLM evaluation suite, which tests for grammatical fluency, cognitive plausibility, and model adaptability (via fine-tuning). The full list of tasks is in Appendix C. The English/Dutch/Chinese task suites differ slightly, so all reported averages are of the per-language scores.

5 Results

Table 2 summarizes performance on a subset of evaluation tasks. We report per-task performance across the entire suite in Table 3 (Appendix C) and also held-out bits per byte in Table 4 (Appendix C).

Code-switching under a curriculum scores highest. Between our four corpus/ordering pairings, we see that CS/curriculum performs highest on average (46.72), followed by CS/shuffled, then non-CS/shuffled, and finally non-CS/curriculum. Each of our training setups outperforms, on average across seeds, the BabyLM baseline (Choshen et al.,

[5] https://github.com/babylm-org/multilingual-training

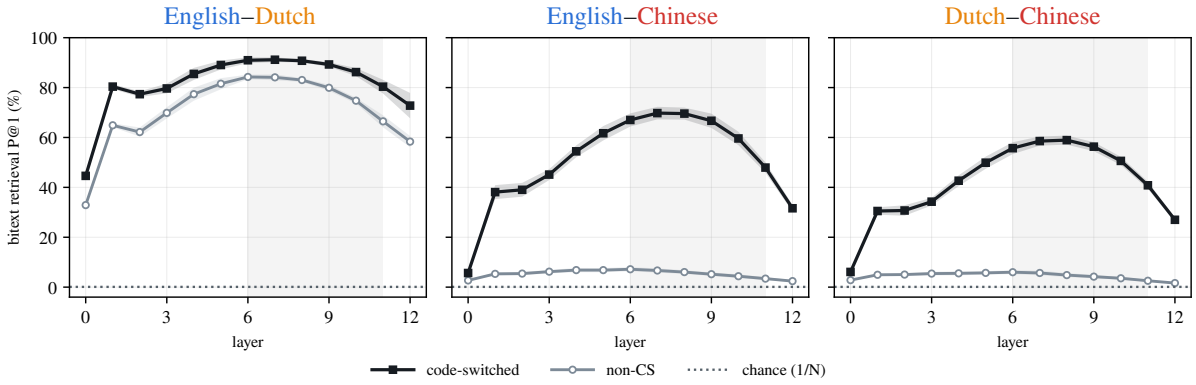


Figure 2: Layerwise bitext retrieval precision@1 (§6) on 997 parallel sentences from FLORES+ across English, Dutch, and Chinese. We compare CS models against non-CS models (ignoring data ordering) and plot the mean ± 1 SD across the 16 runs per group. Retrieval in non-CS models fails for language pairs that differ in script. See Figure 6 (Appendix D) for the median percentile rank of the translation out of the candidate set. Shading marks layers 6–11, which the reported P@1 averages in other experiments are taken over.

2026), which scores 45.94 on the leaderboard[6] We check for statistical significance with two-sided paired t-tests over all $\binom{4}{2} = 6$ pairwise comparisons of the ordering/corpus pairs, applying Holm correction (Holm, 1979) for the familywise error rate. Only the CS/curriculum improvement over non-CS/curriculum is significant ($p = 0.016$); the gap between CS/curriculum and non-CS/shuffled is not ($p = 0.41$), nor is any other comparison.

Fine-tuning tasks under the curriculum. For the models trained with a curriculum, most of the CS models' advantage over the non-CS models comes from the fine-tuning tasks. This suggests that the internal representations of CS models may be more generalizable toward various downstream tasks than those of models trained without CS. We investigate further in §6. However, this boost does not seem to hold under shuffled ordering, where the CS and non-CS models' scores are within noise of each other for both zero-shot and fine-tuning averages.

6 Cross-Lingual Representation Alignment

Does training on code-switched data cause a model to align the representations of parallel sentences? We measure this by seeing whether a sentence and its translation have closer representations in CS models than in non-CS models.

[6]Model `BabyLM-2026-Baseline-GPT2-en_nld_zho_equal` on the leaderboard at https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026. Accessed 17 September 2026.

Cross-lingual alignment metric. Formally, we measure a model's cross-lingual representation alignment using **bitext retrieval precision@1** (P@1) (Artetxe and Schwenk, 2019; Pires et al., 2019; Hu et al., 2020). Given a sentence s^A in language A, and a list of sentences $s^B_1, \ldots, s^B_N$ in language B containing the translation of s^A, we say the model successfully retrieves the translation if it is the original sentence's nearest neighbor in terms of the cosine similarity between their representations. We represent a sentence by mean-pooling the residual stream vectors of its tokens. We additionally apply cross-domain similarity local scaling (CSLS, Lample et al., 2018) to the cosine similarity values to account for the "hubness" issue of naive nearest neighbors. Given a set of sentence pairs $(s^A_1, s^B_1), \ldots, (s^A_N, s^B_N)$, the bitext retrieval precision is the average success rate across all sentences in both directions. We use $N{=}997$ parallel sentences across English, Dutch, and Chinese from the FLORES+ `dev` split (NLLB Team et al., 2024; Goyal et al., 2022). See Appendix D for details.

Code-switched training aligns parallel sentences. Across all layers and language pairs, the CS model better aligns the sentence representations to their translations' representations (Figure 2): between English and Chinese under curriculum ordering, the CS model retrieves the correct translation 61.4% of the time (layers 6–11, 8-seed mean), compared to only 5.5% of the time in the non-CS baseline (Table 6, Appendix D). A similar gap holds for bitext retrieval between Dutch and Chinese. Other training-free measures, in-

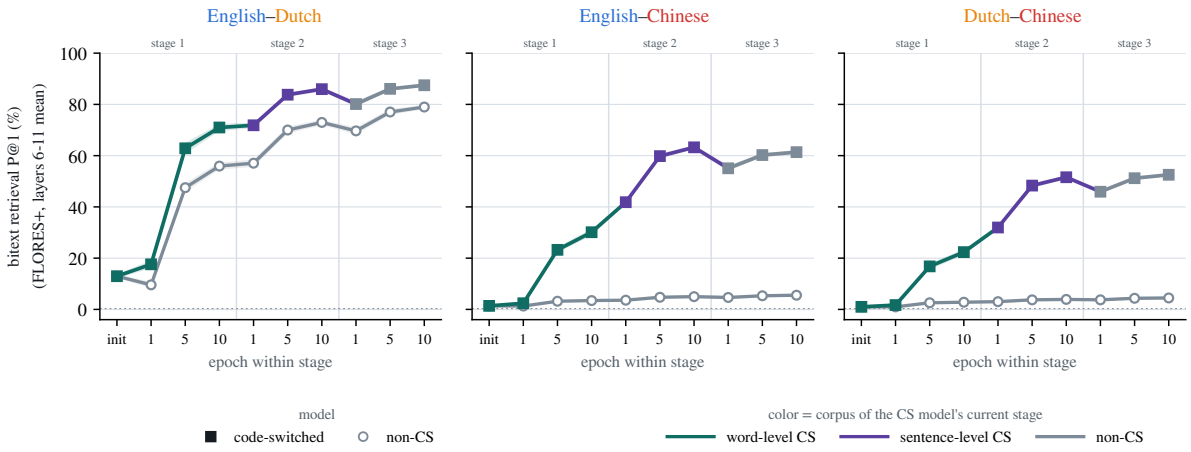


Figure 3: Sentence-level alignment across the learning curriculum. Alignment improves throughout CS training and persists through training on monolingual documents. The color of the CS model's line indicates the CS type of the current stage.

cluding linear centered kernel alignment (CKA), a sentence-level cosine gap, centroid distance, and the MEXA alignment score all corroborate this finding for the cross-script language pairs (Table 5, Appendix D).

Coherence of CS matters. Must code-switching be coherent in order to improve alignment, or is the gain driven simply by co-occurrence across tokens? Our "word salad" control, where words are incoherently switched for foreign-language words, does not induce this same alignment (Figure 7, Appendix D), suggesting that grammatically and semantically coherent code-switching is crucial.

Cross-lingual alignment persists through non-CS training. We measure bitext retrieval precision across model checkpoints (Figure 3). Cross-lingual alignment improves during the word-level and sentence-level CS stages and is maintained even through the non-CS training stage.

7 Representations of Code-Switched Words

Does CS training improve the representations of only the words seen in embedded contexts, or does the cross-lingual alignment extend across the model's entire vocabulary?

Embedded vs. never-embedded words. We call a word *embedded* if it was ever inserted as a foreign-language word during word-level CS, and *never-embedded* if it was neither embedded nor replaced by an embedded word. We exclude replaced words since they have systematically lower frequency in the CS corpus due to being replaced. We compute a word w's representation as follows. We first collect a set of sentence contexts from the original corpus that contain w. We then compute a forward pass and take the mean of the residual stream vectors across the token positions that constitute w, across the set of sentences, and across layers 6–11.

Median percentile rank metric. Similarly to the sentence-level bitext retrieval metric (§6), for each word w^A in language A, we construct a set of words $w_1^B, \ldots w_N^B$ in language B containing the translation w_*^B of w^A. We then rank the set according to the same CSLS-adjusted cosine similarity metric, with regards to the word representations described above, and record the **percentile rank** of w_*^B. (This contrasts with the precision@1 metric where we would only check if w_*^B is the most similar word to w^A.) We compute these percentile ranks across $N = 1{,}102$ embedded/non-embedded word pairs and, for each of the 4 language directions for which we have an openly accessible bilingual dictionary, plot the median in Figure 4. See Appendix E for details on choosing the candidate sets and labeling translation pairs.

Never-embedded words also become aligned by CS training. Relative to the non-CS baseline, CS training raises the median percentile rank of the translation by $+0.064$ for embedded words and $+0.045$ for never-embedded words (Figure 4). The English–Dutch translations are already closely aligned in both the CS and non-CS models. Here in word-level alignment, as in sentence-level align-

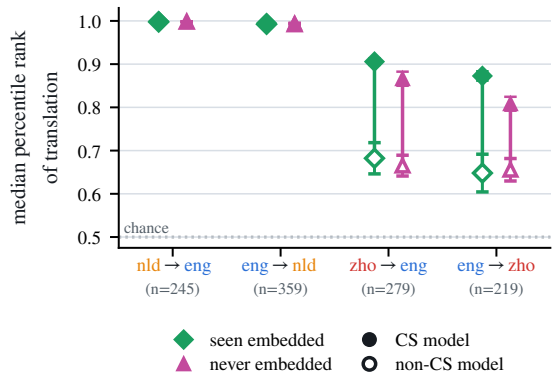


Figure 4: Median percentile rank of a word's correct translation among a set of target-language words matched by part of speech and frequency (Appendix E). The number of word pairs for each language pair is shown in parentheses.

ment (§6), we see that CS training aligns representations especially across distinct scripts.

8 Conclusion

We train and analyze small autoregressive language models on code-switched curriculum learning under BabyLM competition constraints (10 epochs on 100M words shared between Dutch, English, and Chinese). When trained in a three-stage learning curriculum, from word-level code-switching to sentence-level switching to monolingual documents, a model trained on code-switched data achieves a 0.36-point gain on the BabyLM evaluation suite, on average across eight seeds, relative to a baseline trained on a matched corpus of monolingual documents (RQ1, §5). However, under a randomly shuffled data ordering, there is no statistically significant improvement. We show that code-switching in pretraining aligns the representations of parallel sentences (RQ2, §6) and additionally show that embedded words become more closely aligned to their translations than words never seen embedded in a code-switched context (RQ3, §7). We release our corpora and training code to support future research on multilingual data composition and data ordering at a fixed data budget.

Limitations

Language coverage. We study only trilingual models in English, Dutch, and Chinese so as to match the BabyLM challenge evaluation suite. Whether the curriculum and the alignment mechanism generalize to more distant or lower-resource pairs, or to more than three languages at once, is untested.

LLM-generated code-switching. Our LLM-driven synthetic CS corpus generation method would be expensive to scale to larger datasets. It also results in a confound whereby the sentence-level CS data and the document translation control corpus risk having a different text distribution than BabyBabelLM, which might thus be responsible for performance improvements rather than the code-switching intervention itself.

Model and budget scope. All of our results are for a single model size and architecture (∼98M-parameter GPT-2 at the 100M-word budget). Future work could investigate whether the benefits of training on code-switched text persist in larger models and data budgets.

Ethics Statement

Data provenance and released artifacts. All training text derives from the BabyBabelLM corpora (Jumelet et al., 2026a), distributed under the BabyLM community's access terms. We did not collect data from human subjects; manual review of generated samples was done by the joint first authors. Our models are ∼98M-parameter research artifacts with no instruction tuning or safety alignment. We release them and our corpora under the MIT license.

Compute. The experiments in this paper required training 64 models, each on a single NVIDIA H100 80GB GPU. Pretraining one model takes ∼2.3 GPU-hours and evaluating a model takes an additional ∼0.8. This adds up to roughly 200 GPU-hours in total, excluding the LLM inference used to synthesize the corpora, the checkpoint analyses of §6–§7, and re-run jobs.

Acknowledgments

We thank Nihal Nayak, Sara Kangaslahti, Greta Tuckute, and the Harvard ML Foundations Group for conversations regarding multilingual models. AC is grateful to be funded by the Kempner Graduate Fellowship. Computation was performed on the Kempner Institute's H100 cluster at Harvard University. We wrote most of our analysis and plotting code with help from Anthropic's Claude LLM.

References

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O. Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Shunta Asano, Jeonghun Baek, and Toshihiko Yamasaki. 2026. Beyond bilingual transfer: Multilingual code-switching in instruction tuning. *arXiv preprint arXiv:2605.29414*. Preprint; no peer-reviewed version as of September 2026.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The Belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

Terra Blevins, Hila Gonen, and Luke Zettlemoyer. 2022. Analyzing the mono- and cross-lingual pretraining dynamics of multilingual language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3575–3590, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Joyce Y. C. Chan, P. C. Ching, Tan Lee, and Houwei Cao. 2006. Automatic speech recognition of Cantonese-English code-mixing utterances. In *Proceedings of Interspeech 2006 – ICSLP*, Pittsburgh, PA, USA. ISCA.

Tyler A. Chang, Catherine Arnett, et al. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *arXiv preprint arXiv: 2510.24081*.

Tyler A. Chang, Zhuowen Tu, and Benjamin K. Bergen. 2022. The geometry of multilingual language model representations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 119–136, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gül, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gül, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv: 2602.20092*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2): 255–308.

DeepSeek-AI. 2026. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*.

Richard Diehl Martinez, Zébulon Goriely, Hope McGovern, Christopher Davis, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2023. CLIMB – curriculum learning for infant-inspired model building. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 112–127, Singapore. Association for Computational Linguistics.

Philipp Dufter and Hinrich Schütze. 2020. Identifying elements essential for BERT's multilinguality. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4423–4437, Online. Association for Computational Linguistics.

Yukun Feng, Feng Li, and Philipp Koehn. 2022. Toward the limitation of code-switching in cross-lingual transfer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5966–5971, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Juan Carlos Franco and Thamar Solorio. 2007. Baby-steps towards building a Spanglish language model. In *Computational Linguistics and Intelligent Text Processing: 8th International Conference, CICLing 2007*, volume 4394 of *Lecture Notes in Computer Science*, pages 75–84, Berlin, Heidelberg. Springer.

Yuan Gao, Suchir Salhan, Andrew Caines, Paula Buttery, and Weiwei Sun. 2025. BLiSS: Evaluating bilingual learner competence in second language small language models. In *Proceedings of the First BabyLM Workshop*, pages 160–174, Suzhou, China. Association for Computational Linguistics.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The FLORES-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Katharina Hämmerl, Jindřich Libovický, and Alexander Fraser. 2024. Understanding cross-lingual Alignment—A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10922–10943, Bangkok, Thailand. Association for Computational Linguistics.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schütze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81, Vienna, Austria. Association for Computational Linguistics.

Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4411–4421. PMLR.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Julie Kallini, Dan Jurafsky, Christopher Potts, and Martijn Bartelds. 2025. False friends are not foes: Investigating vocabulary overlap in multilingual language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 21138–21154, Suzhou, China. Association for Computational Linguistics.

Amir Hossein Kargaran, Ali Modarressi, Nafiseh Nikeghbal, Jana Diesner, François Yvon, and Hinrich Schütze. 2025. MEXA: Multilingual evaluation of English-centric LLMs via cross-lingual alignment. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 27001–27023, Vienna, Austria. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 3519–3529. PMLR.

Garry Kuwanto, Chaitanya Agarwal, Genta Indra Winata, and Derry Tanti Wijaya. 2026. Linguistics theory meets LLM: Code-switched text generation via equivalence constrained large language models.

In *Proceedings of the 1st Workshop on Computational Developmental Linguistics (CDL)*, pages 1–14, San Diego, CA, USA. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of the 6th International Conference on Learning Representations*.

Jiahuan Li, Shujian Huang, Aarron Ching, Xinyu Dai, and Jiajun Chen. 2024. PreAlign: Boosting cross-lingual transfer by early establishment of multilingual alignment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10246–10257, Miami, Florida, USA. Association for Computational Linguistics.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2020. On the language neutrality of pre-trained multilingual representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1663–1674, Online. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, and 2 others. 2022b. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2026. A systematic assessment of language models with linguistic minimal pairs in Chinese. *Transactions of the Association for Computational Linguistics*, 14:755–771.

Shayne Longpre, Sneha Kudugunta, Niklas Muennighoff, I-Hung Hsu, Isaac Caswell, Alex Pentland, Sercan Arik, Chen-Yu Lee, and Sayna Ebrahimi. 2026. ATLAS: Adaptive transfer scaling laws for multilingual pretraining, finetuning, and decoding the curse of multilinguality. In *Proceedings of the 14th International Conference on Learning Representations*.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, 3rd edition. Lawrence Erlbaum Associates, Mahwah, NJ.

Benjamin Muller, Deepanshu Gupta, Jean-Philippe Fauconnier, Siddharth Patwardhan, David Vandyke, and Sachin Agarwal. 2023. Languages you know influence those you learn: Impact of language characteristics on multi-lingual text-to-text transfer. In *Proceedings of The 1st Transfer Learning for Natural Language Processing Workshop*, volume 203 of *Proceedings of Machine Learning Research*, pages 88–102. PMLR.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2023. Cross-lingual consistency of factual knowledge in multilingual language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10650–10666, Singapore. Association for Computational Linguistics.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. CoSDA-ML: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3853–3860. International Joint Conferences on Artificial Intelligence Organization.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(86):2487–2531.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, et al. 2025. INCLUDE: Evaluating multilingual language understanding with regional knowledge. In *Proceedings of the 13th International Conference on Learning Representations*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740. AAAI Press.

Jiandong Shao, Raphael Tang, Crystina Zhang, Karin Sevegnani, Pontus Stenetorp, Jianfei Yang, and Yao Lu. 2026. The role of mixed-language documents for multilingual large language model pretraining. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 36807–36818, San Diego, California, United States. Association for Computational Linguistics.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Virginia Volterra and Traute Taeschner. 1978. The acquisition and development of language by bilingual children. *Journal of Child Language*, 5(2):311–326.

Hetong Wang, Pasquale Minervini, and Edoardo Ponti. 2024. Probing the emergence of cross-lingual alignment during LLM training. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12159–12173, Bangkok, Thailand. Association for Computational Linguistics.

Zhijun Wang, Jiahuan Li, Hao Zhou, Rongxiang Weng, Jingang Wang, Xin Huang, Xue Han, Junlan Feng, Chao Deng, and Shujian Huang. 2025. Investigating and scaling up code-switching for multilingual language model pre-training. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11032–11046, Vienna, Austria. Association for Computational Linguistics.

Zhijun Wang, Junxiao Liu, Hao Zhou, Hao-Ran Wei, Baosong Yang, and Shujian Huang. 2026. Efficient multilingual reasoning transfer via progressive code-switching. *arXiv preprint arXiv:2607.00485*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Jian Yang, Shuming Ma, Dongdong Zhang, ShuangZhi Wu, Zhoujun Li, and Ming Zhou. 2020. Alternating language modeling for cross-lingual pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9386–9393.

Haneul Yoo, Jiho Jin, Kyunghyun Cho, and Alice Oh. 2026. Gradual code-switching as inference-time cross-lingual representational alignment for LLMs. In *Third Conference on Language Modeling*.

Haneul Yoo, Cheonbok Park, Sangdoo Yun, Alice Oh, and Hwaran Lee. 2025. Code-switching curriculum learning for multilingual transfer in LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 7816–7836, Vienna, Austria. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Linda Zeng, Steven Y. Feng, and Michael C. Frank. 2026. Bringing up a bilingual BabyLM: Investigating multilingual language acquisition using small-scale models. *arXiv preprint arXiv:2603.29552*.

Jianyu Zheng, Fengfei Fan, and Jianquan Li. 2024. Incorporating lexical and syntactic knowledge for unsupervised cross-lingual transfer. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8986–8997, Torino, Italia. ELRA and ICCL.

Zhihong Zhu, Xuxin Cheng, Zhiqi Huang, Dongsheng Chen, and Yuexian Zou. 2023. Enhancing code-switching for cross-lingual SLU: A unified view of semantic and grammatical coherence. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7849–7856, Singapore. Association for Computational Linguistics.

A Data Generation Details

Sources, budget, units. We download the BabyBabelLM (Jumelet et al., 2026a) datasets from HuggingFace (IDs `BabyLM-community/babylm-{eng,nld,zho}`) at revisions (`b78a9336`, `1aa063f7`, `600a6657`) respectively. We normalize according to the following per-language byte premiums (Arnett et al., 2024): English 1.0, Dutch 1.0516, Simplified Chinese 0.935966. The word unit is the `num-tokens` column of the released datasets. For English and Dutch this column counts whitespace tokens. For Chinese, per the `babylm-zho` dataset card, it is the subword count under the Qwen3-0.6B tokenizer (we verified an exact match on a sample of documents). We approximate the Chinese count as 0.711 units per character, the corpus-level ratio of `num-tokens` to characters measured on `babylm-zho`. We split

each source document into passages of about 300 words at sentence boundaries. Chinese passages in which Han characters make up fewer than 50% of Han-plus-Latin characters are dropped as romanized-pinyin transcripts (the Chinese portion of CHILDES (MacWhinney, 2000)), as are passages under 15 words in any language.

Generation, batching, cost. Here we describe our final production system to generate of the word- and sentence-level CS corpus. It uses `deepseek-v4-flash` via OpenRouter. Each request carries eight passages sharing one (matrix, embedded) language pair as JSON lines and returns one JSON line per input id. The full translations used only by the document translation control (§3) were generated with `claude-haiku-4-5` via the Anthropic Message Batches API, with the prompt of Appendix B. Each switch type covers the same 413,302 passages (51,666 requests); the total API cost was ≈$284.

Quality verification. Every generation must contain both the matrix and the embedded language. We check language presence with (`lingua`): Chinese counts as present at ≥3 Han characters; in Latin-script text a language counts as present when it covers ≥8% of the detected span length. We reject outputs that are identical to the source (no switching), that miss the matrix language (a full translation), or that miss the embedded language, and we reject outputs whose character count falls outside 0.25×–4× that of the source, to catch truncations and runaway generations. Acceptance rates, as a fraction of requested passages, are 0.78–0.89 for word-level switching in every direction and 0.79–0.87 for sentence-level switching into an English or Dutch matrix, but fall to 0.47–0.56 for sentence-level switching into a Chinese matrix, where most rejections are unchanged or fully translated outputs. In total ∼348K word-level and ∼323K sentence-level passages make it past this initial quality filter.

Document reassembly. Accepted passages are stitched back into documents in order. A document whose every passage was rejected is dropped, and one with a rejected passage is reassembled with that passage missing (3.7% of CS documents, 1.8% with a gap in the middle of the passage). The non-CS baseline document is stitched from the source text of the same accepted passages, so both the CS and non-CS versions have identical gaps.

Budget accounting. The word-level, sentence-level, and non-CS thirds occupy disjoint document ids, so that no same document is used for two different thirds of our corpus. Because we found that per-language word heuristics do not work well on mixed text, we check compliance to the 100M adjusted-word budget twice. We use a script-aware word count (Han characters ×0.711 plus whitespace tokens of the Latin remainder), and we also count the total UTF-8 bytes (543 MB ≡ 100M English words). By both metrics, the CS corpus is within budget (≈96.6M words / 485 MB) and near-identical in volume to its non-CS baseline (95.7M words / 482 MB); the 99.6M of Table 1 counts the same documents with the per-language heuristic above. Under our 16,384-entry BPE vocabulary the CS corpus is 138.9M tokens and the non-CS corpus 135.0M, a 2.9% difference; the two code-switched thirds tokenize to 2–12% more tokens than their source documents, most for Chinese-matrix text with translated Latin-script sentences.

Examples from control corpora. The following two examples come from the word salad corpus (§3):

> EN ← NL Transduction is a wisselkoersen term. It can mean:
>
> ZH ← NL 曾以"女排精神"引mevrouw领一代风骚的中国voelen女排，今gravin天再度擎起世界冠军大旗，怎能不让国人惊喜？

The following two come from the document translation corpus:

> EN → NL Burien is a city in King County, Washington, United States. ⇒ Burien is een stad in King County, Washington, Verenigde Staten.
>
> ZH → EN 不然它们就吃撑了。 ⇒ Otherwise they will eat until they are stuffed.

B Generation Model and Prompts

We chose the generation model through a couple different steps. First we scored a 36-passage sample (three per matrix×embedded×switch-type cell) from several frontier LLMs (as of June 2026) with the quality gate of Appendix A, supplemented by manual review from native English–Dutch and English–Chinese speakers. Pass rates were DeepSeek-V4-Flash (DeepSeek-AI, 2026) 0.72, `claude-haiku-4-5` 0.67, Gemini 2.5 Flash-Lite 0.33, and Qwen3-30B-A3B 0.19. The Qwen3

	Word-level CS	Sentence-level CS	Non-CS
English	EN ← NL Rondo is a dorp in de VS staat Arkansas.	EN ↔ ZH Landaff is a district in Wales. 它是英国作家罗尔德·达尔的出生地。	Rondo is a town in the US state of Arkansas.
Dutch	NL ← ZH 'Ga 走, voordat het te laat 太晚 is!' snauwde opa 爷爷.	NL ↔ EN Iemand voor schut zetten. To put someone to shame is to insult someone.	'Ga, voordat het te laat is!' snauwde opa.
Chinese	ZH ← EN 但是我不会 swim 呀,你那 swimming posture 也分解不对呀。	ZH ↔ NL 我知道了。 Dan moeten we morgen zeker ijs eten.	但是我不会游泳呀,你那游泳姿势也分解不对呀。

Figure 5: **Verbatim corpus examples** by matrix language (rows) and switch type (columns). Embedded-language spans are colored by language (English, Dutch, Chinese). The non-CS cell in each row is the true source document of that row's word-level example.

models often drifted into full translation on cross-script pairs, whereas Haiku often failed to produce embedded Chinese words. We therefore used `deepseek-v4-flash` for the full run, which from manual inspection gave good results for each kind of code-switching. Unlike Yoo et al. (2025), we did not need to supply parallel translations to the generator to obtain reasonable CS passages.

Our generation prompts follow verbatim. `{matrix}`/`{embedded}` refer to the language names (English/Dutch/Chinese (Simplified)). Each prompt is followed by the instruction to return exactly one JSON line per input id with the same ids, and then by the input passages as JSON lines of the form `{"id": ..., "text": ...}`.

Word-level CS.

```
Task: rewrite each {matrix} text as
INTRASENTENTIAL code-switching with {embedded}.
Rules:
- Keep {matrix} as the matrix language: it
  carries the grammar, word order, and most
  function words. Most words stay in {matrix}.
- CRITICAL: do NOT translate sentences fully
  into {embedded}. {matrix} must remain
  dominant throughout.
- Embed natural {embedded} content words and
  short phrases INSIDE sentences (nouns, verbs,
  adjectives, technical terms, short NPs).
- Switch only at grammatically permissible
  boundaries. Do not switch inside a word.
- Target roughly 25-45% of content words/
  phrases in {embedded}; every non-trivial
  sentence should contain >=1 {embedded} switch.
- Preserve meaning exactly. Do not add, drop,
  summarise, or invent facts.
- Do NOT translate a whole sentence into
  {embedded}: mix both languages within
  sentences.
- Use the native script for {embedded}.
- Preserve sentence order and any list/markup.
```

Sentence-level CS.

```
Task: rewrite each {matrix} text as
SENTENCE-LEVEL code-switching with {embedded}.
Rules:
- Keep the original sentence ORDER. Translate
  roughly half of the sentences fully into
  {embedded}; leave the rest in {matrix}.
- CRITICAL: do NOT translate the whole passage.
  About half of sentences must remain {matrix}.
  Output with no {matrix} sentences is invalid.
- Each sentence stays internally monolingual:
  all {matrix} or all {embedded}. Do NOT mix
  languages inside a sentence.
- Translate faithfully; preserve meaning. Do
  not add, drop, or invent facts.
- Leave short fragments, list items, headings,
  or markup-like lines unchanged if translating
  would corrupt them.
- Use the native script for {embedded}.
```

Document translation control.

```
Task: translate each {matrix} passage FAITHFULLY
and COMPLETELY into {embedded}.
Rules:
- Translate the ENTIRE passage into {embedded};
  no {matrix} words should remain (proper nouns
  may stay).
- Preserve meaning, sentence order, and
  structure exactly; do not add, drop, or
  summarise.
- Natural, fluent {embedded}; not word-for-word
  glossing.
- Use the native script for {embedded}.
```

C Evaluation Suite

We evaluate with the official BabyLM 2026 multilingual evaluation pipeline (Choshen et al., 2026), which scores each task in each of the three languages where available.

Zero-shot tasks. The zero-shot tasks consist of minimal-pair and multiple-choice tasks scored by comparing candidate log-probabilities. Grammar evaluations include BabyLM's filtered subset of BLiMP (Warstadt et al., 2020) for English, with BLiMP-NL (Suijkerbuijk et al., 2025) and ZhoBLiMP (Liu et al., 2026) as its Dutch and Chinese counterparts, and MultiBLiMP (Jumelet et al., 2026b) in English and Dutch. Common-sense semantic reasoning benchmarks include HellaSwag (Zellers et al., 2019); Winogrande (Sakaguchi et al., 2020); XStoryCloze (Lin et al., 2022b); XCOMPS (He et al., 2025); Global PIQA (Chang et al., 2025); and the Chinese Hanzi Structure and Hanzi Pinyin minimal-pair tasks distributed with

Task	Chance	Shuf	CS/Shuf	Curr	CS/Curr	Word	Sent	Par	Salad
English									
BLiMP	50.0	67.00±0.97	68.27±0.56	66.17±0.90	67.18±0.98	67.95±0.45	68.68±0.62	68.03±0.73	65.88±0.78
MultiBLiMP	50.0	83.75±1.23	85.70±1.55	83.64±0.84	84.74±0.85	83.85±1.27	85.55±0.89	85.50±1.22	82.48±1.66
HellaSwag	25.0	26.67±0.14	26.48±0.14	26.60±0.10	26.56±0.26	26.63±0.14	26.59±0.24	26.59±0.17	26.52±0.16
Winogrande	50.0	50.37±0.66	50.06±1.08	50.30±1.06	50.60±0.75	49.97±0.63	50.01±0.62	50.13±0.80	49.89±0.51
XStoryCloze	50.0	50.60±0.60	50.22±0.71	50.24±0.48	50.24±0.37	50.18±0.57	49.63±0.72	50.74±0.62	50.07±0.76
Global PIQA	25.0	38.49±2.20	37.83±1.47	37.57±1.68	37.53±2.07	38.08±1.47	37.34±2.84	38.13±1.88	38.38±2.01
ARC	25.0	26.09±2.47	24.66±2.56	26.07±1.33	26.20±0.81	25.57±1.57	26.25±1.74	25.76±1.21	25.57±1.59
Belebele	25.0	22.44±2.04	21.59±2.17	21.88±2.45	22.02±1.94	22.87±1.91	22.87±2.23	20.88±1.76	23.65±2.10
BMLAMA	—	23.58±3.87	21.29±4.00	22.79±1.91	22.15±3.57	23.83±2.13	23.02±1.61	21.99±3.72	20.79±3.99
MNLI	33.3	45.07±1.74	46.50±1.39	45.09±0.84	47.02±2.43	46.57±1.99	45.43±1.99	46.23±1.45	45.22±1.19
SIB-200	14.3	72.00±3.79	72.25±2.22	71.38±2.63	73.63±1.77	73.31±2.90	71.31±2.42	71.75±2.28	72.12±2.33
TruthfulQA	—	24.00±3.77	23.77±6.17	25.78±3.55	27.12±3.69	27.01±4.26	28.24±4.42	27.57±5.17	28.57±5.10
XNLI	33.3	43.26±0.37	44.62±1.20	43.71±1.36	44.26±1.18	43.72±1.03	44.33±1.24	44.27±0.83	44.39±0.92
POS	—	92.79±0.29	92.98±0.30	92.75±0.26	92.76±0.13	92.90±0.28	92.79±0.23	92.84±0.17	92.92±0.09
Dutch									
BLiMP-NL	50.0	77.35±0.55	77.36±0.63	76.84±0.71	76.80±0.62	77.49±0.44	77.57±0.46	77.27±0.52	76.15±0.79
MultiBLiMP	50.0	90.06±0.56	90.41±0.30	89.40±0.31	89.60±0.62	90.19±0.56	90.28±0.67	90.27±0.60	89.16±0.59
HellaSwag	25.0	26.40±0.20	26.43±0.13	26.47±0.12	26.32±0.20	26.37±0.15	26.38±0.18	26.53±0.13	26.25±0.22
Winogrande	50.0	49.77±0.81	50.59±1.21	49.33±0.61	49.79±0.52	49.95±0.62	49.81±0.45	49.38±0.91	49.99±1.37
XCOMPS	50.0	52.90±0.58	52.34±0.43	52.64±0.69	52.40±0.31	52.35±0.52	52.42±0.56	52.53±0.85	52.28±0.43
XStoryCloze	50.0	48.06±0.49	49.02±0.54	48.57±0.37	48.46±0.58	48.39±0.68	48.82±0.40	48.36±0.48	48.12±0.28
Global PIQA	25.0	39.25±1.74	37.61±1.81	39.64±1.90	39.22±1.00	36.63±1.69	38.35±2.49	39.34±0.87	38.84±2.27
ARC	25.0	26.67±1.28	26.35±1.40	26.25±2.18	26.46±1.81	26.54±2.25	26.07±2.61	24.90±1.72	26.30±1.71
Belebele	25.0	24.01±1.60	23.22±1.62	22.73±1.58	23.86±1.39	21.24±1.49	22.59±2.12	24.22±1.80	23.22±2.05
BMLAMA	—	22.88±2.92	22.12±2.24	23.12±2.71	22.87±1.87	21.78±4.19	22.55±2.51	22.08±3.56	19.42±4.61
INCLUDE	25.0	31.58±3.13	31.47±1.49	30.47±4.95	31.47±2.65	30.13±5.00	32.81±2.51	32.81±5.18	30.58±4.62
MNLI	33.3	46.66±1.26	46.59±1.71	47.22±0.87	46.67±1.61	47.71±1.15	46.99±1.71	47.22±1.22	46.23±1.52
SIB-200	14.3	67.69±3.24	69.50±2.04	69.06±2.26	70.62±2.50	68.81±1.56	68.44±2.15	69.50±1.77	68.00±2.59
TruthfulQA	—	22.77±3.94	24.44±3.13	23.55±2.61	23.55±3.13	23.77±3.88	24.11±3.31	24.78±2.69	23.10±2.76
POS	—	94.45±0.30	94.52±0.18	94.28±0.20	94.24±0.33	94.48±0.16	94.33±0.15	94.54±0.15	94.41±0.21
Chinese									
ZhoBLiMP	50.0	75.95±0.91	76.88±0.81	75.52±1.64	74.84±0.39	74.65±1.21	76.28±0.52	75.22±0.94	72.48±0.97
HellaSwag	25.0	26.91±0.12	26.41±0.17	26.86±0.14	26.63±0.21	26.83±0.09	26.50±0.11	26.64±0.08	26.93±0.05
Winogrande	50.0	48.75±0.84	49.45±0.94	49.40±0.63	49.34±0.86	49.61±0.78	49.88±0.61	49.48±0.27	49.20±0.95
XCOMPS	50.0	53.71±0.56	53.99±0.57	53.55±0.24	53.73±0.32	53.87±0.43	53.81±0.56	53.62±0.42	51.17±0.48
XStoryCloze	50.0	47.93±0.54	48.60±0.54	47.42±0.53	48.42±0.42	49.03±0.52	49.05±0.65	48.90±0.44	47.92±0.50
Global PIQA	25.0	32.77±1.65	32.13±2.15	33.35±1.45	33.22±1.28	32.66±1.64	33.05±2.90	31.12±1.58	31.95±1.80
ARC	25.0	28.02±2.79	26.43±1.22	26.80±1.54	26.77±2.11	26.56±1.59	26.82±2.32	25.49±1.02	26.17±1.30
Belebele	25.0	20.95±1.96	21.16±1.54	22.23±1.34	22.73±2.31	22.94±1.94	21.31±1.05	22.80±2.11	23.08±3.30
BMLAMA	—	18.34±2.56	16.20±4.07	17.42±2.66	15.90±3.86	17.28±2.59	18.49±2.56	15.98±3.67	15.21±2.92
INCLUDE	25.0	26.45±3.44	26.45±3.99	25.11±3.95	26.79±2.70	27.12±3.88	25.56±2.08	24.33±3.96	25.67±4.32
MNLI	33.3	44.64±1.67	47.43±2.33	45.45±2.12	47.42±2.58	45.21±1.83	45.62±1.47	46.38±3.02	46.00±1.29
SIB-200	14.3	75.88±2.46	78.25±2.67	76.56±1.80	78.56±1.40	78.19±2.58	77.81±1.39	77.00±2.24	77.13±1.94
TruthfulQA	—	22.99±2.85	21.99±4.32	23.33±2.31	22.10±3.43	22.32±1.58	23.21±2.29	24.11±2.53	23.88±2.33
XNLI	33.3	42.66±1.01	42.86±0.47	42.47±0.43	43.67±1.54	43.73±1.09	42.83±0.71	41.83±1.50	40.47±2.77
POS	—	89.67±0.22	90.56±0.31	89.81±0.22	90.05±0.29	90.26±0.27	89.87±0.27	90.11±0.38	90.02±0.19
Leaderboard		46.44±0.25	46.55±0.47	46.36±0.37	46.72±0.25	46.59±0.24	46.70±0.29	46.56±0.38	46.08±0.34

Table 3: Accuracy on the BabyLM evaluation suite per task and per language. We report (%) mean ± SD across eight seeds per condition. Chance performance is random-guess accuracy ($1/\#\text{options}$) where applicable. Word/Sent/Par/Salad are the shuffled-ordering controls (§3). *Leaderboard* is the official multilingual-average score as in Table 2.

Condition	English	Dutch	Chinese
Shuf	1.60±0.05	1.56±0.04	1.78±0.03
CS/Shuf	1.45±0.01	1.45±0.01	1.75±0.01
Curr	1.56±0.02	1.53±0.01	1.78±0.01
CS/Curr	1.56±0.03	1.56±0.03	1.80±0.01
Word-level CS only	1.52±0.04	1.53±0.03	1.78±0.02
Sentence-level CS only	1.42±0.02	1.42±0.02	1.73±0.02
Document translation	1.63±0.06	1.61±0.05	1.78±0.03
Word salad	1.65±0.03	1.61±0.03	1.79±0.01

Table 4: Bits per UTF-8 byte on 1,200 held-out BabyBabelLM documents per language (lower is better). The first 512 tokens of each document are scored.

the pipeline. The latter two are hidden tasks: the leaderboard computes their scores server-side and excludes them from every average, so they do not enter the leaderboard score or Table 3.

Fine-tuning tasks. ARC (Clark et al., 2018); Belebele (Bandarkar et al., 2024); BMLAMA (Qi et al., 2023); MNLI (Williams et al., 2018); SIB-200 (Adelani et al., 2024); TruthfulQA (Lin et al., 2022a); XNLI (Conneau et al., 2018); INCLUDE (Romanou et al., 2025); and cross-lingual POS tagging over Universal Dependencies treebanks (de Marneffe et al., 2021). Each classification task is fine-tuned and evaluated within one language (up to 10 epochs with early stopping at patience 3, learning rate 5×10^{-5}, batch size 64, one fine-tuning seed); only POS tagging is trained jointly on all three languages.

Harness details. We run the harness with the organizers' fix for the beginning-of-sequence (BOS) token: the harness recovers the continuation tokens as the suffix of the tokenized prompt+continuation, which breaks for tokenizers that, like the baseline's and ours, wrap every string in `<s>…</s>`; the fix tokenizes without special tokens and prepends a single `<s>`.

D Details on Bitext Retrieval

Protocol. We embed the FLORES+ `dev` set (NLLB Team et al., 2024; Goyal et al., 2022) (`eng_Latn/nld_Latn/cmn_Hans`), truncating each sentence to its first 128 tokens and excluding padding from the mean over positions. Let $S = (s_{ij})$ be the $N \times N$ matrix of cosine similarities such that s_{ij} is the cosine similarity between the representations of the i-th source-language sentence and the j-th target-language sentence. Plain nearest-neighbor retrieval over S suffers from hubness, where a few target sentences are the nearest neighbor of many unrelated queries

Pair	P@1	CKA	cos-gap	centroid↓
En–Nl	91/84	0.69/0.67	0.11/0.09	0.36/0.33
En–Zh	68/7	0.63/0.56	0.10/0.03	0.48/0.53
Nl–Zh	58/6	0.60/0.54	0.10/0.03	0.47/0.53
MEXA Dutch		0.87/0.77		
MEXA Chinese		0.61/0.07		

Table 5: Cross-lingual alignment measures beyond retrieval (curriculum ordering). Each cell shows the mean CS/non-CS values across seeds at the layer of peak CSLS retrieval in the CS model; We measure linear CKA (Kornblith et al., 2019), the cosine gap (mean cosine over the N true translation pairs minus that over a fixed random derangement of the pairs), and the per-language centroid distance (lower is better; Libovický et al., 2020). The last two rows are per-language MEXA (Kargaran et al., 2025) (max over layers vs. the English pivot).

(Radovanović et al., 2010), so we rank with cross-domain similarity local scaling (k=10; Lample et al., 2018), forming the matrix $C = (c_{ij})$ where

$$c_{ij} := 2s_{ij} - r_B(i) - r_A(j),$$

where $r_B(i)$ is the mean cosine between source sentence i and its k nearest targets and $r_A(j)$ the mean cosine between target sentence j and its k nearest sources. Retrieval is correct for sentence i when its translation ranks first under C, and we retrieve in both directions and average:

$$\text{P@1} := \tfrac{1}{2}\big(\text{acc}_{A\to B} + \text{acc}_{B\to A}\big),$$

where

$$\text{acc}_{A\to B} := \frac{1}{N}\sum_i \mathbf{1}[\arg\max_j c_{ij} = i]$$

is the fraction of source sentences whose top-ranked target is their own translation, and $\text{acc}_{B\to A}$ is the same in reverse (C is symmetric in the two directions, since $C^\top$ is the CSLS matrix of $S^\top$). Chance is $1/N \approx 0.1\%$.

E Details on Word-Level Alignment

Constructing candidate sets for retrieval. For each language direction, we first group words by part of speech and subword-token count. We bisect each group into embedded and never-embedded words and match them one-to-one by the Hungarian algorithm on $\log_{10}$ monolingual frequency. Translation labels come from the MUSE dictionaries (Lample et al., 2018) in addition to the pairs of words substituted during corpus generation.

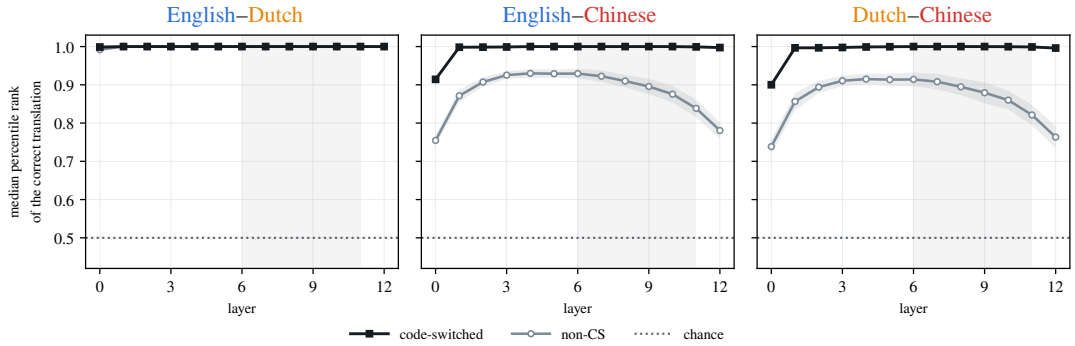


Figure 6: In addition to sentence-level bitext retrieval precision@1 (Figure 2), here we show the per-layer median percentile rank of the correct translation. Averaged over layers 6–11, the non-CS models place the correct translation at median rank 105 and 121 out of 997 for English-Chinese and Dutch-Chinese respectively, far above chance at 499. This suggests that code-switching does not create alignment "from scratch", but rather "sharpens" a general degree of alignment found even in the non-CS models.

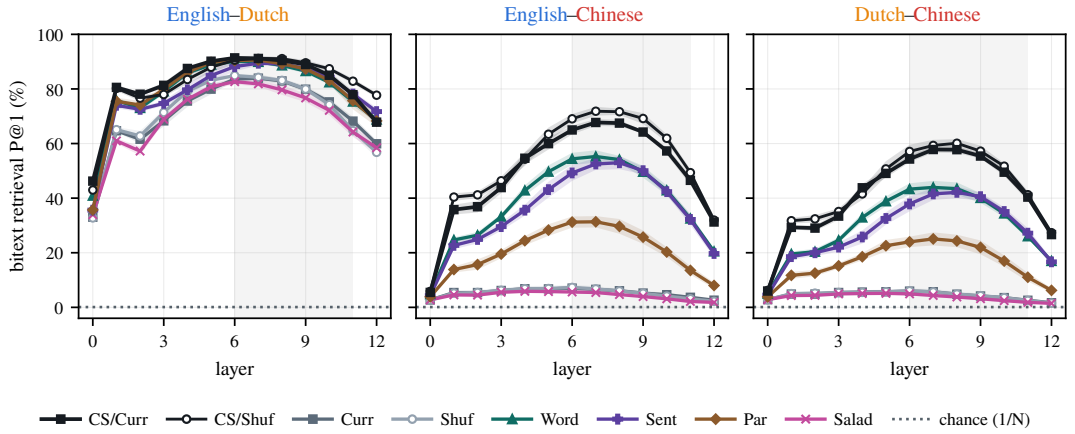


Figure 7: Per-layer bitext retrieval P@1 for all eight experimental conditions (§4), overlaid in each language-pair panel. We use neutral colors for the four main conditions, each pairing of corpus (CS/non-CS) and ordering (shuffled/curriculum), and color for the four shuffled-ordering controls. Lines are 8-seed means, bands ±1 SD, dotted line chance. Shading marks layers 6–11, the range averaged over in Figure 3.

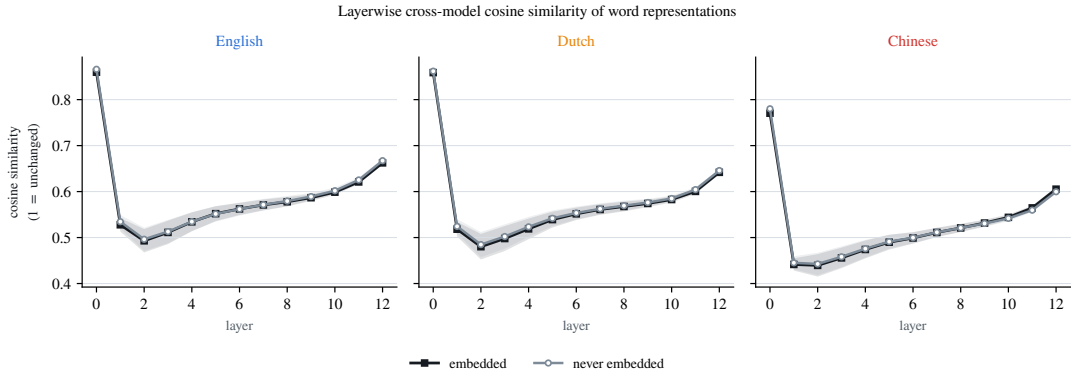


Figure 8: Layerwise cosine similarity between the CS and non-CS models' word representations (§7), shown separately for the embedded words and their never-embedded controls (8-seed mean ±1 SD).

	Bitext retrieval P@1 (%)			
Condition	En–Nl	En–Zh	Nl–Zh	Leaderboard
Shuf	78.5±0.7	5.4±0.8	4.5±0.8	46.44
CS/Shuf	88.8±0.5	65.5±1.1	54.5±1.0	46.55
Curr	79.0±0.8	5.5±0.6	4.5±0.5	46.36
CS/Curr	87.5±0.7	61.4±0.8	52.5±0.6	46.72
Word	85.5±0.8	48.1±1.5	38.5±1.4	46.59
Sent	86.1±0.5	46.6±1.4	37.4±1.5	46.70
Par	86.0±0.7	25.3±2.0	20.6±2.0	46.56
Salad	76.2±1.2	4.2±0.4	3.4±0.5	46.08

Table 6: Cross-lingual alignment of all eight conditions at their final checkpoints; alignment columns are 8-seed mean ± SD. Left: FLORES+ bitext retrieval P@1 (CSLS, k=10) averaged over layers 6–11, per language pair. Right: the leaderboard score of Table 2. Column abbreviations as in Table 3.

How code-switching shifts word representations across layers. For each of the 2,204 source words in the 1,102 matched pairs of §7, at each layer, we take the cosine between the CS and non-CS models' representations (Figure 8). The high cosine similarity between CS and non-CS models at layer zero indicates that code-switching does not affect the word embeddings much. However, early layers seem more affected representationally by the presence of code-switched text in training, as is seen by the low cosine similarity. The representations of words at later layers become again more similar between the CS and non-CS models.

XpertGPT: Multi-Scale Sparse Expert Routing for Data-Constrained Language Modeling

Soham Jain*, **Harsh Singh***, **Divija Dewan***, **Atul Dev**
Vision and Language Group, Indian Institute of Technology Roorkee
Roorkee, Uttarakhand, India – 247667
{soham_rj@mfs, harsh_s@mfs, divija_d@ee, atul_d@ma}.iitr.ac.in

Abstract

We introduce XpertGPT, a sparse decoder-only Transformer for data-constrained language modeling that routes tokens to parallel expert pathways operating at multiple contextual scales. Under the BabyLM strict-small budget of roughly ten million words, conventional dense Transformers spend identical computation on every token and impose a single fixed receptive field. Each XpertGPT layer pairs a dense global-attention stream with four sparse sliding-window expert branches, whose fixed attention spans range from 64 down to 4 tokens; tokens are dispatched to these branches by Expert-Choice routing, which balances expert load by construction without any auxiliary loss, and rotary position embeddings and gated feed-forward networks are applied throughout. Trained on the BabyLM 2026 Strict-Small corpus, our 52.6M-parameter model reaches an overall average of 38.42, surpassing the official GPT-2 (98.4M) baseline (37.38) on average across grammatical acceptability, world-knowledge, and language-understanding benchmarks.

1 Introduction

Human children acquire rich, adult-level linguistic competence from a remarkably small volume of input: by early adolescence, a child encounters no more than roughly 100 million words–orders of magnitude fewer tokens than the web-scale trillions used in contemporary language model pretraining (Warstadt et al., 2023). The BabyLM Challenge formalizes this core developmental question by restricting pretraining to human-scale data budgets, directly testing how efficiently an architecture can extract syntax, semantics, and world knowledge from limited evidence (Warstadt et al., 2023; Choshen et al., 2026). Under such data-constrained regimes–where the budget is restricted to roughly ten million words–conventional dense Transformers waste capacity by activating every parameter uniformly across every token and enforcing a single fixed receptive field. To address this mismatch, we introduce *XpertGPT*, a sparse transformer architecture that improves parameter utilization by routing tokens to specialized expert pathways operating at multiple contextual scales, achieving stronger overall performance than the official BabyLM 2026 GPT-2 baseline under identical training constraints.[1]

Language naturally exhibits hierarchical structure across multiple contextual scales. Local syntactic dependencies, such as subject-verb agreement, typically require only nearby context, whereas discourse coherence, entity tracking, and long-range semantic dependencies span much larger contexts. Recent BabyLM studies further demonstrate that no single sequence length is universally optimal across downstream evaluation tasks (Salhan et al., 2025). These observations suggest that a fixed receptive field is fundamentally mismatched to the multi-scale nature of language, particularly in low-resource pretraining where every parameter must contribute effectively. Existing sparse architectures address only parts of this challenge. Standard Mixture-of-Experts (MoE) models (Shazeer et al., 2017; Fedus et al., 2022) sparsify feed-forward computation while retaining dense attention. Mixture of Layers (MoL) sparsifies entire transformer blocks but does not explicitly specialize experts for different contextual scales (Ternovtsii and Bilak, 2026). Modular routing approaches such as Modular Expert Pathways (MoEP) introduce sparse execution but rely on token-choice routing, which can result in imbalanced expert utilization (Tapaninaho, 2025). Our architecture combines these complementary

* Equal contribution.

[1] Our code is publicly available at https://github.com/sohamjain5035/XpertGPT.

Proceedings of the Second BabyLM Workshop, pages 767–778
October 29, 2026 ©2026 Association for Computational Linguistics

ideas into a unified framework. A global attention stream preserves full-sequence contextual information, while multiple sparse expert branches operate over different sliding-window attention spans to capture linguistic phenomena at different granularities. Expert-Choice routing enables experts to select the tokens they process, encouraging balanced utilization without auxiliary load-balancing losses (Zhou et al., 2022). Within each expert, blocks employ SwiGLU feed-forward networks to increase representational capacity while maintaining computational efficiency (Shazeer, 2020). Together, these components provide an architecture that balances global contextual reasoning with specialized local processing.

We evaluate our model on the BabyLM 2026 Strict-Small benchmark across grammatical acceptability, world knowledge, and natural language understanding tasks. Under identical training conditions, our model achieves an average score of 38.42 compared with 37.38 for the BabyLM 2026 GPT-2 baseline, demonstrating consistent improvements across multiple evaluation families. Because both models are trained on the same corpus, these gains can be attributed to architectural improvements rather than additional data.

Our contributions are summarized as follows:

- We propose XpertGPT, a sparse transformer architecture that combines full-sequence global attention with parallel expert pathways specialized for different contextual scales.
- We integrate Expert-Choice routing within a Mixture-of-Layers framework, enabling balanced expert utilization and contextual-scale specialization without auxiliary load-balancing objectives.
- We demonstrate consistent improvements over the BabyLM 2026 GPT-2 baseline across multiple evaluation families under identical training data constraints.
- We provide empirical evidence that multi-scale expert specialization improves parameter utilization under strict low-resource language model pretraining.
- **Open-Source Release:** We publicly release our implementation and experimental framework to facilitate reproducible research in data-constrained model architectures.

2 Background and Related Work

2.1 Data-Constrained Pretraining and BabyLM

The BabyLM Challenge asks whether language models can learn from developmentally plausible data budgets: a child encounters no more than roughly 100M words by early adolescence, orders of magnitude less than the token counts of contemporary pretraining (Warstadt et al., 2023). The 2026 edition retains the two standard text tracks, Strict (100M words) and Strict-Small (10M words), while additionally capping training at ten epochs and requiring intermediate checkpoints so that learning trajectories are evaluated alongside final scores (Choshen et al., 2026). Earlier editions showed that architectural choices, rather than parameter count alone, dominate outcomes at this scale. Mehta et al. (2025) further demonstrate that combining orthogonal efficiency mechanisms, including sparse experts, compressed attention, and rotary embeddings, provides larger gains than any individual technique. XpertGPT continues this line by adopting a purely causal decoder assembled from sparse architectural components specifically designed for the Strict-Small setting.

2.2 Sparse Mixture-of-Experts and Expert-Choice Routing

Mixture-of-Experts (MoE) replaces a single feed-forward network with multiple expert networks and a learned router that activates only a subset of experts for each token, thereby decoupling parameter count from per-token computation (Shazeer et al., 2017; Fedus et al., 2022). Conventional token-choice routing allows every token to independently select its preferred experts, but this frequently produces imbalanced expert utilization and therefore requires auxiliary load-balancing losses (Zhou et al., 2022). Expert-Choice routing instead reverses the assignment: each expert independently selects its top-k tokens according to router affinity. This guarantees balanced expert utilization by construction, removes the need for auxiliary balancing objectives, and allows more computational resources to be allocated to difficult tokens (Zhou et al., 2022). XpertGPT adopts this routing mechanism throughout all sparse expert layers.

2.3 Parallel Block Architectures

Recent work has extended sparse routing beyond feed-forward layers to complete Transformer

blocks. Mixture of Layers (MoL) replaces full-width Transformer blocks with multiple parallel blocks operating at reduced dimensionality behind learned projection layers, while retaining a dense global attention pathway to preserve full-sequence context (Ternovtsii and Bilak, 2026). Within the BabyLM challenge, MoEP similarly routes tokens through reduced-dimension Transformer blocks placed between shared projection layers, demonstrating improved sample efficiency over dense GPT-2 baselines under the Strict-Small setting (Tapaninaho, 2025). Both approaches rely on token-choice routing together with explicit load-balancing losses. XpertGPT follows the block-level sparse design but replaces token-choice routing with Expert-Choice routing and incorporates multi-scale sliding-window experts.

2.4 Local Attention, Positional Encoding, and Activations

Sliding-window attention limits each query to nearby tokens, substantially reducing computational cost while encouraging locality, making it particularly suitable for short-range linguistic dependencies (Child et al., 2019; Beltagy et al., 2020; Jiang et al., 2023). To encode positional information without adding trainable parameters, Rotary Position Embeddings (RoPE) rotate query and key vectors, enabling attention scores to depend on relative token offsets (Su et al., 2024). Additionally, gated linear unit variants such as SwiGLU have consistently improved optimization and downstream performance over standard feed-forward networks while maintaining comparable parameter efficiency (Shazeer, 2020). In our framework, we pair RoPE and SwiGLU across both the dense global pathway and the sparse expert branches to maximize representational capacity and relative positional stability under tight data constraints.

3 Method

3.1 Overall Architecture

Our model stacks L=6 identical XpertGPT Blocks between a shared token embedding layer and a final layer-normalized language-modeling head. Input tokens are embedded into d_{model}=256 dimensions; the embedding matrix is tied with the output projection to reduce parameter count. Each XpertGPT Block processes the token sequence through two sequential stages: a **global attention block** that establishes full-sequence context, followed by a **sparse multi-scale expert stage** that selectively refines token representations through parallel sliding-window pathways. Figure 1 illustrates the full block architecture.

3.2 Global Attention Block

The first stage of each XpertGPT Block operates on the full sequence at dimension d_{model}. It comprises a pre-LayerNorm multi-head causal self-attention layer with h_g=4 heads (head dimension d_h=64) followed by a gated feed-forward network. Self-attention uses Rotary Position Embeddings (RoPE) (Su et al., 2024) without any window constraint, so every token can attend to all preceding positions. Given query and key vectors $\mathbf{q}_t, \mathbf{k}_s \in R^{d_h}$ at positions t and s, RoPE encodes relative position by applying a rotation matrix R_θ:

$$\text{Attn}(t,s) = \frac{(R_{\theta,t}\mathbf{q}_t)^\top (R_{\theta,s}\mathbf{k}_s)}{\sqrt{d_h}} \quad (1)$$

where $R_{\theta,p}$ rotates consecutive dimension pairs by $p\theta_i$ with frequencies $\theta_i = 10000^{-2i/d_h}$. The output vector is then formed via standard softmax and value-vector aggregation over $\text{Attn}(t,s)$, and the resulting attention output is projected through a linear layer and added to the residual stream.

All feed-forward sublayers use the SwiGLU activation (Shazeer, 2020), defined as

$$\text{SwiGLU}(\mathbf{x}) = (\text{SiLU}(W_1\mathbf{x}) \odot W_2\mathbf{x})\, W_3 \quad (2)$$

with hidden dimension $\lfloor 8d/3 \rceil_8$, maintaining a parameter count comparable to a conventional $4d$ FFN while improving optimization.

3.3 Sparse Multi-Scale Expert Stage

After the global block produces a contextually enriched representation $\mathbf{X} \in R^{B\times T\times d_{\text{model}}}$, the expert stage routes tokens through $E = 4$ parallel expert branches.

Routing. A lightweight router computes per-token expert affinities. The router first normalizes $\mathbf{X}$ using LayerNorm and projects each token through a bias-free linear layer $W_R \in R^{d_{\text{model}}\times E}$,

$$\mathbf{r}_t = \text{softmax}\left(W_R\, \text{LN}(\mathbf{x}_t)\right) \in R^E \quad (3)$$

where $r_{t,i}$ denotes the affinity between token t and expert i.

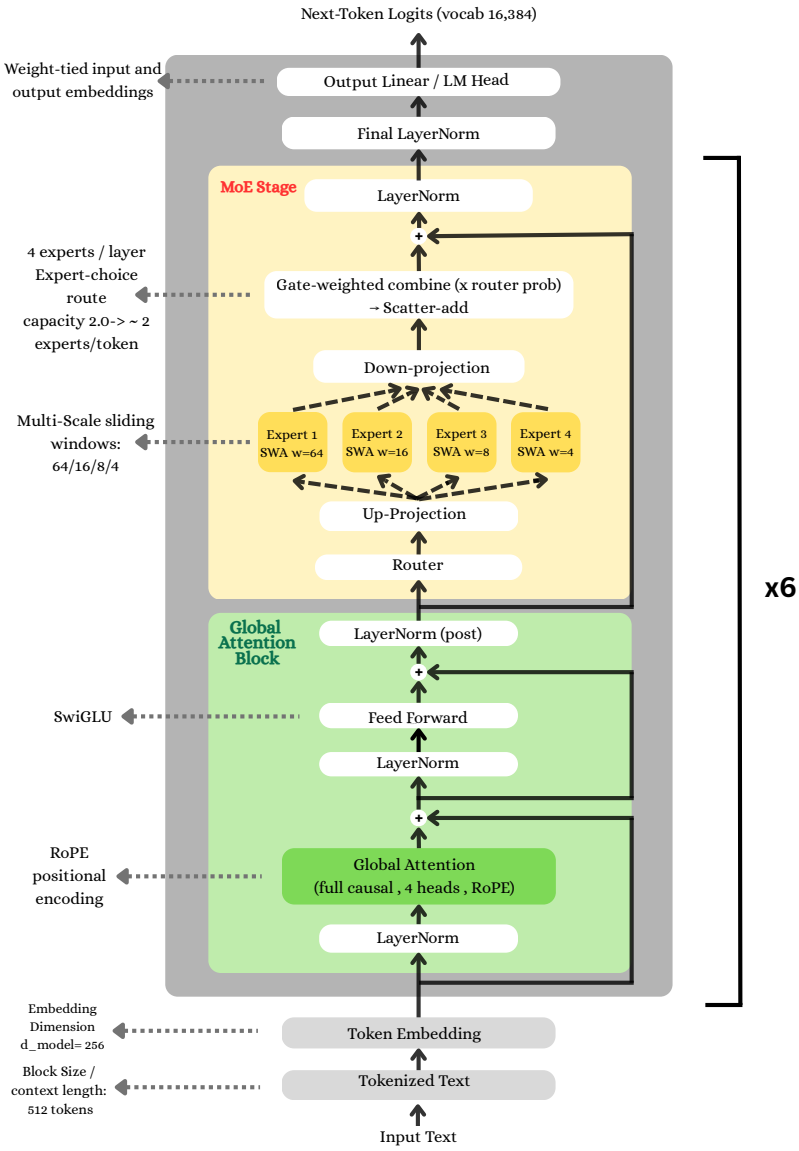


Figure 1: The full XpertGPT layer architecture, illustrating the sequential integration of the Global Attention Block and the Sparse Multi-Scale Expert Stage.

Expert-Choice Selection. We adopt Expert-Choice routing (Zhou et al., 2022), where each expert independently selects its top-k most relevant tokens per sequence rather than each token selecting an expert:

$$\mathcal{S}_i = \text{Top-}k\left(\{r_{t,i}\}_{t=1}^{T}, k\right), \qquad k = \frac{cT}{E} \tag{4}$$

where T is sequence length, $c = 2.0$ is the capacity factor, and E is the expert count. The router applies selection independently across each sequence in the batch to prevent cross-sequence leakage. With $E = 4$ experts, each expert processes $k = T/2$ tokens per sequence, forming $2T$ token–expert assignments per sequence per layer. Because every expert selects exactly k tokens, balanced expert utilization is guaranteed by construction without auxiliary load-balancing objectives. Because top-k selection operates over the full sequence, a token's expert assignment can depend on other tokens later in the same sequence. Crucially, this affects only which expert branch a token is routed to–not the context available during representation learning, as all attention mechanisms (global and expert) remain strictly causally masked. Consequently, token log-likelihoods are computed without future-token leakage, maintaining full validity under the teacher-forced evaluation protocol used across all BabyLM benchmarks.

Dimension Projection. Before expert processing, token representations are projected from d_{model}

to $d_{\text{expert}} = 384$ using a shared bias-free projection

$$W_{\uparrow} \in R^{d_{\text{model}} \times d_{\text{expert}}}.$$

MSIT Expert Branches. Each expert consists of a complete Multi-Scale Information Transmission (MSIT) branch comprising pre-LayerNorm sliding-window causal self-attention with RoPE, followed by a pre-LayerNorm SwiGLU feed-forward network and a final LayerNorm.

All experts operate at $d_{\text{expert}} = 384$ with $h_e = 6$ attention heads but differ in their attention window sizes,

$$[w_1, w_2, w_3, w_4] = [64, 16, 8, 4]. \quad (5)$$

Each expert processes only its selected tokens,

$$\mathbf{y}_{t,i} = \text{MSITBranch}_i\left(W_{\uparrow}\mathbf{x}_t; w_i\right) \in R^{d_{\text{expert}}}. \quad (6)$$

The outputs are projected back to the model dimension using a shared projection $W_{\downarrow} \in R^{d_{\text{expert}} \times d_{\text{model}}}$ and combined according to their routing weights,

$$\mathbf{o}_t = \sum_{i:t \in \mathcal{S}_i} r_{t,i} W_{\downarrow} \mathbf{y}_{t,i}. \quad (7)$$

The final block output is obtained by adding the expert contribution to the residual stream followed by LayerNorm,

$$\mathbf{h}_t = \text{LN}(\mathbf{x}_t + \mathbf{o}_t). \quad (8)$$

3.4 Training Objective

The model is trained using the standard autoregressive cross-entropy objective,

$$\mathcal{L} = -\frac{1}{T}\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \quad (9)$$

As established in §3.3, no auxiliary load-balancing loss is required.

3.5 Model Scale

The complete model contains approximately 48.4M non-embedding parameters and 52.6M total parameters. The tied embedding contributes approximately $16{,}384 \times 256 \approx 4.2$M parameters. Within each XpertGPT Block, the dense stream operates at $d_{\text{model}} = 256$, while all expert branches operate at $d_{\text{expert}} = 384$, providing higher expert capacity while maintaining bounded active parameters through Expert-Choice routing.

4 Experimental Setup

4.1 Dataset

We train on the **BabyLM 2026 Strict-Small** corpus (Warstadt et al., 2023; Choshen et al., 2026), a curated collection of approximately 10 million words of child-directed and child-accessible English text. The corpus aggregates transcribed speech (CHILDES), children's literature (Children's Book Test, Gutenberg), simple English Wikipedia, and subtitles (OpenSubtitles), reflecting the linguistic input available to a young language learner. Under the strict track rules, no additional pretraining data, data augmentation, or synthetic data generation is permitted.

We reserve 5% of the training split as a held-out validation set for monitoring training loss. The main model and all architectural ablations (reported in §5.2 and Appendix B) are trained on this identical data partition with a fixed random seed (42). The seed-sensitivity analysis in Appendix C additionally trains the main configuration under three further seeds (0, 3407, 1234) to quantify variance due to initialization; the data partition itself remains fixed across all seeds.

4.2 Tokenization

We train a custom Byte-Pair Encoding (BPE) tokenizer on the training partition using the HuggingFace `tokenizers` library. The vocabulary contains $|\mathcal{V}| = 16{,}384$ sub-word units, including five reserved special tokens (`[PAD]`, `[UNK]`, `[CLS]`, `[SEP]`, `[MASK]`). We use whitespace-based pre-tokenization. The same tokenizer is shared across all model variants.

4.3 Model Configuration

The model comprises L=6 XpertGPT Blocks with model dimension d_{model}=256, expert dimension $d_{\text{expert}} = 384$, E=4 experts per layer, and capacity factor c=2.0, totaling approximately 52.6M parameters (48.4M non-embedding). The complete architectural and training hyperparameter specification is provided in Appendix A (Table 5).

4.4 Training Protocol

We train on a single workstation equipped with an NVIDIA RTX 6000 Ada Generation GPU (48 GB VRAM, 91.1 TFLOPS FP32 compute) and 16 CPU cores. We optimize using AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\varepsilon = 10^{-8}$, and weight decay $\lambda = 0.1$. The learning rate follows a cosine schedule with a

peak of 3×10^{-4}, decaying to 5% of peak, preceded by 800 steps of linear warm-up. Gradient norms are clipped at 1.0. We train for 10 epochs with batch size 16 and no gradient accumulation, using mixed-precision training (bfloat16).

Weight decay is applied only to parameters with ≥ 2 dimensions (i.e., weight matrices); biases and LayerNorm parameters are excluded.

4.5 Evaluation

We evaluate using the official **BabyLM 2026 evaluation pipeline** (BabyLM Organizers, 2026), which comprises: (i) six **zero-shot NLP benchmarks**–BLiMP (Warstadt et al., 2020), BLiMP Supplement, EWoK (Ivanova et al., 2025), Entity Tracking, COMPS, and GlobalPIQA—scored via causal log-likelihood; and (ii) seven **fine-tuned SuperGLUE tasks** (BoolQ, MultiRC, RTE, WSC, MRPC, QQP, MNLI) using the pipeline's default hyperparameters. Following the BabyLM leaderboard protocol, we report NLP Average and Overall Average in Table 2 for the main model and baseline; these are computed via the official leaderboard's task-weighting scheme rather than a simple unweighted mean of the six zero-shot benchmarks shown in Table 1.

4.6 Baseline

We compare against the official **BabyLM 2026 GPT-2 baseline** (BabyLM Community, 2026a) (98.4M parameters), a standard dense decoder-only transformer trained on the same Strict-Small corpus under identical data constraints. All reported baseline scores are taken directly from the official BabyLM leaderboard (BabyLM Community, 2026b).

5 Results

We evaluate XpertGPT against the official BabyLM 2026 GPT-2 Strict-Small baseline (§5.1) and conduct a controlled ablation study isolating global dense context, Expert-Choice routing, and multi-scale windows (§5.2). We track layer-wise distribution balance via router utilization (§5.3) and evaluate computational efficiency (§5.4).

5.1 Overall Performance

XpertGPT outperforms the baseline on both Overall Average and zero-shot NLP Average, though performance varies across linguistic task families (Table 1 and Table 2). While the dense baseline remains marginally stronger on standard BLiMP, XpertGPT demonstrates clear advantages on BLiMP Supplement and EWoK, showing that multi-scale sparse mixtures better adapt to low-frequency tail constructions and core world-knowledge interactions than fixed-width contexts.

The most pronounced gain occurs on Entity Tracking, confirming that maintaining a dense global-attention stream preserves long-range historical context while Expert-Choice layers allocate specialized parameters for rapid referential transitions. On long-passage text integration challenges (COMPS, GlobalPIQA) and multi-choice SuperGLUE tasks (BoolQ, MultiRC), XpertGPT underperforms the baseline, likely because capacity-constrained pathways drop non-salient boundary tokens needed for macro-paragraph synthesis. Conversely, XpertGPT maintains an edge on shorter sentence-classification contexts (MRPC, QQP, RTE, WSC).

Benchmark	GPT-2 Baseline	XpertGPT (Seed 42)	XpertGPT (Mean $\pm\sigma$)
BLiMP	**65.23**	64.66	63.80 ± 0.76
BLiMP Supp	57.25	**61.00**	59.51 ± 1.14
EWoK	50.63	**51.34**	50.91 ± 1.27
Entity Tracking	19.10	**21.08**	20.02 ± 0.76
COMPS	**51.81**	49.62	50.23 ± 0.50
GlobalPIQA	**35.09**	32.65	33.03 ± 1.46

Table 1: Zero-shot task breakdown comparing the dense GPT-2 baseline against XpertGPT across Seed 42 and multi-seed robustness runs ($n = 4$). Bold indicates superior performance.

Metric	XpertGPT (52.6M)	GPT-2 Baseline (98.4M)
Overall Averages		
Overall Average	**38.42**	37.38
NLP Average	**49.04**	48.99
Fine-tuned Tasks		
BoolQ	64.59	**67.71**
MNLI	48.55	**49.84**
MRPC	**82.74**	81.37
MultiRC	57.55	**65.76**
QQP	**62.30**	61.67
RTE	**57.55**	56.83
WSC	**67.31**	63.46

Table 2: Fine-tuned SuperGLUE classification benchmarks and overall structural performance metrics under the Strict-Small footprint.

5.2 Ablation Study

Table 3 isolates individual architectural mechanisms, highlighting key trade-offs between global

representation stability and local specialization. Removing the sparse experts (Ablation 1) yields strong local syntactic performance, while removing the dense global stream entirely (Ablation 2) causes widespread performance collapse across BLiMP and Entity Tracking, indicating that sliding windows cannot substitute for global attention tracking. Replacing learned projections with stochastic token dispatching (Ablation 3) degrades both BLiMP and Entity Tracking, confirming the necessity of learned semantic routing. Activating all expert weights concurrently without gating (Ablation 4) maximizes short-range acceptability on BLiMP but degrades macro-reasoning on GlobalPIQA and Entity Tracking, showing that un-gated dense updates overfit local surfaces at the expense of sequence-level tracking. Notably, removing gating alone has a smaller net effect than removing either the global stream (Ablation 2) or the learned routing signal (Ablation 3), a trend we return to in §6. The full Expert-Choice model (Ablation 5) achieves optimal multi-task stability, leading on Entity Tracking and BLiMP Supplement while maintaining robust overall zero-shot performance.

5.3 Router Utilization Analysis

To verify non-degenerate token-expert assignments, routing statistics were evaluated on the validation split using the trained checkpoint. Aggregate selections across all layers show zero token starvation (Figure 2a), with an empirical mean of 2.0 experts per token matching the theoretical capacity factor ($c = 2.0$). This balanced distribution remains consistent across all six layers without collapse or saturation (Figure 2b). Furthermore, pairwise co-occurrence rates stay within a tight off-diagonal band (Figure 2c), demonstrating uniform token sharing without expert collusion. Notably, the diagonal values in the co-occurrence matrix represent each expert's overall token-selection rate rather than an artifact of self-co-occurrence: because the capacity factor $c = 2.0$ routes each token to two of the four experts on average, each individual expert is included in roughly half of all token–expert assignments, yielding diagonal intensities near 50% rather than 100%.

5.4 Computational Efficiency Analysis

As detailed in Table 4, XpertGPT yields substantial compute savings over the dense baseline. While the model contains 52.6M total parameters, Expert-Choice routing ($c = 2.0, E = 4$) activates an average of two experts per token, capping active parameters per forward pass at 32.14M—a 67.3% reduction relative to the baseline.

Following standard Transformer scaling conventions (Kaplan et al., 2020; Hoffmann et al., 2022; Narayanan et al., 2021), forward pass compute is $2 \times P_{\text{active}}$ FLOPs per token, while training requires $6 \times P_{\text{active}}$ FLOPs per token. For sequence length $T = 512$, XpertGPT requires 32.9 GFLOPs per forward pass (1,928.4 TFLOPs total per epoch across the 10M-token corpus), compared to 100.8 GFLOPs per pass (5,904 TFLOPs total per epoch) for the baseline. Benchmarked on an NVIDIA RTX 6000 Ada (batch size 16, sequence length 512), XpertGPT executes steps at an average latency of $\Delta t = 82.8$ ms, yielding a training throughput of **98,937 tokens/second**.

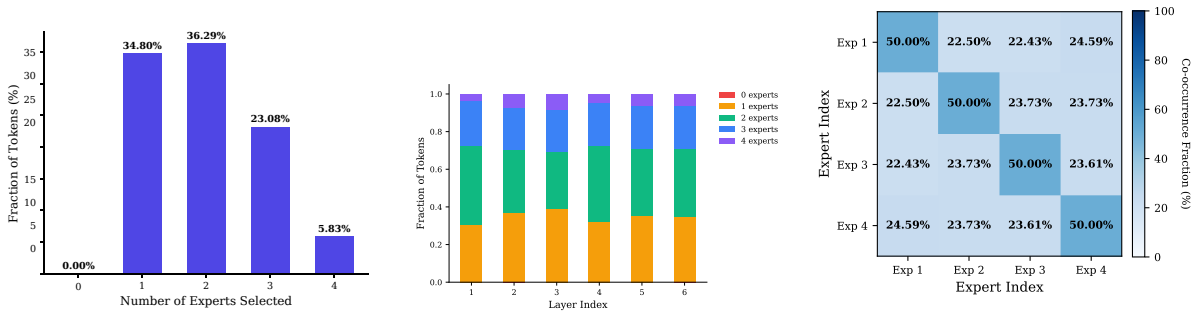
6 Discussion

Core Architectural Takeaways. Three primary structural observations emerge from our control experiments (Table 3). First, removing full-sequence global attention (Ablation 2) causes a collapse in BLiMP (55.72 vs. 64.66) and Entity Tracking (17.49 vs. 21.08), confirming that wide-horizon visibility underpins basic syntactic and contextual consistency. Second, random dispatching (Ablation 3) drops performance across BLiMP (63.36) and Entity Tracking (19.56) compared to XpertGPT (64.66 and 21.08), suggesting that routing benefits stem primarily from learned, content-dependent token alignment rather than parallel windows alone. Third, while activating all experts (Ablation 4) maximizes short-range acceptability on BLiMP (66.70), it degrades macro-reasoning on GlobalPIQA (29.70 vs. 32.65); sparse Expert-Choice gating regularizes representations, successfully balancing localized syntax with macro-sequence tracking. Notably, removing gating alone (Ablation 4) has a smaller net effect on the model's overall capability profile than removing either the global stream (Ablation 2) or replacing learned routing with random dispatch (Ablation 3), suggesting that *whether* multi-scale windows and global context are present matters more than the specific mechanism used to gate access to them.

Architectural design plays a primary role in performance gains. XpertGPT achieves an overall score of 38.42, outperforming the GPT-2 baseline (37.38) while activating only two experts per token on average. Given identical pretraining data, tok-

Ablation Framework	BLiMP	BLiMP Supp	EWoK	Entity Track	COMPS	GlobalPIQA
(1) Global Attention Only (No Experts)	61.73	57.32	49.96	18.68	50.17	33.12
(2) Routed 4-Expert Only (No Global Stream)	55.72	51.00	49.92	17.49	49.63	**35.64**
(3) Random Dynamic Routing ($c = 2.0$)	63.36	57.91	49.77	19.56	50.26	31.20
(4) All Active Experts (No Routing Gating)	**66.70**	59.98	**51.57**	20.46	**51.46**	29.70
(5) XpertGPT (Expert-Choice Final Model)	64.66	**61.00**	51.34	**21.08**	49.62	32.65

Table 3: Controlled ablation results tracking deletions of structural self-attention context, routing token constraints, and expert parameter activations. Bold indicates the best score in each column.



(a) Experts Selected per Token (b) Expert-Count Breakdown by Layer (c) Pairwise Co-Occurrence Matrix

Figure 2: Router utilization statistics across the validation set: (a) Overall distribution of experts activated per token, showing no token starvation; (b) Layer-wise breakdown showing consistent utilization across model depth; (c) Pairwise expert co-occurrence fractions indicating near-uniform token sharing across expert pairs.

Efficiency Metric	XpertGPT	GPT-2 Baseline
Total Parameters	**52.6M**	98.40M
Active Parameters / Token	**32.14M**	98.40M
Forward Pass Compute ($T = 512$)	**32.9 GFLOPs**	100.8 GFLOPs
Total Pretraining Compute ($D = 10$M, per epoch)	**1,928.4 TFLOPs**	5,904 TFLOPs
Step Execution Time (Δt)	**82.8 ms**	–
Training Throughput	**98,937 tok/s**	–

Table 4: Computational complexity and empirical runtime metrics compared against the standard dense GPT-2 baseline under the BabyLM Strict-Small footprint. Total pretraining compute denotes the cost per epoch over the 10M-token corpus; see Appendix A for the full 10-epoch training schedule.

enization, and validation splits, these gains stem from architectural inductive biases rather than scale, supporting prior evidence that design choices dominate in low-resource regimes (Warstadt et al., 2023; Charpentier and Samuel, 2024).

Expert-Choice routing outperforms token-choice. Replacing Expert-Choice routing with token-choice top-1 routing and entropy load balancing lowers EWoK from 51.34 to 48.57 (-2.77), though the comparison is mixed across benchmarks overall (Appendix D). Because each expert selects exactly $k = cT/E$ tokens, load balancing holds by construction (§3.3), ensuring equal training signals without tuning auxiliary loss weights (Zhou et al., 2022). This balanced utilization guarantees every scale-specialized pathway processes diverse contextual cues, driving the observed gains.

Each token is refined at two scales on average. With capacity factor $c = 2.0$ and $E = 4$ experts, the router forms $2T$ token-expert assignments per sequence per layer, so on average each token is processed by two of the four sliding-window experts (spans 64, 16, 8, 4) in addition to the dense global stream. Tokens that several experts find salient—typically phrase and clause boundaries—receive more computation, while easier tokens are handled by fewer experts, all at fixed parameter cost.

Multi-scale specialization helps. Expanding from one or two experts to the full four-scale configuration improves EWoK and Entity Tracking, consistent with the view that different linguistic phenomena demand different receptive fields and that any single fixed window is suboptimal under a tight data budget.

7 Conclusion

We introduced XpertGPT, a sparse decoder-only Transformer that pairs a dense global-attention stream with multi-scale sliding-window expert

branches, coordinated by Expert-Choice routing. Under the BabyLM 2026 Strict-Small budget, XpertGPT outperforms the official GPT-2 baseline on Overall Average and NLP Average while activating fewer than a third of the baseline's parameters per token, with the largest gains concentrated on world-knowledge and entity-tracking benchmarks and on psychometric human-likeness metrics. Our ablations indicate that both the dense global-attention pathway and Expert-Choice routing are necessary components, and that removing either causes the largest performance losses. Because the model and baseline are trained on identical data, we attribute these gains to architectural design rather than scale, reinforcing the broader BabyLM finding that architecture, not parameter count, dominates outcomes under strict data constraints.

Limitations

Our experiments are restricted to the BabyLM Strict-Small track (10M words) and ∼50M-parameter scale, leaving billion-parameter regimes and web-scale data unexamined; because the costs of Expert-Choice routing and sliding-window attention both scale sub-quadratically and independently of corpus size, we expect the efficiency gains to hold at larger scales, though this remains an untested extrapolation. Additionally, the official evaluation suite proxies linguistic competence well but omits open-ended generation quality. Finally, while achieving ∼98.9k tokens/sec and a 67% active parameter reduction over GPT-2, single-GPU execution relied on standard PyTorch gather/scatter operations; custom sparse kernels (e.g., Triton or Megablocks) remain for future deployment.

The sequence-level routing does mean XpertGPT is not a strictly online/streaming model as configured: autoregressive sampling one token at a time would require a causal or streaming variant of the router, which we leave to future work.

Acknowledgements

We thank the BabyLM challenge organizers for providing the datasets, baseline implementations, and standardized evaluation pipelines.

References

BabyLM Community. 2026a. BabyLM 2026 GPT-2 strict-small baseline model. Hugging Face Model Checkpoint.

BabyLM Community. 2026b. BabyLM 2026 official leaderboard. Hugging Face Space.

BabyLM Organizers. 2026. BabyLM 2026 evaluation pipeline. Official evaluation code for the BabyLM 2026 Shared Task.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? In *Proceedings of the 2nd BabyLM Challenge at CoNLL.*

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *Preprint*, arXiv:1904.10509.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(1):5232–5271.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030.

Anna Ivanova, Aalok Sathe, Benjamin Lipkin, Sonia Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Josh Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*.

Albert Q. Jiang et al. 2023. Mistral 7B. *Preprint*, arXiv:2310.06825.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Sushant Mehta, Raj Dandekar, Rajat Dandekar, and Sreedath Panat. 2025. Unifying mixture of experts and multi-head latent attention for efficient language models. In *Proceedings of the First BabyLM Workshop*.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Anand Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. 2021. Efficient large-scale language model training on GPU clusters using Megatron-LM. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15.

S. Salhan, R. Diehl Martinez, Z. Goriely, and P. Buttery. 2025. What is the best sequence length for BabyLM? In *Proceedings of the First BabyLM Workshop*. Association for Computational Linguistics.

Noam Shazeer. 2020. GLU variants improve transformer. *Preprint*, arXiv:2002.05202.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

J. Tapaninaho. 2025. MoEP: Modular expert paths for sample-efficient language modeling. In *Proceedings of the First BabyLM Workshop*. Association for Computational Linguistics.

I. Ternovtsii and Y. Bilak. 2026. Mixture of layers with hybrid attention: Parallel thin blocks for sparse transformer compute. *Preprint*, arXiv:2605.09516.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjap, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew Dai, Zhifeng Chen, Quoc Le, and James Laudon. 2022. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*, volume 35, pages 7103–7114.

A Training Hyperparameters

Table 5 lists all model architecture and pretraining hyperparameters.

Parameter	Value
Layers (L)	6
Model dimension (d_{model})	256
Expert dimension (d_{expert})	384
Global attention heads	4
Expert attention heads	6
Experts	4
Capacity factor (c)	2.0
Window sizes (w)	[64,16,8,4]
Vocabulary size ($\|\mathcal{V}\|$)	16,384
Context length (T)	512
Batch size (B)	16
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	0.1
Warmup steps	800
Epochs	10
Gradient clipping	1.0

Table 5: Complete model architecture and pretraining hyperparameter configuration.

B Additional Ablation Studies

B.1 Router Mechanism and Capacity Factor Analysis

We evaluate the routing mechanics of XpertGPT against three operational baselines–an un-gated dense layout activating all blocks, a conventional token-choice top-1 router with entropy load balancing, and a fixed balanced random router–alongside varying capacity factors ($c \in \{0.5, 1.0, 2.0\}$) under the Expert-Choice framework (Zhou et al., 2022).

As demonstrated in Table 6, while bypassing gating entirely leverages maximum parameter weight to excel on short-range benchmarks like BLiMP (66.70), it severely degrades macro-context integration on GlobalPIQA (29.70). Shifting routing responsibility from tokens to experts drastically mitigates this issue; unlike token-choice methods that rely on soft entropy regularization and underperform on EWoK (48.57), Expert-Choice routing guarantees a deterministic token allocation balance by construction. Furthermore, scaling the capacity factor to $c = 2.0$ expands the aggregate per-sequence throughput to $2T$ token–expert assignments per layer, allowing crucial boundary tokens to activate multiple specialized scale-window pathways concurrently to successfully balance strict macro-context representation with acute local processing.

B.2 Multi-Scale Expert Count and Window Size Configurations

We evaluate the impact of the active parallel expert pool size and designated sliding-window spans (w) under a constant capacity factor of $c = 2.0$.

As detailed in Table 7, scaling from narrow single-expert pipelines to our four-expert layout ($w \in \{64, 16, 8, 4\}$) introduces strong contextual advantages, achieving strong conceptual metrics across structural acceptability tasks like EWoK (51.34) and long-range Entity Tracking (21.08). Receptive window clipping regularizes sequence noise effectively, accelerating downstream abstract reasoning patterns under data-constrained pretraining environments (Choshen et al., 2026).

B.3 Secondary Design Choices: Activations and Positional Routing

Finally, we isolate the performance impacts of feed-forward network modules alongside positional features embedded directly within the routing framework.

As shown in Table 8, transitioning from standard MLP blocks to SwiGLU activations yields uniform enhancements to local grammatical sensitivity and tracking on tasks like BLiMP Supplement (61.00) and EWoK (51.34). Appending RoPE to the SwiGLU variant reaches the highest GlobalPIQA score observed in this ablation (35.67), though this configuration is not used in the final model, which instead prioritizes the stronger BLiMP Supplement performance of the no-RoPE-in-router variant.

C Seed Sensitivity and Robustness Analysis

To verify the stability and initialization invariance of our multi-scale architecture, we evaluate XpertGPT across multiple training executions utilizing distinct random seed initializations (Seed $\in \{0, 3407, 42, 1234\}$).

The model displays remarkable robustness to parameter initialization paths, as summarized in Table 9. The bounded standard deviations across grammatical acceptability benchmarks like BLiMP ($\sigma = \pm 0.76$) and structural tracking blocks like COMPS ($\sigma = \pm 0.50$) demonstrate that our combined routing layout minimizes optimization variance under heavily constrained text boundaries.

Ablation Framework	BLiMP	BLiMP Supp	EWoK	Entity Track	COMPS	GlobalPIQA
(1) All Active Blocks (No Gating)	**66.70**	59.98	51.57	20.46	**51.46**	29.70
(2) Conventional Token-Choice Top-1	64.51	**64.14**	48.57	21.07	49.71	**33.12**
(3) Fixed Balanced Random ($c = 2$)	63.36	57.91	49.77	19.56	50.26	31.20
(4) Expert-Choice ($c = 0.5$)	63.18	59.96	**51.64**	20.52	50.35	31.71
(5) Expert-Choice ($c = 1.0$)	63.71	58.36	49.20	19.73	50.18	30.64
(6) Expert-Choice ($c = 2.0$, Main Model)	64.66	61.00	51.34	**21.08**	49.62	32.65

Table 6: Empirical evaluation of various routing strategies, parameter density configurations, and capacity factor variations across zero-shot language benchmarks under the Strict-Small data layout.

Ablation Framework	BLiMP	BLiMP Supp	EWoK	Entity Track	COMPS	GlobalPIQA
(1) One Expert ($w = 64$)	63.18	59.87	50.53	19.43	50.18	31.68
(2) One Expert ($w = 16$)	63.78	60.77	48.89	19.77	50.16	32.20
(3) Two Experts ($w \in \{16, 64\}$)	63.19	60.34	50.02	18.46	**50.58**	31.23
(4) Four Experts ($w \in \{64, 16, 8, 4\}$, Main)	64.66	**61.00**	**51.34**	21.08	49.62	32.65
(5) Four Experts Small ($w \in \{16, 16, 4, 4\}$)	**64.88**	59.91	50.54	**21.19**	50.43	33.59
(6) Four Experts Very Small ($w \in \{8, 8, 4, 4\}$)	63.42	58.81	50.40	20.78	50.57	**34.11**

Table 7: Zero-shot benchmark comparison analyzing variations in parallel expert block counts and local sliding-window attention configurations under the Strict-Small setup. Bold indicates the best score in each column.

Ablation Framework	BLiMP	BLiMP Supp	EWoK	Entity Track	COMPS	GlobalPIQA
(1) Standard MLP (No RoPE in Router)	62.67	59.77	50.54	20.50	50.89	33.94
(2) Standard MLP + RoPE in Router	63.56	58.24	51.00	21.00	**51.00**	30.66
(3) SwiGLU (No RoPE in Router, Main Model)	**64.66**	**61.00**	**51.34**	**21.08**	49.62	32.65
(4) SwiGLU + RoPE in Router	64.41	56.00	50.68	19.99	50.41	**35.67**

Table 8: Zero-shot evaluation isolating feed-forward activation selections and the inclusion of Rotary Position Embeddings (RoPE) inside the dynamic routing layer. Bold indicates the best score in each column.

Experimental Run	BLiMP	BLiMP Supp	EWoK	Entity Track	COMPS	GlobalPIQA
(a) Run with Seed 0	64.15	58.30	**51.64**	19.39	50.55	33.64
(b) Run with Seed 3407	62.97	59.05	51.63	19.54	**50.71**	34.64
(c) Run with Seed 42 (Main)	**64.66**	**61.00**	51.34	**21.08**	49.62	32.65
(d) Run with Seed 1234	63.40	59.70	49.02	20.08	50.02	31.20
Standard Deviation (σ)	$\pm$0.76	$\pm$1.14	$\pm$1.27	$\pm$0.76	$\pm$0.50	$\pm$1.46

Table 9: Seed sensitivity performance and computed sample standard deviation (σ) evaluated over zero-shot text understanding families. Bold indicates the best score in each column.

D Expert-Choice vs. Token-Choice Routing

We compare **Expert-Choice** against **Token-Choice Top-1** routing under identical constraints (see §2.2, §3.3). In Token-Choice, tokens independently select experts, requiring auxiliary entropy load-balancing ($\mathcal{L}_{\text{bal}} = \sum_i p_i \log p_i$, where p_i is mean batch probability, $\alpha = 0.01$). Conversely, Expert-Choice guarantees exact per-expert capacity ($k = cT/E$) by construction, eliminating tuning. As Table 10 shows, this yields a +2.77 EWoK boost (51.34 vs. 48.57) and small BLiMP and Entity Tracking gains; equal multi-scale expert allocation prevents representation collapse in low-resource regimes and simplifies optimization.

Benchmark	Expert-Choice	Token-Choice Top-1	Δ
BLiMP	**64.66**	64.51	+0.15
BLiMP Supp	61.00	**64.14**	-3.14
EWoK	**51.34**	48.57	+2.77
Entity Track	**21.08**	21.07	+0.01
COMPS	49.62	**49.71**	-0.09
GlobalPIQA	32.65	**33.12**	-0.47

Table 10: Downstream zero-shot performance comparison between Expert-Choice and Token-Choice Top-1 routing.

BabyConceptLM: Exploring Dynamic Concept Computation under Restricted Language Exposure

Yizhe Xu **Ming Song** **Danni He** **Jianghao Liu** **Yitong Wang** **Qing Cai***
School of Psychology and Cognitive Science, East China Normal University
3663 North Zhongshan Road, Shanghai 200062, China

Abstract

Human language learning takes place under limited exposure and is shaped by structured regularities in the input, whereas standard Transformers apply a largely uniform sequence of computations at every token position. We ask whether dynamic latent segmentation can provide a useful inductive bias for language modeling under BabyLM-scale data constraints. BabyConceptLM causally encodes tokens, groups contextual states into learned contiguous latent segments, processes the shorter sequence with a deeper latent-level backbone, and returns to token resolution through a leakage-safe causal readout. We evaluate two profiles using recipe-matched token-only controls, with analyses of computational cost and segmentation behavior. Across English, multilingual, and Chinese comparisons, BabyConceptLM trails matched token-only controls by 0.60–1.47 points. Estimated padding-aware dense-forward FLOPs are reduced by 26% and 39% for the English and multilingual 5/8/3 models, respectively, but increase for the Chinese 2/4/2 configuration. Segmentation audits show weak alignment with word boundaries and predominantly length-driven behavior, with the larger model frequently reaching its eight-token segment cap. These results show that latent segmentation can yield a favorable compute–accuracy trade-off in some configurations, but does not by itself produce linguistically structured units or improved predictive performance. BabyConceptLM serves as a diagnostic test of hierarchical latent computation under restricted language exposure.

1 Introduction

The BabyLM Challenge provides a controlled setting for asking which inductive biases support language learning under restricted exposure, where training data cannot simply be scaled up (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). This raises an architectural question: how should computation be allocated across a sequence under limited exposure? Standard Transformer language models apply largely uniform computation at every token position (Vaswani et al., 2017), although linguistic information spans multiple scales, from lexical patterns to longer multi-token units. This motivates testing whether computation can be redistributed more efficiently.

A natural alternative is hierarchical computation: local token representations are integrated into larger latent units, allowing deeper computation on a shorter sequence. Related approaches use sentence representations (LCM team et al., 2024), entropy-defined byte patches (Pagnoni et al., 2025), or learned chunks (Hwang et al., 2026); continuous latent reasoning explores a complementary use of model-internal states (Hao et al., 2025). Dynamic Large Concept Models provides the closest architectural basis, combining token-state segmentation, latent processing, and causal readout (Qu et al., 2026). Multilingual interpretability studies also describe cross-language concept alignment and specialized intermediate processing stages (Wendler et al., 2024; Zhao et al., 2024; Brinkmann et al., 2025; Tamo et al., 2026). Whether learned latent segmentation is useful under strongly constrained language exposure remains unclear.

We test this hypothesis through dynamic latent segmentation. The model predicts boundaries between contiguous token representations without requiring an external segmenter at inference time. Such grouping could compress locally predictable information while reserving deeper computation for higher-level representations. We evaluate two complementary dimensions: the performance–compute trade-off and the extent to which learned boundaries reflect structure beyond segment length.

*Corresponding author: qcai@psy.ecnu.edu.cn

Proceedings of the Second BabyLM Workshop, pages 779–804
October 29, 2026 ©2026 Association for Computational Linguistics

BabyConceptLM adapts the Dynamic Large Concept Models design to BabyLM-scale pretraining using a GPT-BERT-derived Hugging Face scaffold (Charpentier and Samuel, 2024). It causally encodes tokens, predicts boundaries, pools states within each segment, processes the shorter sequence with a deeper latent-level backbone, and returns to token resolution through a leakage-safe causal readout. We use *concept* operationally for learned contiguous latent segments, not assumed semantic or linguistic concepts.

We evaluate token-encoder/concept-backbone/readout layer-count profiles 2/4/2 and 5/8/3 against recipe-matched token-only controls, denoted TA1. We ask three questions. **RQ1**: Can latent-segment computation remain competitive while reducing estimated computation? **RQ2**: Do learned boundaries reflect structure beyond segment length? **RQ3**: What multilingual performance and participant-level brain-predictivity patterns does the model exhibit? The first two questions form the primary architectural test. English–Dutch–Chinese evaluation and a separate ChineseBabyLM case study (Song et al., 2026) provide complementary diagnostics, with fMRI analyses motivated by prior model–brain correspondence studies (Caucheteux and King, 2022).

The results reveal a configuration-dependent trade-off. Across three frozen recipe-matched comparisons, BabyConceptLM trails token-only controls by 0.60–1.47 points on shared non-finetuning task aggregates. Padding-aware dense-forward FLOPs are lower for English and multilingual 5/8/3, but slightly higher for Chinese 2/4/2 (Table 2a). The held-out English segmentation audit shows predominantly length-driven behavior: 2/4/2 concentrates near six-token segments, whereas 5/8/3 frequently reaches its eight-token cap, with weak source-word-start alignment (Appendix Table A20). Multilingual effects are heterogeneous (Table 3), and participant-level Chinese fMRI contrasts remain inconclusive (Table 2b).

These findings show why learned latent units require direct audits: sequence compression alone does not establish meaningful higher-level structure. Our contributions are: (1) a causal hierarchical language model adapting learned latent segmentation to BabyLM-scale pretraining; (2) recipe-matched token-only comparisons with padding-aware computational accounting; and (3) segmentation, multilingual, and exploratory brain-predictivity analyses that characterize the potential and limitations of the learned hierarchy.

2 Method

2.1 Architecture overview

Intuitively, BabyConceptLM encodes tokens causally, predicts boundaries, pools two segment-state sequences, applies the same concept-backbone weights to each, and maps the readout branch back to token resolution. The auxiliary branch provides probe states and optional syntax supervision; the readout branch supplies causal token memory. Figure 1 shows this flow, and Appendix Figure A1 gives the full computation graph.

Given tokens $x_{1:T}$, BabyConceptLM predicts x_{t+1} causally while replacing a uniform token-only stack with a token encoder, a concept backbone, and a token-level readout. We write a profile as token-encoder/concept/readout layer counts, so 2/4/2 has two token, four concept, and two readout layers, whereas 5/8/3 has five, eight, and three. The representation-producing path, with both calls to the shared concept backbone shown explicitly, is

$$\begin{aligned} H &= E(x_{1:T}), & (s, A) &= \Phi(H), \\ Z^{(0)}_{\text{aux}} &= P(H, A), & Z_{\text{aux}} &= M(Z^{(0)}_{\text{aux}}), \\ Z^{(0)}_{\text{rd}} &= P(H, A_{\text{rd}}), & Z_{\text{rd}} &= M(Z^{(0)}_{\text{rd}}), \\ r_t &= D(h_t, Z_{\text{rd}}, A_{\text{rd}}), & u_t &= F(r_t, h_t, p_t). \end{aligned} \tag{1}$$

where E is the causal token encoder, Φ returns boundary scores s and straight-through assignments A, P is the shared pooling/projection module, M is the shared causal concept backbone, D is the leakage-safe readout, and F is optional residual fusion. The reported 2/4/2 profiles set $A_{\text{rd}} = \text{sg}(A)$, whereas the 5/8/3 profiles set $A_{\text{rd}} = A$; sg is forward-pass identity with zero derivative. D uses the corresponding forward hard segment indices to construct its causal memory mask; detachment affects readout-branch pooling gradients, not those indices. The boundary loss acts directly on s. The two calls to M share parameters but are separate executions.

Output logits are computed from u_t; when fusion is disabled, $F(r_t, h_t, p_t) = r_t$. We refer to each learned contiguous latent segment as a concept. The implementation also executes the pool/projection embedded inside Φ before the two

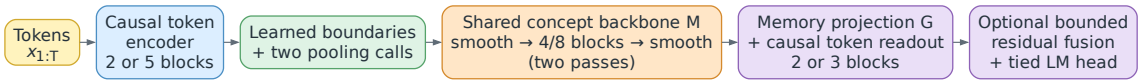


Figure 1: BabyConceptLM overview. Both branches execute the shared backbone, including causal smoothing before and after its blocks. The readout projects completed concept states through G and accesses the causal prefix of the current segment; optional token residual fusion is a separate operation.

named pooling calls; its returned pooled tensors are not consumed downstream. Compute estimates include all three pooling/projection executions and both concept-backbone executions.

2.2 Dynamic segmentation and pooling

The input sums token, position, and, in multilingual training, language embeddings before the token encoder produces $H = (h_1, \ldots, h_T)$. For $t > 1$, the content boundary scorer computes

$$\begin{aligned} s_t &= \mathrm{MLP}[h_t; h_{t-1}; |h_t - h_{t-1}|; \\ &\qquad h_t \odot h_{t-1}; f_t; f_{t-1}] + b, \\ p_t &= \sigma((s_t - \beta)/\tau)\,, \end{aligned} \tag{2}$$

where f_t is a configuration-dependent structural-label embedding and is zero when that input is disabled. Sequence starts, EOS separators and their following positions, and BOS tokens when present are forced boundaries.

The mass decoder scans left to right. It starts a new segment when a forced boundary occurs, accumulated probability reaches a profile-specific threshold ρ, or the current segment has reached $L_{\max} = 8$ tokens; the accumulator is then reset. Frozen-checkpoint summaries record $\rho = 1.0$ and 0.35 for the audited 2/4/2 and 5/8/3 English profiles, respectively. Both use the same hard mass-decoding rule during training and inference. Let $d_t \in \{0, 1\}$ mark a forced sequence/document boundary, $\bar{p}_t = \max(p_t, d_t)$, and $b_t \in \{0, 1\}$ be the decoder's hard boundary indicator. Gradients pass through

$$\widetilde{b}_t = \bar{p}_t + \mathrm{sg}(b_t - \bar{p}_t), \tag{3}$$

whose forward value is b_t and whose gradient follows $\bar{p}_t$. For relaxed segment coordinate $\gamma_t = \sum_{i \le t} \widetilde{b}_i - 1$, let C be the number of active segments in the sequence and normalize over their zero-based indices:

$$\begin{aligned} q_{tc} &= \frac{\exp[-(\gamma_t - c)^2/(2\sigma_A^2)]}{\mathcal{N}_t}, \\ \mathcal{N}_t &= \sum_{j=0}^{C-1} \exp[-(\gamma_t - j)^2/(2\sigma_A^2)]. \end{aligned} \tag{4}$$

for $c = 0, \ldots, C-1$, with $\sigma_A = 0.7$. If $c(t)$ is the hard segment index of token t, then $A_{tc}^{\mathrm{hard}} = \mathbf{1}[c = c(t)]$ and $A = q + \mathrm{sg}(A^{\mathrm{hard}} - q)$. Thus the forward assignment is hard while its gradient follows the Gaussian relaxation. The cap changes the forward hard boundary, but the implementation applies no cap-specific gradient mask: away from forced positions, the straight-through derivative still follows p_t. For branch $k \in \{\mathrm{aux}, \mathrm{rd}\}$, take $A^{(\mathrm{aux})} = A$ and $A^{(\mathrm{rd})} = A_{\mathrm{rd}}$. The shared pooling module initializes active segment c as

$$\begin{aligned} \bar{h}_{c,k} &= \frac{\sum_t A_{tc}^{(k)} h_t}{\max\{1, \sum_t A_{tc}^{(k)}\}}, \\ z_{c,k}^{(0)} &= \bar{h}_{c,k} W_{\mathrm{up}} + e_c^{\mathrm{seg}}. \end{aligned} \tag{5}$$

Here W_{up} is the learned projection from token width to concept width, the denominator reproduces the implementation's lower clamp at one, and e_c^{seg} is the segment-position embedding. In both branches, M applies causal smoothing before and after its shared Transformer stack; Appendix A defines the smoother and architectural settings.

2.3 Leakage-safe token-level readout

Pooling a complete current segment would reveal later tokens within that segment. Let $c(t)$ be the hard segment containing position t. Its causal prefix representation is

$$\begin{aligned} n_{c(t),t} &= \max\left\{1, \sum_{i \le t} \mathbf{1}[c(i) = c(t)]\right\}, \\ \widetilde{z}_{c(t),t} &= \frac{\sum_{i \le t} \mathbf{1}[c(i) = c(t)] h_i}{n_{c(t),t}} W_{\mathrm{up}} + e_{c(t)}^{\mathrm{seg}}. \end{aligned} \tag{6}$$

Within D, $G(z) = \mathrm{Linear}(\mathrm{RMSNorm}(z))$ projects completed backbone states at concept width, independently of optional token residual fusion F. The query at t cross-attends to

$$\mathcal{M}_t = \{G(z_{\mathrm{rd},c}) : 0 \le c < c(t)\} \cup \{\widetilde{z}_{c(t),t}\}, \quad (7)$$

where the current-prefix state bypasses M and G. Consequently, the representation used to predict x_{t+1} contains no position later than t, including within the current segment.

2.4 Bounded token residual fusion

The English 2/4/2 variants and the Chinese BabyConceptLM model combine readout state r_t and token-encoder state h_t with a bounded gate,

$$g_t = g_{\max}\,\sigma(\mathrm{MLP}[r_t; h_t; h_t - r_t; p_t])\,, \quad (8)$$

$$u_t = \mathrm{RMSNorm}(r_t + g_t \odot (h_t - r_t))\,. \quad (9)$$

The gate permits a bounded contribution from token-encoder states while retaining the concept-mediated readout as the main prediction path. The English 5/8/3 variants and the shared multilingual model predict directly from the token-level readout; Table A2 records enabled variants, and gate bounds appear in Appendix A.

2.5 Depth weighted averaging as an auxiliary variant

We test DWA within 2/4/2 (105.6M parameters) and 5/8/3 (194.6M) token/concept/readout profiles. The noDWA and DWA variants share the corpus, exposure, base architecture, and optimization recipe; DWA adds only 24 and 244 scalar coefficients, so these comparisons are near-parameter-matched. The 2/4/2 implementation uses an unconstrained layer-level combination, while 5/8/3 applies softmax-weighted mixtures after each attention and feed-forward residual update. DWA operates in the token, concept, and readout paths. Cross-profile contrasts also change depth, size, schedule, and output fusion and are not DWA ablations; implementation details appear in Appendix A.

2.6 Matched controls and FLOPs accounting

The 19-layer English and multilingual controls use width 768 and 192.4M parameters, versus 194.6M for 5/8/3; the 10-layer Chinese control has 107.4M parameters, versus 105.6M for 2/4/2. Within each frozen pair, corpus, tokenizer, exposure, optimizer/schedule, batch construction, and seed are held fixed. Our compute metric is estimated dense-forward FLOPs: counts cover matrix products in every executed path, including both concept-backbone passes, and assign two FLOPs per multiply–accumulate. Padding-aware values allocate the within-microbatch maximum concept length over every dev sequence. Appendix A specifies the formulas, tensor shapes, batching, and exclusions.

2.7 Training objective

All reported pretraining checkpoints use causal next-token prediction only; the retained training arguments set MNTP sampling probability to zero throughout. The enabled loss is

$$\mathcal{L} = \mathcal{L}_{\mathrm{CLM}} + \lambda_{\mathrm{syn}}\mathcal{L}_{\mathrm{syn}} + \lambda_{\mathrm{bnd}}\mathcal{L}_{\mathrm{bnd}} + \lambda_{\mathrm{seg}}\mathcal{L}_{\mathrm{seg}} + 10^{-4}\mathcal{L}_z. \quad (10)$$

Boundary supervision is one binary cross-entropy on a composite soft target, not a sum of separately weighted boundary losses. The target combines forced document boundaries, detached adjacent-state cosine change, and enabled tokenizer-derived heuristic-label or phrase-role transitions. In Chinese, an aligned LTP word-start label replaces this composite target before forced document boundaries are restored. The segmentation term regularizes expected and straight-through boundary rates, active-segment lengths, and optionally boundary entropy. Appendix A.1 formally defines every target, coefficient, index, clamp, and loss term and records the profile-specific weights.

3 Pretraining and evaluation

The four English models form two near-parameter-matched DWA ablations. Because the 5/8/3 profile was more competitive in the English NLP and exposure-indexed analyses, we use that profile for the shared MULTILINGUAL entry rather than conducting a second architecture search across languages. The ChineseBabyLM case study instead uses 2/4/2 to retain the smaller 105.6M-parameter budget and bounded token residual, which preserves a direct route for character-level evidence in Hanzi and fMRI evaluation (Song et al., 2026).

3.1 Corpora and pretraining

English STRICT. The English models use the official 100M-word STRICT corpus (Choshen et al., 2026) and a case- and punctuation-preserving 16,384-entry NFKC ByteLevel BPE tokenizer. Tokenized documents are concatenated

with EOS separators and then sliced into fixed-length packed sequences; a sequence may therefore contain portions of multiple documents, while the separator-aware boundary mechanism prevents an unmarked transition between them.

MULTILINGUAL. The multilingual mixture contains 43.095M English, 40M Dutch, and 15M Chinese raw words, corresponding to 100M Byte-Premium-adjusted words under the official factors (Choshen et al., 2026). A shared 16,384-entry tokenizer is trained round-robin; each language is packed separately and identified by a language embedding.

Chinese case study. The Chinese model uses 101.6M Jieba-counted tokens filtered from the `baike2018qa` encyclopedia-question-answer corpus (hereafter BaikeQA) (Xu, 2019), with 80% used for training. A 16,384-entry NFKC Metaspace BPE tokenizer is trained from scratch. Offline N-LTP processing (Che et al., 2021) provides word-start and part-of-speech (POS) sidecar labels. The retained environment and cache resolve PyPI `ltp==4.2.14` and `LTP/small@dea87bf0` (full hash in Appendix B). The original preprocessing run recorded the model name but not its revision in a lock file. No LTP parameters, hidden states, or embeddings initialize or enter BabyConceptLM, and LTP is absent at evaluation. Appendix B gives the filtering and accounting procedure.

Matched controls. Each TA1 model removes segmentation, concept, and readout modules while retaining its paired corpus, tokenizer, exposure sequence, packed batches, AdamW configuration and schedule, and seed 42. English TA1 matches English matched BabyConceptLM (token/concept/readout DWA, initialization 8/8/8), and multilingual TA1 matches Multilingual matched BabyConceptLM (DWA, 6/4/4); the matched references are archived recipes, with public-final submissions reported separately. The Chinese pair also shares the absence of DWA. Its intervention compares the complete concept pathway with token-only depth allocation: boundary prediction, the concept backbone, token-level readout, and layer allocation vary jointly, with a 1.7% parameter-count difference. Appendix Tables A8 and A9 give pinned identities and public-final recipe differences, respectively.

All models are initialized from scratch and trained with AdamW, mixed precision, gradient clipping, and seed 42. Exposure is accumulated from the actual whitespace word counts of consumed microbatches, with Byte Premiums applied per language for MULTILINGUAL. The four STRICT runs stop at exactly 1B words, and the multilingual run reaches the 1B adjusted-word target within floating-point tolerance. Optimization and checkpoint details appear in Appendix Table A4.

3.2 Evaluation

We pin the official `babylm-eval` repository at `6f825c2`[1] and `chinese-babylm-pipeline-final` at `f4437e1`[2]. Source benchmarks include BLiMP (Warstadt et al., 2020), EWoK (Ivanova et al., 2025), Entity Tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), and GLUE/SuperGLUE (Wang et al., 2019b,a). Challenge-specific variants and aggregation follow the pinned pipeline. STRICT Reading is normalized incremental R^2, not comprehension accuracy; AoA is 100 times the model–child correlation and is zeroed when $p > .1$. Appendix D gives the complete task/source inventory and metric caveats.

For Chinese fMRI, matched participant IDs give $n = 12$ in six discourse domains and $n = 11$ in Word-fMRI. The focal exploratory Cognition and Language contrasts use paired bootstrap intervals, exact sign-flip tests, Holm correction, and paired effect d_z (Appendix D.1). These tests quantify participant sampling uncertainty conditional on two seed-42 models, not training-seed variance.

Appendix C records checkpoint identities, recipe differences, and score verification; Appendix E gives the English Hippocorpus audit protocol (Sap et al., 2020).

4 Results

English STRICT performance, matched controls, and compute accounting address RQ1; the frozen segmentation audit addresses RQ2; and the multilingual and Chinese analyses address RQ3. Scores are percentages unless stated otherwise.

[1] Full commit: https://github.com/babylm-org/babylm-eval/tree/6f825c291e2c4c78ad33b1935fd64d45f52642dc

[2] Full commit: https://github.com/chinese-babylm/chinese-babylm-pipeline-final/tree/f4437e1aaecf18ab5defb6b148857ef7d056e9d6

4.1 English STRICT performance and estimated forward compute

The primary leaderboard summary is the official *Overall* aggregate (hereafter *Overall*). "Strict-Interaction" reproduces the official baseline label and scores from the pinned evaluation README; it is the 98.4M-parameter public artifact at commit `ba5a7aa`, not a model trained in this study[3].

All four English variants have complete final evaluations. The 5/8/3 variants are the more competitive profile: noDWA is 1.09 Overall points below GPT-2 but only 0.13 lower on NLP, while scoring 1.09 higher on BLiMP and 2.55 higher on GlobalPIQA. Within-profile DWA effects remain mixed. In 2/4/2, DWA changes Overall by -1.41, NLP by -0.33, and (Super)GLUE by -0.73; in 5/8/3, the corresponding changes are $+0.16$, -0.22, and -0.35. The larger 2/4/2 Overall difference is driven primarily by the thresholded Human-like component rather than its NLP aggregate. DWA therefore changes task balance without a consistent aggregate improvement.

Matched token-only control. All main non-finetuning evaluations are complete for the three frozen TA1 pairs. Table 2 separates these within-pair comparisons from the public-final rows above. The English shared macro contains exactly BLiMP, BLiMP Supplement, EWoK, Entity Tracking, and COMPS. Token-only TA1 leads its paired concept reference by 1.47 English points, 0.60 multilingual points, and 0.82 Chinese points. These single-seed pairs estimate non-finetuning performance for the frozen matched recipes; public-final comparisons are reported separately. Because TA1 finetuning is incomplete, Appendix Table A10 reports the shared non-finetuning tasks rather than an *Overall* control score.

Under this FLOPs metric, complete-dev random batching gives 25.94% and 39.33% estimated savings for English and multilingual 5/8/3, whereas Chinese batches make 2/4/2 5.5% more expensive than TA1. Appendix Tables A6–A7 give the distributions and exclusions.

The 2/4/2 profile remains weaker on NLP after complete finetuning. Its Human-like peak is driven by a marginal AoA correlation ($r = .104$, $p = .091$) crossing the pipeline's $p \leq .1$ threshold; Reading is 0.13–0.22 and has a target-tokenization mismatch. We therefore retain the official values but do not treat them as robust cross-tokenizer cognitive rankings (Appendix D).

4.2 Exposure-indexed learning

All five BabyLM submissions have fast evaluations at the 28 required revisions. STRICT curves macro-average BLiMP, its Supplement, EWoK, Entity Tracking, GlobalPIQA, and official Reading; tokenizer choice is controlled within each English pair, but the macro inherits the Reading mismatch. MULTILINGUAL curves average official zero-shot families within language and then across languages; panels are not compared because their inventories differ.

Both 5/8/3 variants cross 60 BLiMP between the saved 20M and 30M checkpoints, whereas both 2/4/2 variants cross between 300M and 400M. Thus 30M and 400M are checkpoint upper bounds, not exact acquisition times. Within-profile noDWA/DWA crossings coincide, and their log-exposure AUCs differ by at most 0.26.

Appendix Figure A2 and Table A16 provide the complete trajectories and numerical anchors.

This comparison is recipe-level: relative to 2/4/2, 5/8/3 nearly doubles parameters and blocks, receives about twice as many updates per early token, uses larger learning rates, and adds heuristic syntax/phrase supervision. Isolating the contribution of depth, segmentation, or any other factor requires matched interventions; Appendix Table A5 enumerates the co-varying factors.

4.3 English held-out segmentation behavior

Table A20 quantifies concentration around six-token segments, saturation at the eight-token cap, and alignment with source-word starts. The learned boundary rate excludes window starts and cap-attributed cuts; word-start AUC measures threshold-independent alignment between boundary probabilities and source-word starts.

The 2/4/2 profiles concentrate near six tokens but retain measurable length variation; six-token segments account for 70.16% (noDWA) and 63.32% (DWA), with word-start AUCs of 0.543 and 0.546 indicating weak word-start sensitivity. The 5/8/3 profiles are predominantly cap-driven: DWA raises $P(L = 8)$ by 11.42 points and the cap-attributed boundary share from 58.20% to 69.63%, with word-start AUC 0.448 in both variants. Together, these results characterize length-dominated model-grounded segmentation;

[3] Hugging Face artifact at the full pinned revision.

Model	Overall	NLP	Human-like	BLiMP	GlobalPIQA	(Super)GLUE
GPT-2 Strict	**40.73**	**53.03**	−2.32	74.73	36.62	**67.75**
Strict-Interaction	39.89	52.42	−3.96	72.89	36.18	67.28
BabyConceptLM 2/4/2 noDWA	38.67	48.21	**5.27**	62.35	35.67	64.45
BabyConceptLM 2/4/2 DWA	37.26	47.88	0.09	62.44	34.17	63.72
BabyConceptLM 5/8/3 noDWA	39.64	52.90	−6.78	75.82	**39.17**	67.45
BabyConceptLM 5/8/3 DWA	39.80	52.68	−5.29	**76.58**	37.59	67.10

Table 1: English STRICT results at 1B word exposures. All four BabyConceptLM rows include complete zero-shot and seven-task finetuning evaluations. Baseline rows are recomputed from the official evaluation README. Human-like is the pipeline-reported mean of Reading and AoA and inherits the metric caveats described in the text. Complete benchmark-family and finetuning results appear in Appendix Tables A11–A12.

(a) Matched non-finetuning performance and forward FLOPs

Frozen pair	Concept	TA1	Δ score	Ideal GF C/T	Padded GF C/T	Δ padded
English 5/8/3 matched BabyConceptLM–TA1-19L†	52.74	54.21	−1.47	72.14/102.07	75.59/102.07	−25.9%
MULTI 5/8/3–TA1-19L†	50.43	51.02	−0.60	61.59/102.07	61.92/102.07	−39.3%
Chinese 2/4/2–TA1-10L	57.73	58.55	−0.82	47.07/56.77	59.91/56.77	+5.5%

(b) Paired Chinese CogBench contrasts

Contrast	n	Δ (C−T)	95% CI	d_z	Exact p / Holm p	C>T
Cognition	12	+0.288	[−0.274, 0.797]	0.29	.324/.436	9/12
Language	12	+0.281	[−0.118, 0.687]	0.38	.218/.436	7/12
Cognition + Language mean	12	+0.284	[−0.086, 0.645]	0.42	.174/–	9/12
All six discourse domains	12	+0.178	[−0.109, 0.448]	0.35	.257/–	9/12
Word-fMRI	11	−0.159	[−0.566, 0.251]	−0.22	.496/–	4/11

Table 2: Matched-control results. Panel (a): scores are concept minus token. English averages the five explicitly named shared families in the text; MULTILINGUAL is the official fixed-denominator mean of language-level zero-shot averages; Chinese averages ZhoBLiMP, XCOMPS-ZH, and open/hidden Hanzi structure/pinyin. Daggers mark the pinned matched-recipe concept repositories, which are distinct from the public-final submission checkpoints. Ideal FLOPs evaluate the analytic formula at each frozen checkpoint's full-dev mean concept count. For Chinese, all 89,909 packed dev sequences give $\bar{C} = 86.883$ concepts per sequence and hence 47.07 GF. English/MULTILINGUAL padded values average batch maxima over 200 complete-dev-set random groupings at $T = 256$, $B = 16$; Chinese uses all 89,909 dev sequences at $B = 32$. Negative padded deltas indicate savings. Panel (b): correlations are multiplied by 100; C/T denote concept/token. CIs use paired bootstrap; p-values use exact two-sided sign-flip tests. Holm values in the two focal exploratory rows correct Cognition and Language.

full distributions and paired intervals appear in Appendix E. The nearly closed 2/4/2 fusion gate suppresses the direct token-level residual, although h_t still serves as the readout query.

4.4 Multilingual results

The public multilingual entry is 0.54 points above trilingual GPT-2 and narrows the language spread from 4.29 to 1.61; lower Dutch zero-shot together with higher Chinese *Overall* and Dutch/Chinese finetuning show heterogeneous transfer. This public comparison differs in mixture, size, tokenizer, and architecture; the matched zero-shot pair is reported separately, with TA1 at 51.02 versus 50.43 for 5/8/3. Appendix Tables A13–A15 report all tasks. MECO is excluded from the aggregate; displayed MECO and Hanzi scores were public-leaderboard-verified at manuscript freeze.

Block	Model	Multi	EN	NL	ZH	Spread
Overall	GPT-2 tri.	45.94	**47.60**	**46.92**	43.31	4.29
	Ours	**46.48**	47.47	46.11	**45.86**	**1.61**
Zero-shot	GPT-2 tri.	n/a	53.63	**55.05**	47.25	n/a
	Ours	n/a	**53.75**	50.73	**49.52**	n/a
Finetune	GPT-2 tri.	n/a	**43.09**	39.82	40.69	n/a
	Ours	n/a	42.77	**42.06**	**43.42**	n/a

Table 3: Public MULTILINGUAL aggregates for babyconceptLM-multi (Ours) and the official trilingual GPT-2 reference. Spread is the range across language Overall scores.

4.5 Chinese linguistic and cognitive trade-off

We analyze ChineseBabyLM as a separate case study outside the BabyLM 2026 submission. Ap-

pendix Table A17 separates matched TA1 from published and unmatched baselines (Charpentier and Samuel, 2024; Song et al., 2026). The ChineseBabyLM 2026 final-leaderboard Space is accessible and lists babyconceptLM-baike102M_bpe40; its displayed task scores match the retained outputs produced by pinned pipeline commit `f4437e1`. fMRI correlations are multiplied by 100.

Across six linguistic/Hanzi measures, TA1 averages 58.55 versus 57.73; BabyConceptLM is higher only on hidden pinyin (+0.05). Word-fMRI also favors TA1, 56.77 versus 56.61, whereas mean discourse fMRI favors BabyConceptLM, 10.92 versus 10.74.

Participant contrasts remain inconclusive: Cognition is +0.288 for BabyConceptLM (exact $p = .324$, Holm $p = .436$), Language is +0.281 ($p = .218/.436$), and their participant-wise mean is +0.284 with 95% CI $[-0.086, 0.645]$ and $p = .174$. No domain survives six-way Holm correction; Reward favors TA1 before correction ($p = .032/.193$). Appendix Figures A3–A5 instead use an unmatched GPT-BERT baseline and are descriptive.

5 Discussion and Conclusion

This work investigates whether learned latent segmentation can provide a useful inductive bias for language modeling under BabyLM-scale exposure. BabyConceptLM groups tokens into latent segments and processes the shorter sequence before returning to token-level prediction. Across three frozen recipe-matched comparisons, it scores 0.60–1.47 points below token-only controls on the evaluated non-finetuning aggregates. Its computational benefit depends on the configuration: padding-aware dense-forward estimates are 26% lower for English and 39% lower for multilingual 5/8/3, but 6% higher for Chinese 2/4/2. Thus, latent compression offers a configuration-dependent trade-off between estimated computation and predictive performance.

The held-out English audit shows that learned segmentation is dominated by segment length and the imposed length cap, with weak alignment to word starts. A learned boundary mechanism can therefore settle into length-driven grouping under the present objectives. Future hierarchical models should test stronger objectives or constraints that encourage linguistically informed segmentation, together with fixed-length and random-boundary controls to establish the value of learned boundaries.

Overall, BabyConceptLM provides a controlled study of latent segmentation under limited language exposure. Its findings highlight three priorities for evaluating alternative architectures: verify what the learned segments capture, account for the computation actually allocated under padding, and compare predictive performance under matched training conditions. Together, these checks clarify when hierarchical computation offers a useful trade-off for small-scale language learning.

Limitations

First, the learned segmentation remains largely length-driven. Our English narrative audit measures word-start alignment; testing syntactic, semantic, and discourse boundaries and other languages requires broader evaluation. Stronger structural objectives and fixed-length or random-boundary controls would help assess which benefits depend on learned boundaries.

Second, some matched token-only comparisons jointly change segmentation, latent computation, readout, and layer allocation, leaving individual component effects to be isolated. The controls cover shared non-finetuning tasks, with English and multilingual pairs based on archived recipes rather than final submissions. The 30M-versus-400M BLiMP contrast also compares recipes differing in model size, optimization, and supervision, so it identifies a recipe-level learning-speed difference.

Third, experiments use a single training seed and BabyLM-scale models. Multiple seeds and larger models are needed to assess robustness across runs and scales. Participant-level fMRI effects are small and inconclusive, limiting cognitive interpretation.

Finally, our compute analysis estimates dense forward FLOPs, including batch padding. End-to-end speed, peak memory, and energy benefits require direct hardware measurements.

References

Jannik Brinkmann, Chris Wendler, Christian Bartelt, and Aaron Mueller. 2025. Large language models share representations of latent grammatical concepts across typologically diverse languages. In *Proceedings of the 2025 Conference of the Nations of the*

Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 6131–6150, Albuquerque, New Mexico. Association for Computational Linguistics.

Charlotte Caucheteux and Jean-Rémi King. 2022. Brains and algorithms partially converge in natural language processing. *Communications Biology*, 5:134.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Wanxiang Che, Yunlong Feng, Libo Qin, and Ting Liu. 2021. N-LTP: An open-source neural language technology platform for Chinese. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 42–49, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *arXiv preprint arXiv:2602.20092*.

Wenhan Han, Yifan Zhang, Zhixun Chen, Binbin Liu, Haobin Lin, Bingni Zhang, Taifeng Wang, Mykola Pechenizkiy, Meng Fang, and Yin Zheng. 2025. MuBench: Assessment of multilingual capabilities of large language models across 61 languages. *arXiv preprint arXiv:2506.19468*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2025. Training large language models to reason in a continuous latent space. In *Proceedings of the Second Conference on Language Modeling*.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schütze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81, Vienna, Austria. Association for Computational Linguistics.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Sukjun Hwang, Brandon Wang, and Albert Gu. 2026. Dynamic chunking for end-to-end hierarchical sequence modeling. In *International Conference on Learning Representations*.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Jaap Jumelet, Leonie Weissweiler, and Arianna Bisazza. 2025. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *arXiv preprint arXiv:2504.02768*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

LCM team, Loïc Barrault, Paul-Ambroise Duquenne, Maha Elbayad, Artyom Kozhevnikov, Belen Alastruey, Pierre Andrews, Mariano Coria, Guillaume Couairon, Marta R. Costa-jussà, David Dale, Hady Elsahar, Kevin Heffernan, João Maria Janeiro, Tuan Tran, Christophe Ropers, Eduardo Sánchez, Robin San Roman, Alexandre Mourachko, and 2 others. 2024. Large concept models: Language modeling in a sentence representation space. *arXiv preprint arXiv:2412.08821*.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. 2025. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9238–9258, Vienna, Austria. Association for Computational Linguistics.

Xingwei Qu, Shaowen Wang, Zihao Huang, Kai Hua, Fan Yin, Rui-Jie Zhu, Jundong Zhou, Qiyang Min, Zihao Wang, Yizhi Li, Tianyu Zhang, He Xing, Zheng Zhang, Yuxuan Song, Tianyu Zheng, Zhiyuan Zeng, Chenghua Lin, Ge Zhang, and Wenhao Huang. 2026. Dynamic large concept models: Latent reasoning in an adaptive semantic space. In *The Fourteenth International Conference on Learning Representations*.

Maarten Sap, Eric Horvitz, Yejin Choi, Noah A. Smith, and James Pennebaker. 2020. Recollection versus imagination: Exploring human memory and cognition via neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1970–1978, Online. Association for Computational Linguistics.

Noam Siegelman, Sascha Schroeder, Cengiz Acartürk, Hee-Don Ahn, Svetlana Alexeeva, Simona Amenta, Raymond Bertram, Rolando Bonandrini, Marc Brysbaert, Daria Chernova, Sara Maria Da Fonseca, Nicolas Dirix, Wouter Duyck, Argyro Fella, Ram Frost, Carolina A. Gattei, Areti Kalaitzi, Nayoung Kwon, Kaidi Lõo, and 12 others. 2022. Expanding horizons of cross-linguistic research on reading: The multilingual eye-movement corpus (MECO). *Behavior Research Methods*, 54(6):2843–2863.

Siyuan Song, Zhiheng Qian, Yunhao Zhang, Linyang He, Xiaozhe Ji, Yingxin Lin, Hongao Zhu, Chongtian Shao, Chuhan Lang, Luan Li, Rui Wang, Renfen Hu, Shaonan Wang, and Hai Hu. 2026. The first ChineseBabyLM challenge: Training data-efficient and cognitively plausible language models for Chinese. *arXiv preprint arXiv:2607.10745*.

J. Ben Tamo, Daniel Carlander-Reuterfelt, Jonathan Rubin, Oleg Poliannikov, Dezhi Hong, and Mingxian Wang. 2026. LinguaMap: Which layers of LLMs speak your language and how to tune them? In *The Fourteenth International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Bright Xu. 2019. NLP Chinese Corpus: Large scale chinese corpus for NLP.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024. How do large language models handle multilingualism? In *Advances in Neural Information Processing Systems 37*, pages 15296–15319. Curran Associates, Inc.

A Implementation and optimization details

Token and readout blocks have width 768 with 3,072-dimensional feed-forward layers and 12 attention heads; concept blocks have width 896 with 3,584-dimensional feed-forward layers and 14 heads. All blocks use RMSNorm, GEGLU feed-forward layers, normalized queries and keys, and tied input–output token embeddings. Hidden and attention dropout are 0.1; activation dropout is zero.

Concept smoothing and memory projection. Writing $\mathcal{B}$ for the causal concept Transformer stack, including enabled DWA, the shared backbone is $M = S \circ \mathcal{B} \circ S$. For an active concept position c with input state v_c, the causal smoother is

$$S(v)_c = (1 - \alpha_S)v_c + \frac{\alpha_S}{|\mathcal{J}_c|} \sum_{j \in \mathcal{J}_c} v_j,$$

where $\mathcal{J}_c$ contains the valid concept indices from $\max(0, c - k_S + 1)$ through c; padded outputs are zero. All reported concept profiles use $k_S = 3$ and $\alpha_S = 0.35$ for both smoothing calls in each branch. The readout applies G, an RMSNorm followed by an affine 896×896 projection, to completed concept memory. Current-prefix states use the token-to-concept pooling projection directly and bypass M and G. The projection G is distinct from the optional token residual fusion F.

Path	Width	FFN	Shared settings
Token encoder	768	3,072	RMSNorm, GEGLU
Concept backbone	896	3,584	normalized Q/K
Token-level readout	768	3,072	tied embeddings

Table A1: Architectural dimensions shared across profiles.

For path states $v_0, \ldots, v_i$, DWA computes $\widehat{v}_i = \sum_{j=0}^{i} \alpha_{ij} v_j$. The 2/4/2 coefficients are layer-level, unnormalized, and initialized as $(0, \ldots, 0, 1)$, selecting the current residual state. In 5/8/3, each attention sub-block first updates the residual state, which is appended to the state history; DWA then mixes that history and replaces its latest entry with the mixture. The same sequence follows the feed-forward residual update. These coefficients are softmax normalized; English initializes prior-state logits to 0 and the current-state logit to 4. The multilingual model uses current-state logits 6, 4, and 4 in the token, concept, and readout paths. DWA adds 24 and 244 parameters to the English profiles, respectively. The bounded fusion gate uses $g_{\max} = 0.40$ for the English 2/4/2 models and 0.45 for babyconceptLM-baike102M_bpe40.

The English heuristic syntax lookup maps tokenizer pieces to eight buckets: noun-like suffixes, verb-like forms, modifiers, function words, linkers/punctuation, prepositions, clause markers, and a residual/special bucket. A separate eight-way phrase-role lookup uses other/content, verb, adjective, determiner, preposition, linker, clause, and special roles. Phrase starts are generated deterministically at major heads, object starts, and linker/special transitions, except preposition–object continuations. The two builders contain 324 hard-coded set memberships (218 unique normalized strings) and 25 unique suffix endings. They are literal constants in our source, applied only to the in-budget tokenizer vocabulary; no NLTK resource, dictionary file, external corpus lookup, tagger, or parser is loaded, and they add no training text. Code inspection cannot retrospectively establish what references, if any, informed the authors' original item selection.

A.1 Objective and compute definitions

The retained training arguments for every reported BabyConceptLM checkpoint set `mntp_final_ratio`, `mntp_late_ratio`, `mntp_start_ratio`, and `mntp_ramp_end_ratio` to zero. A legacy `masked_fraction` field remains serialized but is not consulted once the scheduled MNTP probability is zero. Every training batch therefore enters `mode=clm`, which enables triangular masks in token and concept self-attention.

Boundary supervision is a single binary cross-entropy. Its composite target is

$$\begin{aligned} y_t = \max\{ & y_t^{\text{doc}}, \tfrac{1}{2} d_{\cos}(\text{sg}(h_{t-1}), \text{sg}(h_t)), \\ & \tfrac{3}{5} y_t^{\text{syn}}, \tfrac{17}{20} y_t^{\text{phr}} \}, \\ \mathcal{L}_{\text{bnd}} = & \ \text{BCEWithLogits}(s_t, y_t), \end{aligned} \tag{11}$$

where $d_{\cos}(a, b) = (1 - \cos(a, b))/2$; $y_t^{\text{doc}} = 1$ at forced sequence/document boundaries and is zero elsewhere; and y_t^{syn} and y_t^{phr} indicate an adjacent change in the heuristic-label and phrase-role categories defined above, including the deterministic phrase-start rules. Disabled structural targets are omitted. A valid aligned Chinese LTP word-start label replaces the composite target before forced

Model	Setting / profile	DWA / output fusion	Boundary target / structural input or loss	Size
babyconceptLM-en-noDWA	STRICT / 2/4/2	None / bounded residual	Semantic change / none	105.6M
babyconceptLM-en-DWA	STRICT / 2/4/2	Layer linear / bounded residual	Semantic change / none	105.6M
babyconceptLM-en-noDWA	STRICT / 5/8/3	None / readout only	Semantic + heuristic phrase / heuristic syntax + $\mathcal{L}_{\mathrm{syn}}$	194.6M
babyconceptLM-en-DWA	STRICT / 5/8/3	Sub-block softmax / readout only	Semantic + heuristic phrase / heuristic syntax + $\mathcal{L}_{\mathrm{syn}}$	194.6M
TA1 token-only	STRICT / 19 token layers	Token DWA / token only	None / none	192.4M
babyconceptLM-multi	MULTILINGUAL / 5/8/3	Sub-block softmax / readout only	None / none	194.6M
TA1 token-only	MULTILINGUAL / 19 token layers	Token DWA / token only	None / none	192.4M
babyconceptLM-baike102M_bpe40	ChineseBabyLM / 2/4/2	None / bounded residual	LTP word labels; composite fallback / none	105.6M
TA1 token-only	ChineseBabyLM / 10 token layers	None / token only	None / none	107.4M

Table A2: Model configurations and supervision sources. In the supervision column, the slash separates the boundary target from structural input or loss. For Chinese, valid aligned LTP word labels replace the composite target; the latter is used only as a fallback at invalid or unaligned positions. Forced document boundaries and the eight-token decoding cap apply across concept profiles and are not listed as supervision.

document boundaries are restored. For English 5/8/3, $\mathcal{L}_{\mathrm{syn}}$ predicts the majority heuristic bucket in segment $c+1$ from auxiliary-branch state c.

Let ℓ^* be the target tokens per segment, $r^* = 1/\ell^*$ its target boundary rate, $\bar{p}$ the valid-token mean of p_t, $\bar{b}$ the valid-token mean of $\widetilde{b}_t$, and ℓ_c the length of active segment c. For valid positions V, $N = |V|$, and probabilities clamped within logarithms, define

$$\mathcal{H}(p) = -\tfrac{1}{N}\sum_{t\in V}\big[p_t \log p_t + (1-p_t)\log(1-p_t)\big], \tag{12}$$

$$\mathcal{L}_{\mathrm{seg}} = w_{\exp}(\bar{p}-r^*)^2 + w_{\mathrm{hard}}(\bar{b}-r^*)^2 + w_{\mathrm{len}}\mathcal{L}_{\mathrm{len}} + w_{\mathrm{ent}}\mathcal{H}(p), \tag{13}$$

$$\mathcal{L}_{\mathrm{len}} = \frac{1}{C}\sum_{c=0}^{C-1}\Big([L_{\min}-\ell_c]_+^2 + [\ell_c - L_{\max}]_+^2 + w_*[(\ell_c-\ell^*)/\ell^*]^2\Big), \tag{14}$$

where $[a]_+ = \max(a,0)$, C is the number of active segments, $L_{\min}$ and $L_{\max}$ are configured bounds, and the four w coefficients plus w_* are nonnegative profile-specific weights. The implementation averages the final length term over all active segments in the minibatch. For vocabulary logit o_{tv} at a valid target, $\mathcal{L}_z = N^{-1}\sum_t(\log\sum_v e^{o_{tv}})^2$.

For dense-forward accounting, a self-attention block at sequence length L, width d, and GEGLU feed-forward width f contributes

$$F_{\mathrm{self}} = 8Ld^2 + 4L^2d + 6Ldf, \tag{15}$$

using two FLOPs per multiply–accumulate. The dense readout concatenates C_B concept slots and T current-prefix slots, allocating memory length $M_B = C_B + T$, where C_B is the within-microbatch maximum concept count. Its causal mask restricts access without shrinking these dense tensors. With token width d, concept width d_c, and readout GEGLU width f_{rd}, one readout block contributes

$$F_{\mathrm{rd}} = 4Td^2 + 4M_Bd_cd + 4TM_Bd + 6Tdf_{\mathrm{rd}}.$$

This counts query/output projections at length T, key/value projections at length M_B, both $T\times M_B$ attention products, and GEGLU. Evaluated shapes are $T=256$, $d=768$, $d_c=896$, token/readout FFN 3,072, and concept FFN 3,584. Counts also include boundary projections, all three pooling/projection calls, both shared backbone executions, prefix-current pooling, the concept-memory projection G once ($2C_Bd_c^2$), tied vocabulary projection, and the enabled Chinese gate. They exclude embedding lookup, normalization, smoothing averages, nonlinearities, softmax, loss, optimizer operations, segmentation control flow, and hardware utilization. English and multilingual use 200 complete-dev-set random partitions at $B=16$; Chinese uses all 89,909 dev sequences at $B=32$. These estimates cover matrix operations rather than profiler-measured runtime.

For the audited English 2/4/2 models, the frozen summaries record $\rho = 1.0$, $\ell^* = 4.0$, and $(w_{\exp}, w_{\mathrm{hard}}, w_{\mathrm{len}}, w_*, w_{\mathrm{ent}}) = (0.3, 0.5, 1.0, 0.1, 0)$. For English 5/8/3, the corresponding values are $\rho = 0.35$, $\ell^* = 3.2$, and $(0.5, 0.5, 0.25, 0.08, 0)$. Both profiles use $L_{\min}=1$, $L_{\max}=8$, boundary offset $\beta = 0.18$, probability temperature $\tau = 1.0$, Gaussian

Setting	λ_{syn}	λ_{bnd}	λ_{seg}	Structure / schedule
English 2/4/2	0	$0.08 \rightarrow 0.06$	0.03	semantic target; boundary decay
English 5/8/3	0.08	$0.16 \rightarrow 0.12$	0.02	syntax input/loss; phrase target; boundary decay
MULTILINGUAL	0	0	0.02	auxiliary term decays at 30–60%
Chinese 2/4/2	0	$0.08 \rightarrow 0.06$	0.03	LTP word overwrite; composite fallback; stage decay

Table A3: Auxiliary objectives and schedules. All reported runs use causal language modeling, no masked-objective sampling, and logit regularization $10^{-4}\mathcal{L}_z$.

assignment width $\sigma_A = 0.7$, and the hard mass decoder in training and inference. A sampling temperature of 0.5 is configured but inactive under mass decoding. The same summaries record $\lambda_{\text{syn}} = 0.08$ for both audited English 5/8/3 checkpoints; all other rows use zero. The English schedule decays only the outer boundary weight.

Learning rates use 3% linear warmup followed by cosine decay and a final linear cooldown. Public-final English 5/8/3 uses a 0.1 floor and 1.6% cooldown; English matched BabyConceptLM/TA1 and both multilingual recipes use a 0.03 floor and 3% cooldown. The 2/4/2 μP groups use base width 768: token and unclassified matrix parameters retain 4×10^{-5} and Adam $\epsilon = 10^{-8}$, while width-896 concept matrices use 3.43×10^{-5} and 8.57×10^{-9}; biases, norms, embeddings, and the tied head are assigned to the unscaled "other" group.

Exposure is counted from original whitespace-delimited words before tokenization; multilingual samples additionally store the language and Byte-Premium-adjusted count. Counts are summed across microbatches and devices. The first state meeting each official target is saved at every 1M words through 10M, every 10M through 100M, and every 100M through 1B. The multilingual terminal checkpoint differs from the nominal 1B adjusted-word target only by floating-point round-off and supplies the 1B evaluation revision. Each checkpoint records target and actual exposure, overshoot, tokenizer, model configuration, and loading code; optimizer state is retained separately for resumption.

B Corpus construction

The English corpus contains 100M whitespace-delimited words in 11,601,896 documents. The multilingual mixture uses official English, Dutch, and Chinese streams with Byte Premiums 1.0, 1.0516, and 0.9894, respectively. One pass contains 43.095M, 40M, and 15M raw words, or exactly 100M adjusted words. Tokenization and packing are language-specific after training the shared tokenizer, so no sequence crosses a language boundary.

For the Chinese self-compiled corpus, only the `answer` field of BaikeQA (`baike2018qa`) (Xu, 2019) is retained. Answers are split at newlines; paragraphs shorter than 32 or longer than 4,096 Chinese characters are removed, and only candidates with more than 32 Jieba tokens are retained. Filtering stops at 101,603,104 Jieba tokens, split contiguously into 81,282,497 training and 20,320,607 validation tokens. The tokenizer is trained on the training portion. Offline N-LTP processing (Che et al., 2021) produces word-boundary sidecars and token-aligned Universal POS (UPOS) sidecars; adjacent UPOS changes are derived inside the model when constructing the composite fallback boundary target. At aligned positions, the word-start label overwrites that target, while invalid or unaligned positions retain the composite fallback. The retained environment and cache retrospectively identify PyPI `ltp==4.2.14` and `LTP/small@dea87bf0da3c5d054bbd426a78d7fa9efd347799`; the original metadata stored `LTP/small` but did not explicitly pin its revision. These sidecars are training labels only: no LTP parameters, hidden states, or embeddings initialize or enter BabyConceptLM, and LTP is not loaded for evaluation. The suffix *bpe40* denotes 40 training epochs, not vocabulary size.

C Matched-control provenance

Table A8 identifies all matched concept and TA1 checkpoints by public repository, immutable commit, and weight hash. English matched BabyConceptLM and Multilingual matched BabyConceptLM are the references used for the paired comparisons; they are

Models	Corpus / exposure	Length / global batch	Token / concept LR
English 2/4/2 public pair	official 100M / 1B words	256 / 128 → ∼512	4×10^{-5} / 3.43×10^{-5}
English 5/8/3 public pair	official 100M / 1B words	128 → 256 at 80% / 128 → 64	1.5×10^{-4} / 1.2×10^{-4}
English matched BabyConceptLM / TA1	archived 100M / 1B words	128 → 256 at 60% / 128 → 64	1.2×10^{-4} / 1.0×10^{-4}
babyconceptLM-multi public final	43.095/40/15M raw / 1B adjusted	128 → 256 at 60% / 128 → 64	1.0×10^{-4} / 8.0×10^{-5}
Multilingual matched BabyConceptLM / TA1	adjusted thirds / 1B adjusted	128 → 256 at 60% / 128 → 64	1.0×10^{-4} / 8.0×10^{-5}
babyconceptLM-baike102M_bpe40	81.283M training tokens / 40 epochs	256 / 128 → ∼512	4×10^{-5} / 3.43×10^{-5}

Table A4: Optimization settings. All runs use AdamW, gradient clipping at 1.0, mixed precision, weight decay 0.1, $(\beta_1, \beta_2) = (0.9, 0.98)$, and seed 42. The 2/4/2 concept rate is the implemented $768/896$ μP scaling of the nominal 4×10^{-5} rate.

Factor	2/4/2	5/8/3	Interpretive consequence
Parameters / blocks	105.6M / 8	194.6M / 16	Capacity and depth are not controlled
Early length / global batch	256 / ∼128	128 / 128	5/8/3 receives about twice as many updates per token
Nominal token / concept LR	$4.0/3.43 \times 10^{-5}$	$1.5/1.2 \times 10^{-4}$	Optimization scale differs
Early scaled token / concept LR	$1.0/0.857 \times 10^{-5}$	$15/12 \times 10^{-5}$	Approximately 15/14-fold early rate difference
Structural signal	Semantic target only	Syntax input/loss + phrase target	Additional structural targets could affect BLiMP
Output fusion	Bounded residual	Readout only	The prediction path also changes

Table A5: Cross-profile factors that co-vary with the exposure-indexed BLiMP difference. The early scaled 2/4/2 rates follow its linear token-batch scaling at the initial 128-sequence global batch; both recipes use 3% warmup. The table enumerates co-varying factors and plausible contributors; estimating individual effects requires matched interventions.

distinct from the final leaderboard artifacts `babyconceptLM-583-en@cb0668f69a87` and `babyconceptLM-multi@12d023b28c30`. Matched analyses use the pinned reference checkpoints; public-final artifacts are reported separately because their preprocessing, tokenizer, exposure, or schedule differs. The Chinese pair is likewise pinned to the public repositories in Table A8. All multilingual values in Table A10 use the task definitions and fixed-denominator collation pinned at `babylm-eval@6f825c2`.

Checkpoint interfaces and score verification. The five BabyLM submissions use Transformers-compatible model/tokenizer interfaces, support autoregressive string scoring, and retain the 28 exposure checkpoints described in Appendix A. Their English and MULTILINGUAL training uses no LTP model or other pretrained tagger/parser; LTP sidecars belong only to the separate Chinese case study (Appendix B). At manuscript freeze, the BabyLM leaderboard verified the reported scores, including server-scored MECO and hidden Hanzi. Chinese case-study scores come from frozen local outputs of `chinese-babylm-pipeline-final@f4437e1`; the currently accessible ChineseBabyLM 2026 final leaderboard displays matching values for babyconceptLM-baike102M_bpe40.

D Evaluation inventory and inference

STRICT zero-shot evaluation covers BLiMP (Warstadt et al., 2020), the BLiMP Supplement, EWoK (Ivanova et al., 2025), Entity Tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), Reading, and the challenge-defined GlobalPIQA variant; finetuning covers BoolQ, MultiRC, RTE, WSC, MRPC, QQP, and MNLI drawn from GLUE and SuperGLUE (Wang et al., 2019b,a). Reading averages normalized incremental-R^2 scores from four eye-tracking outcomes and spillover-controlled self-paced reading. The frozen implementation independently to-

Pair	$T/\bar{C}$	Ideal GF C/T	Padded GF C/T	History PF C/T	History saving	Padding evidence
English	4.539	72.14/102.07	75.59/102.07	509.27/631.39	19.3%	2,687 seq.; 200× random $B = 16$
MULTILINGUAL	7.983	61.59/102.07	61.92/102.07	348.10/565.18	38.4%	156,303 seq.; 200× random $B = 16$
Chinese	2.946	47.07/56.77	59.91/56.77	636.00/816.82	22.1%*	89,909 seq.; random $B = 32$

Table A6: Forward FLOPs audit. C/T denote concept/token. For the final-checkpoint columns, $\bar{C}$ is the deterministic concept count averaged over every packed dev sequence, and $T/\bar{C}$ uses $T = 256$; this ratio is not a training-history mean or the mean of per-segment lengths. Ideal GF evaluates the analytic formula at $\bar{C}$, whereas padded GF evaluates it at within-batch maximum concept counts. The Chinese full-dev values are $\bar{C} = 86.883$, $T/\bar{C} = 2.946$, and 47.07 ideal GF. History values separately integrate every logged training interval using its sequence length, effective batch size, and logged mean segment length. A rough forward-plus-backward estimate is three times the history column. The asterisk marks the Chinese history estimate as optimistic: current tensors pad concept sequences to the within-batch maximum, and historical logs do not retain that distribution. Padded final values use the evidence in the last column.

Frozen pair	Dev seq.	Concepts/seq.	Batch max	P5/P50/P95	Saving / repeat 95% interval
English matched BabyConceptLM–TA1	2,687	56.39 ± 4.44	64.32 ± 2.53	61/64/69	25.94% / 25.88–26.02%
Multilingual matched BabyConceptLM–TA1	156,303	32.07 ± 0.35	32.84 ± 0.95	32/32/34	39.33% / 39.329–39.336%
Chinese BabyConceptLM–TA1	89,909	86.88 ± 33.92	143.64 ± 6.17	134/143/154	-5.52% / $[-5.554, -5.469]\%$

Table A7: Full-dev padding distributions for all matched pairs. Each of 200 repetitions randomly partitions every dev sequence without replacement into groups of $B = 16$ for English/MULTILINGUAL and $B = 32$ for Chinese; incomplete final groups are retained. Batch-max means, SDs, and P5/P50/P95 pool allocated batches. Saving is signed relative to TA1, so the negative Chinese value denotes additional concept-model FLOPs. "Repeat 95% interval" is the central interval of the 200 repetition-level mean savings, not a population confidence interval.

kenizes the target word; our post-hoc audit found a continuation mismatch on 1,726/1,726 items and a different piece count on 462. Fast evaluations are applied to the required exposure checkpoints. AoA fits acquisition points from the full checkpoint sequence, computes Pearson r with CDI-derived child AoA, and retains r only when $p \leq .1$.

MULTILINGUAL evaluation covers language-specific zero-shot suites, the challenge-defined GlobalPIQA variant, MECO reading-time prediction (Siegelman et al., 2022), finetuned classification, and cross-lingual part-of-speech tagging; Chinese additionally includes Hanzi structure and pinyin. MultiBLiMP and XCOMPS follow their released benchmarks (Jumelet et al., 2025; He et al., 2025); HellaSwag, Winogrande, and XStoryCloze use the MuBench versions (Han et al., 2025). MECO and hidden Hanzi predictions use server-side scoring. Task definitions, language subsets, preprocessing, and aggregation are fixed by `babylm-eval@6f825c2`; this pinned inventory is the source for challenge-specific variants without a separate archival citation.

ChineseBabyLM contains 14 tasks in three tracks. NLU comprises ZhoBLiMP, XCOMPS-ZH, AFQMC, OCNLI, TNEWS, CLUEWSC2020, C3, and diagnostic NLI. The Hanzi track contains open and hidden structure and pinyin tests. The cognitive track uses responses from 11 participants reading 672 isolated words and 12 participants listening to 60 stories comprising 52,269 words. The official pipeline fits encoding models from frozen representations to fMRI responses and reports correlation-based scores.

D.1 Participant-level matched contrasts

Pinned encoding implementation. At `f4437e1`, discourse features average the selected hidden state's subtokens within each word, apply an HRF convolution, and downsample to a 0.71-second TR with a 19-TR offset. Word-fMRI instead mean-pools non-padding positions for each isolated word. The default `layer_index=-1` selects the final exported state (`last_hidden_state` for Word-fMRI). Discourse ridge regression scores each voxel by the temporal mean of the product of sample-standardized predicted and observed responses. The report takes the NaN-aware voxel mean within each participant/ROI; the nested-CV branch first averages voxel scores over five outer folds. Word-fMRI uses Pearson correlation across valid voxels for each trial and averages the highest

Pair	Matched concept repository@commit (SHA-256)	DWA	TA1 repository@commit (SHA-256)
English	babyconceptLM-English-matched@e82add172267 SHA 55ba4c4cb71d	T/C/R; 8/8/8	babyconceptLM-TA1-token-only-en@49c3d3ac3f1d SHA 4fdac3f1f3e4
MULTI	babyconceptLM-Multilingual-matched@ea60a3df6a34; SHA 82181ab8371a	T/C/R; 6/4/4	babyconceptLM-TA1-token-only-multi@0e0296602741 SHA 49da0bb34d85
Chinese	babyconceptLM-baike102M_bpe40@05fda2ae8658; SHA 6345b814d5d2	none	babyconceptLM-TA1-token-only-zh@8bbf037e56a7 SHA 512c6bbef0dc

Table A8: Frozen matched-control identities. Repository names are relative to the zondaxyz/ Hugging Face namespace; commits are immutable revisions and values marked SHA are weight-hash prefixes. T/C/R denotes token/concept/readout paths; numbers are DWA current-state initialization logits.

Recipe field	Matched concept reference	Public-final concept model
English: A-stable reference versus STRICT final (both 5/8/3)		
Corpus/tokenizer	Archived STRICT processing; tokenizer_100M.json	Reprocessed official STRICT; tokenizer_babylm2026_strict.json
Packed train sequences	1,251,114	1,300,682
Training steps	97,750	101,620
Length switch	60% of steps	80% of steps
Token/concept learning rate	$1.2/1.0 \times 10^{-4}$	$1.5/1.2 \times 10^{-4}$
DWA initialization (T/C/R)	8/8/8	4/4/4
Cooldown / LR floor ratio	0.03 / 0.03	0.016 / 0.10
MULTILINGUAL: archived reference versus public final (both 5/8/3)		
Corpus mixture	Approximately equal adjusted thirds	43.095M/40M/15M raw ENG/NLD/ZHO words
Tokenizer directory	tokenizers-multi/	tokenizers-multi-babylm2026/
Packed train sequences	1,119,914	2,019,494
Training steps	87,500	157,780

Table A9: Archived matched versus public-final training recipes. TA1 uses the matched recipe in the middle column. MULTILINGUAL retains DWA initialization 6/4/4, token/concept learning rates $1.0/0.8 \times 10^{-4}$, a 60% length switch, and cooldown/floor ratios 0.03/0.03 in both recipes. Table A8 identifies the matched checkpoints.

10% of trial correlations, averaging these scores across outer folds when using nested CV. The two tasks therefore use different correlation and aggregation procedures. The scorer supports explicit train/dev/test splits or nested CV; without a test directory, the split branch reports dev performance.

We pair identical participant IDs in the frozen BabyConceptLM–TA1 outputs. For each discourse domain, the analysis uses one reported domain score per participant and model; Word-fMRI uses one word-task score per participant and model. Let $s^C_{i,d}$ and $s^T_{i,d}$ denote the reported correlations, multiplied by 100, for participant i and domain d. The paired difference is $\Delta_{i,d} = s^C_{i,d} - s^T_{i,d}$. For the Cognition–Language and six-domain summary contrasts, we first average the relevant domain scores within participant, then compare models. Participants, not domains or voxels, are the resampling units.

The archived analysis uses 200,000 paired participant-bootstrap samples for each unadjusted 95% interval, preserving each participant's two model scores together. Exact two-sided sign-flip tests enumerate all 2^n assignments ($4,096$ for discourse; $2,048$ for Word-fMRI). With $T_{\text{obs}} = |n^{-1}\sum_i \Delta_i|$, the exact p-value is the fraction of sign vectors $\boldsymbol{\epsilon} \in \{-1,+1\}^n$ for which $|n^{-1}\sum_i \epsilon_i\Delta_i| \geq T_{\text{obs}}$. The paired standardized effect is

$$d_z = \frac{\overline{\Delta}}{s_\Delta}, \qquad s_\Delta = \sqrt{\frac{\sum_i(\Delta_i - \overline{\Delta})^2}{n-1}}. \quad (16)$$

Holm adjustment is applied separately to the two focal Cognition/Language tests (Table 2) and to all six discourse-domain tests (Table A19). The within-participant averages and Word-fMRI row report unadjusted exact p-values. Cognition and Language are focal exploratory contrasts; this designation makes no claim of preregistration. All six domains and their six-test correction remain re-

ported.

These intervals characterize participant sampling uncertainty conditional on the two fixed seed-42 models and the retained evaluation outputs. They do not include training-seed variance, story sampling, or uncertainty across analysis choices. The records available for this report do not specify the original bootstrap RNG seed, interval construction, or matched-run layer/split settings. The six-participant descriptive sweeps below are a separate analysis.

The detailed inventory is organized as follows. Table A10 reports every shared task entering the matched-control aggregates. Tables A11–A12 decompose English STRICT results; Tables A13–A15 provide the complete MULTILINGUAL outputs; Figure A2 and Table A16 report the exposure trajectories and anchors; and Tables A17–A19 together with Figures A3–A5 document the complete ChineseBabyLM results and descriptive layer-wise profiles.

E Segmentation protocol

The audit covers the four English STRICT models. Its frozen artifact uses the `story` field of Hippocorpus V3 (Sap et al., 2020), a collection independent of the BabyLM training corpus containing recalled, imagined, and retold narratives. We decode HTML entities, apply NFKC normalization, remove control characters, collapse whitespace, and preserve case and punctuation. Requiring 100–800 whitespace-delimited words yields 6,854 stories, 1,799,978 words, and 2,135,141 BPE tokens under the STRICT submission tokenizer. Exact text round trips are verified for every document, and token-aligned whitespace-word and word-start sidecars are stored with the artifact.

For each frozen checkpoint, the analyzer selects up to 2,000 documents by reservoir sampling with seed 42 and extracts one 256-token window per document. It micro-averages tokenizer tokens and source words per concept. Learned boundary rate excludes window starts and cap-attributed cuts. The complete length distribution includes P50, P90, standard deviation, coefficient of variation, Shannon entropy, and $P(L = k)$ for $k = 1, \ldots, 8$; sampled-window edge segments are excluded. Word-start alignment includes boundary-probability AUC and the hard-boundary-rate difference between word starts and within-word positions. Bounded-gate statistics apply only to the 2/4/2 profile. Confidence intervals use 1,000 document-level bootstrap resamples. Matched noDWA/DWA comparisons use identical windows and resample aligned document pairs to form intervals over differences. All values are computed offline from frozen checkpoints rather than reconstructed from batch-level training summaries; each artifact retains document-level metrics plus model, configuration, and tokenizer fingerprints for audit.

Illustrative 2/4/2 cuts. For one shared Hippocorpus window, noDWA produces "the show, near the top | of the theater, and laid | down a blanket. The | opener came on, and," whereas DWA produces "the show, near the | top of the theater, and | laid down a blanket. | The opener came on." These examples illustrate realized length variability; corpus-level interpretation relies on the aggregate audit above.

Evaluation	Shared family/task	Concept	TA1	Δ (C−T)
English	BLiMP	75.90	77.37	−1.47
English	BLiMP Supplement	60.28	65.27	−4.99
English	EWoK	55.62	54.89	+0.73
English	Entity Tracking	16.19	16.47	−0.28
English	COMPS	55.72	57.04	−1.32
MULTI–EN	BLiMP	69.06	70.05	−0.99
MULTI–EN	HellaSwag	26.53	26.51	+0.01
MULTI–EN	MultiBLiMP	84.16	85.71	−1.56
MULTI–EN	Winogrande	51.61	49.30	+2.31
MULTI–EN	XStoryCloze	51.78	51.70	+0.08
MULTI–EN	GlobalPIQA	38.14	41.09	−2.96
MULTI–NL	BLiMP-NL	67.26	70.63	−3.37
MULTI–NL	HellaSwag	26.04	26.13	−0.09
MULTI–NL	MultiBLiMP	76.15	71.86	+4.29
MULTI–NL	Winogrande	51.11	49.71	+1.40
MULTI–NL	XCOMPS	51.16	53.50	−2.35
MULTI–NL	XStoryCloze	47.60	46.98	+0.62
MULTI–NL	GlobalPIQA	37.12	37.14	−0.01
MULTI–ZH	HellaSwag	27.17	26.85	+0.32
MULTI–ZH	Winogrande	48.39	49.79	−1.40
MULTI–ZH	XCOMPS	51.71	51.70	+0.01
MULTI–ZH	XStoryCloze	49.38	48.61	+0.77
MULTI–ZH	ZhoBLiMP	66.60	71.93	−5.33
MULTI–ZH	GlobalPIQA	37.65	40.06	−2.41

Table A10: Every task entering the English and multilingual matched-control macros. English compares English matched BabyConceptLM with English TA1; GlobalPIQA is excluded because it was not shared by both frozen outputs. MULTILINGUAL compares Multilingual matched BabyConceptLM with multilingual TA1 and follows the official fixed-denominator collator, including equal weighting of GlobalPIQA parallel and nonparallel subsets. Scores are percentages; minor delta discrepancies reflect calculation before display rounding. Chinese task-level values appear in Table A18.

Model	BLiMP	Supp.	EWoK	Entity	COMPS	GPIQA	SGLUE	Reading	AoA
GPT-2 Strict	74.73	65.00	54.37	16.91	55.85	36.62	67.75	6.93	−11.58
Strict-Interaction	72.89	65.09	54.23	15.89	55.37	36.18	67.28	3.43	−11.36
babyconceptLM-en-noDWA (2/4/2)	62.35	56.63	49.51	17.67	51.18	35.67	64.45	0.18	10.36
babyconceptLM-en-DWA (2/4/2)	62.44	56.41	49.83	17.54	51.10	34.17	63.72	0.18	0.00
babyconceptLM-en-noDWA (5/8/3)	75.82	63.22	53.96	15.07	55.61	39.17	67.45	0.22	−13.79
babyconceptLM-en-DWA (5/8/3)	76.58	63.52	53.30	14.79	55.88	37.59	67.10	0.13	−10.71

Table A11: English STRICT benchmark-family scores. GPIQA and SGLUE abbreviate GlobalPIQA and (Super)GLUE. Reading and AoA are official pipeline outputs retained with the tokenizer and threshold caveats in the text.

Model	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg.
GPT-2 Strict	69.66	60.76	85.34	65.92	71.56	57.55	63.46	67.75
Strict-Interaction	67.46	60.68	86.27	66.21	70.51	58.27	61.54	67.28
babyconceptLM-en-noDWA (2/4/2)	68.07	48.72	82.99	61.34	63.02	59.71	67.31	64.45
babyconceptLM-en-DWA (2/4/2)	68.38	48.35	82.49	61.01	61.67	56.83	67.31	63.72
babyconceptLM-en-noDWA (5/8/3)	70.76	57.99	85.90	64.81	70.95	58.27	63.46	67.45
babyconceptLM-en-DWA (5/8/3)	70.28	57.86	83.87	65.80	71.56	56.83	63.46	67.10

Table A12: English (Super)GLUE results. Accuracy is reported except MRPC and QQP, which use F1. All BabyConceptLM entries contain the complete seven-task suite.

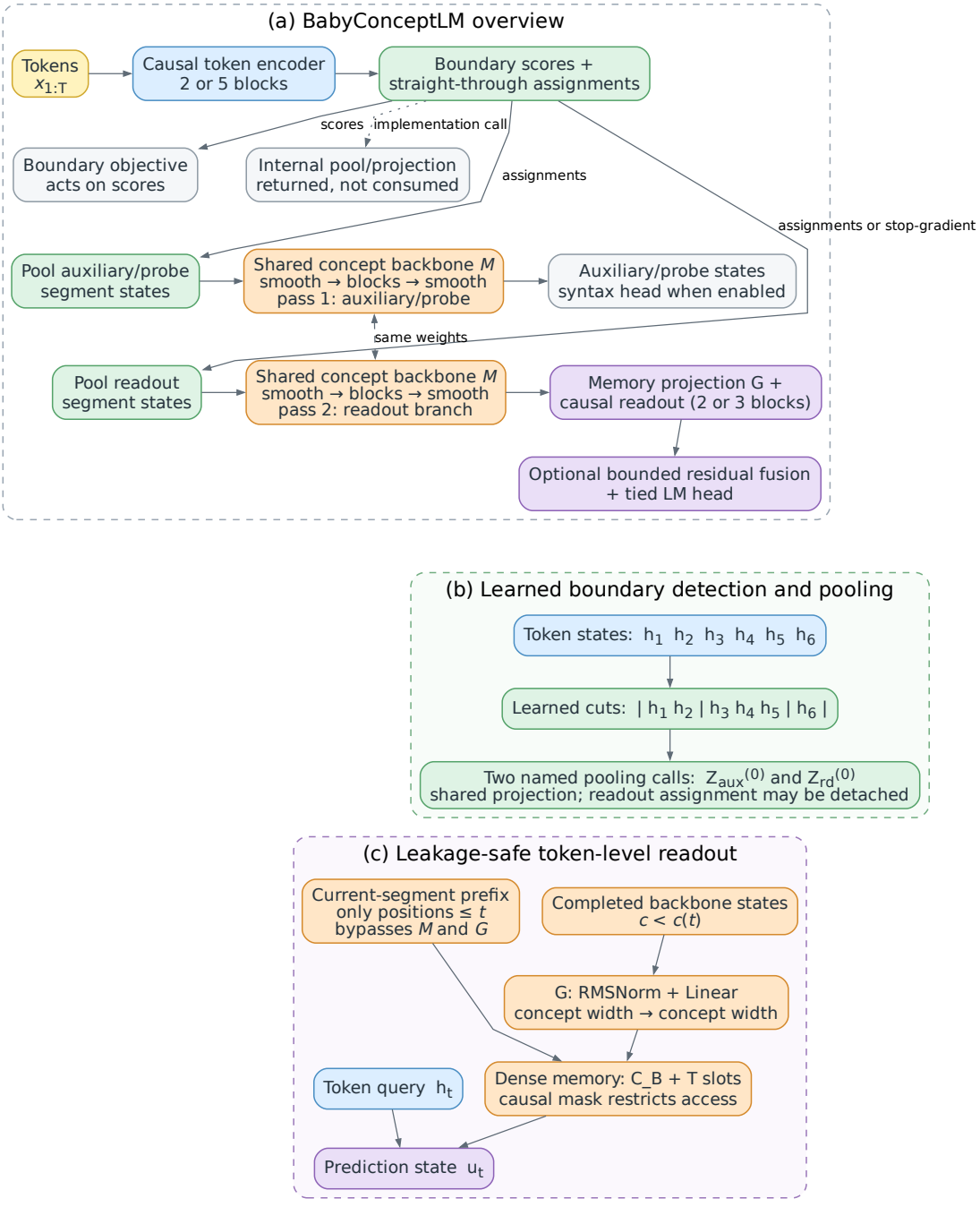


Figure A1: Detailed BabyConceptLM computation graph. Panel (a) separates boundary supervision from the two shared-backbone executions, each with pre- and post-stack causal smoothing, and shows the unused internal pooled tensors. Panels (b) and (c) expand pooling and readout, including completed-memory projection G and the current-prefix bypass.

Language	Zero-shot task	GPT-2 tri.	babyconceptLM-multi
English	BLiMP	70.49	71.24
English	HellaSwag	26.51	26.35
English	MultiBLiMP	85.97	86.88
English	Winogrande	50.12	49.30
English	XStoryCloze	49.54	50.08
English	GlobalPIQA	39.12	38.64
Dutch	BLiMP-NL	77.04	66.34
Dutch	HellaSwag	26.65	26.17
Dutch	MultiBLiMP	92.11	75.42
Dutch	Winogrande	49.38	51.61
Dutch	XCOMPS	53.04	50.93
Dutch	XStoryCloze	48.99	48.07
Dutch	GlobalPIQA	38.11	36.61
Chinese	HellaSwag	26.74	26.96
Chinese	Winogrande	49.63	51.03
Chinese	XCOMPS	53.47	53.46
Chinese	XStoryCloze	47.91	51.63
Chinese	ZhoBLiMP	74.03	75.90
Chinese	GlobalPIQA	31.74	38.15

Table A13: Complete MULTILINGUAL zero-shot tasks used in the aggregate. GPT-2 values reproduce the rounded official README entries.

Language	Finetuning task	GPT-2 tri.	babyconceptLM-multi
English	ARC / Belebele / BMLAMA	24.79 / 26.14 / 12.00	21.88 / 17.61 / 11.67
English	MNLI / part-of-speech / SIB200	50.00 / 93.40 / 69.50	55.35 / 93.97 / 66.50
English	TruthfulQA / XNLI	23.21 / 45.65	22.32 / 52.85
Dutch	ARC / Belebele / BMLAMA	24.38 / 26.14 / 9.93	25.62 / 19.89 / 12.09
Dutch	INCLUDE / MNLI / part-of-speech	19.64 / 49.27 / 94.97	33.04 / 55.35 / 95.25
Dutch	SIB200 / TruthfulQA	71.00 / 23.21	72.00 / 23.21
Chinese	ARC / Belebele / BMLAMA	25.21 / 22.73 / 10.93	25.21 / 19.32 / 10.84
Chinese	INCLUDE / MNLI / part-of-speech	28.57 / 47.58 / 90.40	25.89 / 56.87 / 91.97
Chinese	SIB200 / TruthfulQA / XNLI	72.50 / 23.21 / 45.05	84.00 / 25.89 / 50.75

Table A14: Complete MULTILINGUAL finetuning tasks used in the aggregate. Slash-separated values follow the task order shown in each cell.

Non-aggregate output	GPT-2 tri.	babyconceptLM-multi
Hanzi structure	55.05	53.40
Hanzi pinyin	50.05	52.05
MECO English L1	5157.10	5206.26
MECO Dutch L1	388.27	173.85
MECO Chinese L1	632.02	627.38
MECO Dutch L2	55.92	71.43
MECO Chinese L2	75.47	166.91

Table A15: MULTILINGUAL Hanzi and MECO outputs verified against the public leaderboard at manuscript freeze. MECO delta-log-likelihood values are not normalized across reader groups; both output types are excluded from the aggregate.

Curve	1M	10M	100M	200M	500M	1B	Log-AUC
STRICT noDWA (2/4/2)	32.64	33.93	35.15	36.65	37.02	37.41	35.06
STRICT DWA (2/4/2)	32.63	33.91	35.64	36.59	37.45	37.35	35.04
STRICT noDWA (5/8/3)	34.73	34.80	39.71	40.17	41.51	41.18	37.72
STRICT DWA (5/8/3)	34.99	35.16	39.13	40.16	40.84	41.14	37.46
MULTILINGUAL mean	44.67	45.04	48.38	49.64	50.68	51.33	47.35

Table A16: Learning-curve anchors. Log-AUC is the trapezoidal mean over $\log_{10}$ exposure from 1M to 1B. The corresponding public Hugging Face repositories expose all 28 evaluated checkpoint revisions.

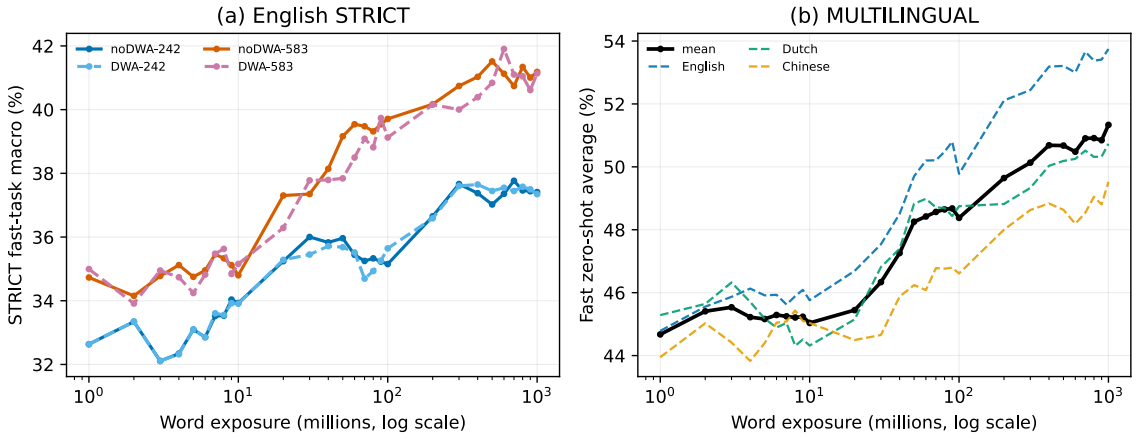


Figure A2: Fast evaluation at all 28 official exposure checkpoints. Left: within-profile English DWA/noDWA pairs. Right: the multilingual mean and language decomposition. Lines connect observed checkpoints and do not imply monotonic learning. Anchor values and log-exposure AUCs appear in Table A16.

Model	ZhoBLiMP	XCOMPS-ZH	Hanzi struct.	Hanzi pinyin	Word fMRI	Discourse fMRI
T5 (14M)	75.82	56.56	51.10	51.40	55.01	8.24
BERT (14M)	75.82	62.66	55.80	45.85	56.02	10.74
Mamba (14M)	74.23	58.88	52.05	**52.90**	47.18	9.37
Pythia (14M)	73.95	57.02	53.70	50.10	49.89	8.67
Token-level GPT-BERT	**76.88**	58.98	54.60	50.40	56.45	9.92
Matched TA1	72.25	**63.29**	**56.70**	52.85	**56.77**	10.74
BabyConceptLM	70.08	61.82	56.45	52.15	56.61	**10.92**

Table A17: Selected ChineseBabyLM results. Official-baseline values reproduce the aligned columns of Table 4 in Song et al. (2026); token-level GPT-BERT is an unmatched recipe. The displayed BabyConceptLM values match the accessible ChineseBabyLM 2026 final leaderboard. Matched TA1 shares the BabyConceptLM data, tokenizer, exposure, optimizer/schedule, batches, seed, and absence of DWA, with a 1.7% parameter difference. The contrast evaluates the complete concept pathway, jointly varying segmentation, concept processing, readout, and layer allocation. Bold marks the largest displayed value.

Model	ZhoB	XComp	H-Str.	H-Pin.	H-Str.-H	H-Pin.-H	Word	Disc.
Matched TA1	72.25	63.29	56.70	52.85	55.95	50.25	56.77	10.74
babyconceptLM-baike102M_bpe40	70.08	61.82	56.45	52.15	55.55	50.30	56.61	10.92

Model	AFQMC	OCNLI	TNEWS	WSC	C3	Diag. NLI
babyconceptLM-baike102M_bpe40	69.14	60.92	53.56	63.49	30.19	45.10

Table A18: Completed non-finetuning ChineseBabyLM inventory for the matched pair, followed by BabyConceptLM finetuning tasks. H and Disc. denote hidden and discourse; correlations are multiplied by 100. TA1 finetuning was not run.

Discourse domain	n	Δ (C−T)	95% CI	d_z	Exact p	Holm-6 p
Cognition	12	+0.288	[−0.274, 0.797]	0.29	.324	.871
Language	12	+0.281	[−0.118, 0.687]	0.38	.218	.871
Manipulation	12	+0.406	[−0.024, 0.789]	0.54	.088	.442
Memory	12	−0.073	[−0.545, 0.311]	−0.09	.802	.871
Reward	12	−0.153	[−0.257, −0.029]	−0.72	.032	.193
Vision	12	+0.319	[−0.136, 0.850]	0.35	.270	.871

Table A19: Participant-level matched TA1 contrasts for every CogBench discourse domain. Correlations are multiplied by 100 and positive differences favor BabyConceptLM. Intervals are unadjusted paired bootstrap intervals; exact two-sided sign-flip tests use all 2^{12} assignments, and Holm-6 corrects the six exploratory domain tests. The focal exploratory two-domain correction for Cognition and Language is reported in Table 2.

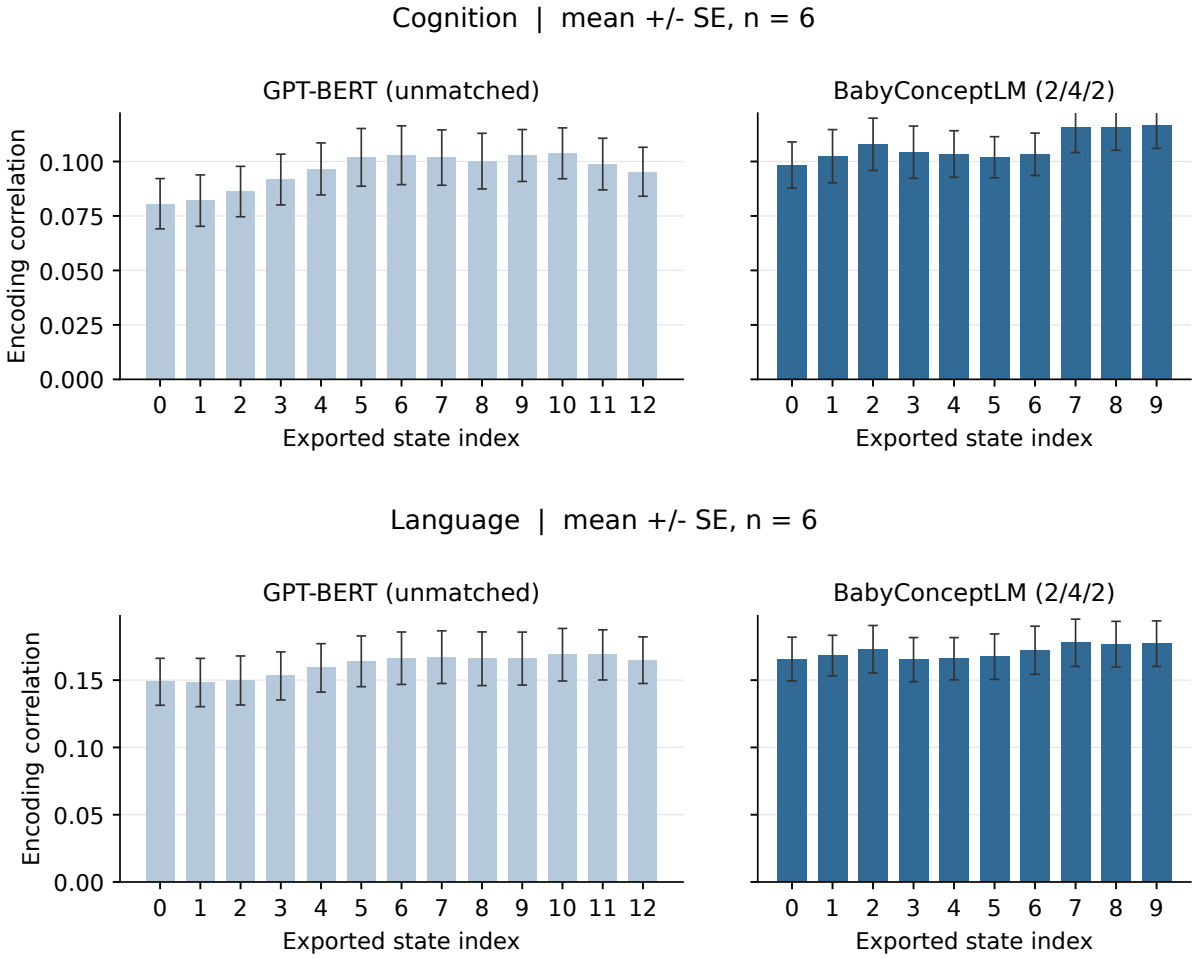


Figure A3: Descriptive discourse-fMRI layer sweeps in Cognition (top) and Language (bottom). Each bar averages the six participant-domain correlations in the retained layer-sweep export; error bars are ± 1 SE (sample SD/$\sqrt{6}$), not confidence intervals. Scores are averaged across participants within each model, state, and domain, without pooling other domains or layers. These unmatched GPT-BERT–BabyConceptLM sweeps are separate from the matched TA1 $n = 12$ tests in Table 2. Index 0 is the embedding output; subsequent indices enumerate within-model exported states, not homologous depths. GPT-BERT has token-level Transformer blocks, not concept/readout stages.

English model	**Tok./con.**	**Word/con.**	**Learned bnd.**	**P50/P90**	$P(L = 6)$	$P(L = 8)$	**Word AUC**	**Gate**
babyconceptLM-en-noDWA (2/4/2)	5.807	4.899	16.74	6/6	70.16	0.57	0.543	0.0003
babyconceptLM-en-DWA (2/4/2)	5.706	4.813	17.09	6/6	63.32	0.43	0.546	0.0003
babyconceptLM-en-noDWA (5/8/3)	5.911	4.987	6.93	8/8	3.61	57.02	0.448	n/a
babyconceptLM-en-DWA (5/8/3)	6.405	5.403	4.64	8/8	2.48	68.44	0.448	n/a

Table A20: Hippocorpus segmentation statistics under each model's native decoder. Boundary and length probabilities are percentages; P50/P90 exclude window-edge-censored segments. Complete histograms and dispersion statistics appear in Table A21 and Figure A6.

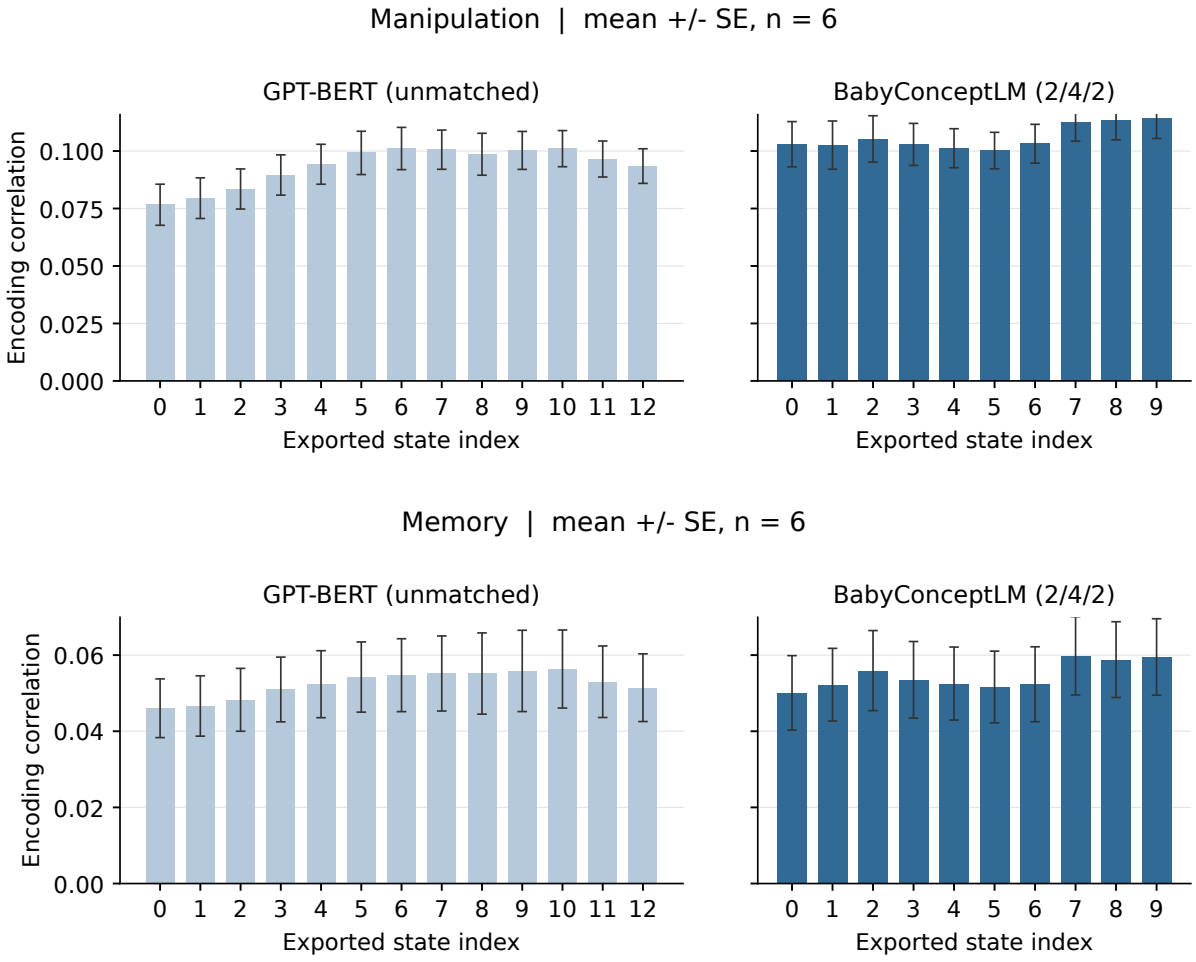


Figure A4: Descriptive discourse-fMRI layer sweeps for unmatched GPT-BERT and BabyConceptLM in Manipulation (top) and Memory (bottom). Bars average six participant-domain correlations separately for each model/state/domain; whiskers are ± 1 SE (sample SD/$\sqrt{6}$), with no pooling across domains or layers. Index 0 denotes embeddings; subsequent indices are within-model exported states and do not imply matched depth or shared concept/readout stages. These six-participant exports are distinct from the matched TA1 $n = 12$ analysis in Table 2.

Model	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	**SD**	**CV**	H
noDWA-242	0.17	0.16	0.15	0.12	22.54	70.16	6.13	0.57	0.604	0.104	1.189
DWA-242	0.15	0.15	0.16	0.15	30.95	63.32	4.68	0.43	0.617	0.108	1.239
noDWA-583	9.94	8.92	7.10	5.66	4.48	3.61	3.26	57.02	2.671	0.449	2.145
DWA-583	7.95	6.39	5.42	4.03	3.12	2.48	2.18	68.44	2.497	0.386	1.741

Table A21: Complete-segment length distributions. P_k is the percentage of uncensored segments with k tokenizer tokens; SD is in tokens, CV is SD divided by mean length, and entropy H is in bits. The 2/4/2 rows exclude a literal fixed-six decoder but show a narrow near-fixed-six regime.

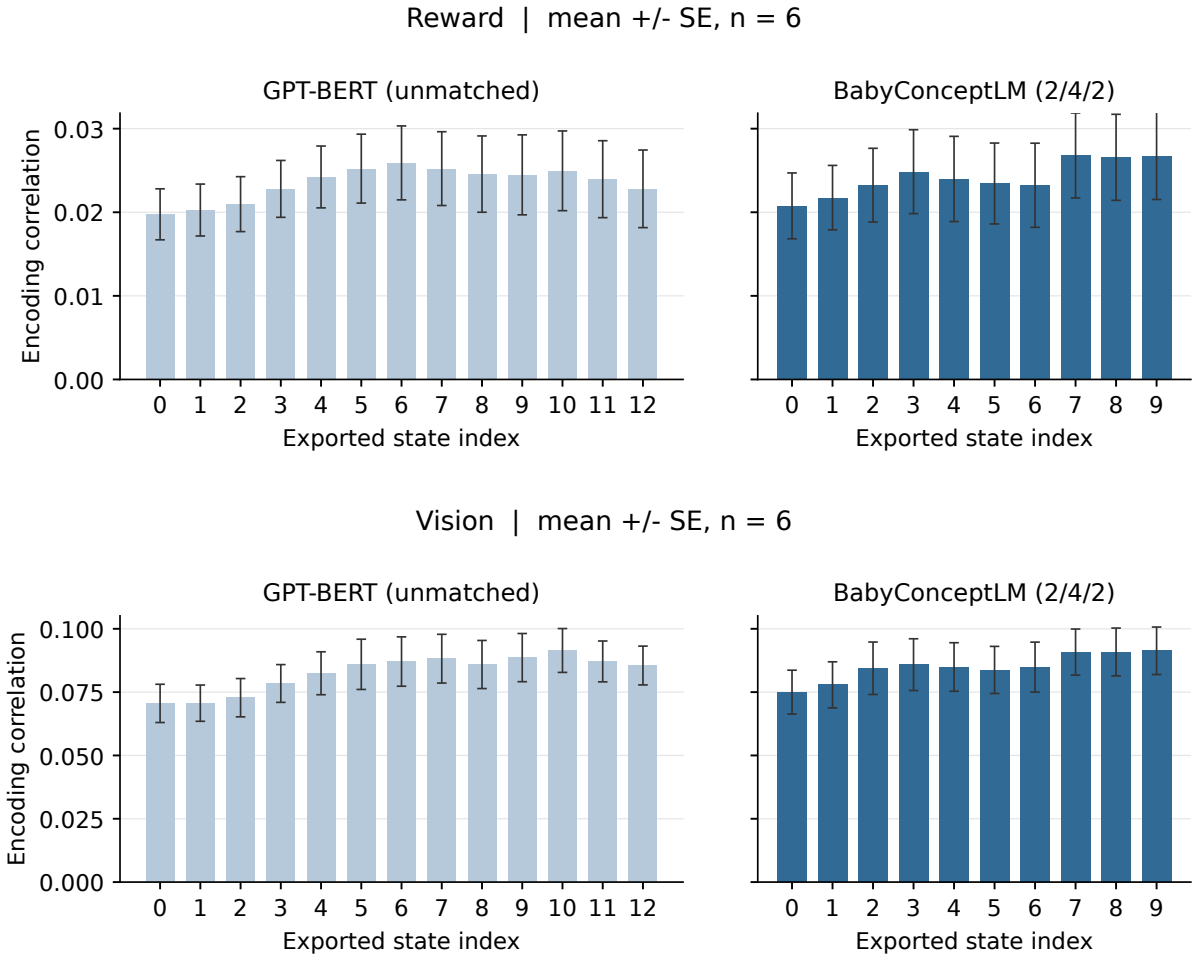


Figure A5: Descriptive discourse-fMRI layer sweeps for unmatched GPT-BERT and BabyConceptLM in Reward (top) and Vision (bottom). Bars average six participant-domain correlations separately for each model/state/domain; whiskers are ± 1 SE (sample SD/$\sqrt{6}$), with no pooling across domains or layers. Index 0 denotes embeddings; subsequent indices are within-model exported states and do not imply matched depth or shared concept/readout stages. These six-participant exports are distinct from the matched TA1 $n = 12$ analysis in Table 2.

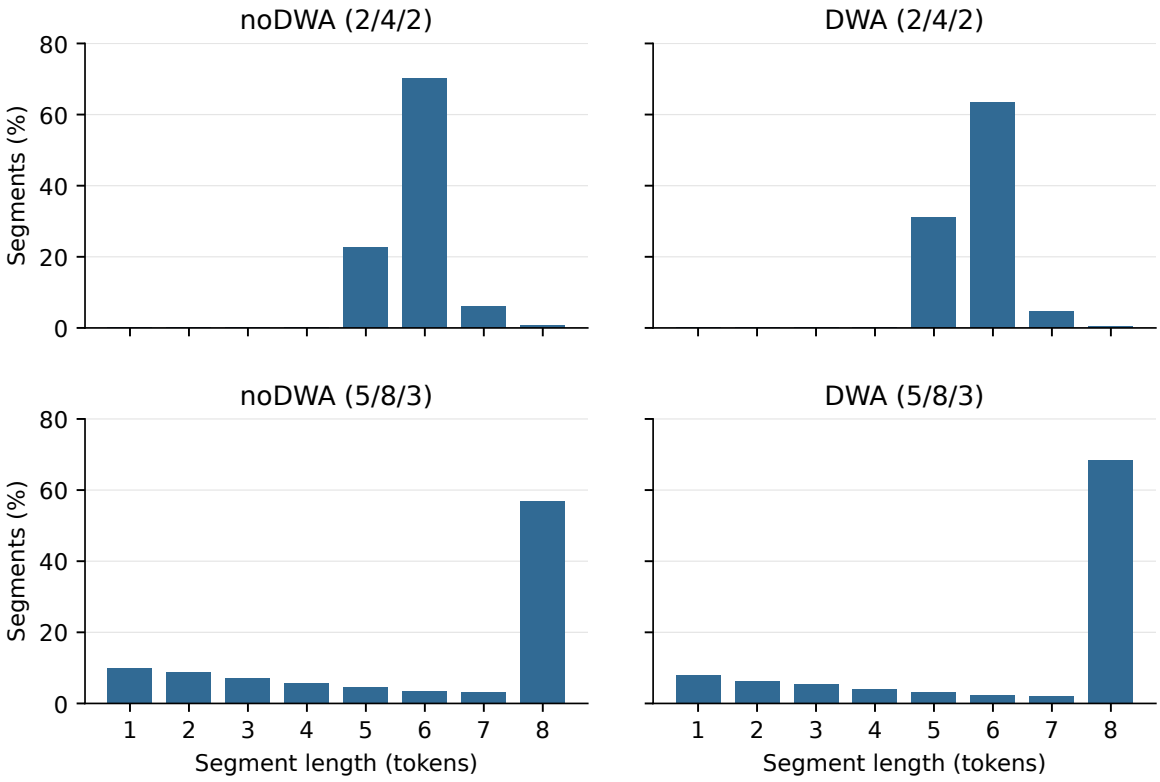


Figure A6: Uncensored Hippocorpus segment-length distributions for the four English models, with shared bins and percentage axes. Each panel divides the retained segment counts by its own total: 80,328 (noDWA-242), 81,801 (DWA-242), 78,870 (noDWA-583), and 72,604 (DWA-583). The shallow profile concentrates at lengths five and six; the deep profile concentrates at the eight-token cap while retaining a lower-length tail.

Model / contrast	Quantity	Estimate	95% CI	Resampling unit
noDWA-242	Tokens/concept	5.807	5.801–5.813	document
noDWA-242	Words/concept	4.899	4.887–4.911	document
noDWA-242	Learned boundary (%)	16.74	16.70–16.78	document
noDWA-242	$P(L = 8)$ (%)	0.57	0.43–0.73	document
noDWA-242	Gate mean	0.000295	0.000285–0.000307	document
noDWA-242	Word-start rate difference (pp)	0.26	0.00–0.53	document
DWA-242	Tokens/concept	5.706	5.700–5.712	document
DWA-242	Words/concept	4.813	4.802–4.825	document
DWA-242	Learned boundary (%)	17.09	17.04–17.12	document
DWA-242	$P(L = 8)$ (%)	0.43	0.30–0.59	document
DWA−noDWA (242)	Tokens/concept	-0.101	$[-0.105, -0.098]$	paired documents
DWA−noDWA (242)	Words/concept	-0.086	$[-0.088, -0.083]$	paired documents
DWA−noDWA (242)	Learned boundary (pp)	$+0.34$	0.33–0.35	paired documents
DWA−noDWA (242)	$P(L = 8)$ (pp)	-0.14	$[-0.17, -0.11]$	paired documents
noDWA-583	Tokens/concept	5.911	5.881–5.942	document
noDWA-583	Words/concept	4.987	4.956–5.017	document
noDWA-583	Learned boundary (%)	6.93	6.80–7.06	document
noDWA-583	$P(L = 8)$ (%)	57.02	56.46–57.59	document
noDWA-583	$P(L = 1)$ (%)	9.94	9.67–10.21	document
noDWA-583	Word-start rate difference (pp)	-1.23	$[-1.50, -0.95]$	document
DWA-583	Tokens/concept	6.405	6.376–6.432	document
DWA-583	Words/concept	5.403	5.376–5.430	document
DWA-583	Learned boundary (%)	4.64	4.54–4.74	document
DWA-583	$P(L = 8)$ (%)	68.44	67.92–68.91	document
DWA-583	Word-start rate difference (pp)	-0.82	$[-1.09, -0.55]$	document
DWA−noDWA (583)	Tokens/concept	$+0.493$	0.483–0.505	paired documents
DWA−noDWA (583)	Words/concept	$+0.416$	0.407–0.425	paired documents
DWA−noDWA (583)	Learned boundary (pp)	-2.29	$[-2.35, -2.24]$	paired documents
DWA−noDWA (583)	$P(L = 8)$ (pp)	$+11.42$	11.19–11.64	paired documents
DWA−noDWA (583)	Word-start rate difference (pp)	$+0.41$	0.19–0.62	paired documents

Table A22: Document-bootstrap intervals from the English segmentation artifacts. Paired rows use identical Hippocorpus windows for DWA and noDWA; pp denotes percentage points. Exported P50/P90 intervals collapse to their point values, while word-start AUC has no exported interval; these point estimates appear in Table A20.

Introducing Code-Switched Contexts to Cognitively-Inspired Bilingual Model Training

Zhuojing Huang **Luise Pohlmann** **Lisa Beinborn**
University of Göttingen, Germany
firstname.lastname@uni-goettingen.de

Abstract

During language acquisition, bilingual children are regularly exposed to code-switched input and use it as a cognitive scaffold to accelerate vocabulary growth and cross-linguistic syntactic mapping. In contrast, computational bilingual models are conventionally pretrained on interleaved monolingual corpora. While introducing synthetic code-switching during pretraining has become a promising strategy to enhance cross-lingual alignment and downstream performance, the structural and developmental parameters governing the success remain poorly understood. In this work, we investigate the efficiency of training with synthetic code-switched data across two typologically distinct language pairs by controlling two key variables: the structural location of code-switches and the dynamic switching rate across training stages. Our results show that training with code-switched data improves cross-lingual alignment for typologically close languages.[1]

1 Introduction

In many bilingual families, children are regularly exposed to code-switched input during language acquisition (Byers-Heinlein, 2013; Bail et al., 2015). Psycholinguistic findings suggest that child code-switching is not a result of confusion or failure (Meisel, 2019). Instead, it is seen as an attempt to communicate given limited linguistic resources (Paradis et al., 2000; Werker and Byers-Heinlein, 2008). By using code-switching as a scaffold for currently weaker language (Yow et al., 2018; Schneider et al., 2026), bilingual children actively draw on their linguistic resources from both languages to support lexical growth (Deuchar and Quay, 1994; Meisel, 2019; Gaskins et al., 2022) with the effect that they need fewer repetitions for novel word acquisition (Kaushanskaya et al., 2023). The frequency of language switches varies among bilingual children and over time, as it correlates with language proficiency (Kapantzoglou et al., 2021). With more stable proficiency in both languages, code-switching loses its role as a scaffold, and the *switch rate* decreases (Smolak et al., 2020; Kapantzoglou et al., 2021).

In contrast to random lexical substitution, the position in an utterance at which code-switching can occur is strongly constrained by structural principles. The *switch point* can be dictated by the structural frame of one of the languages (Myers-Scotton, 1993) or restricted to positions at which the word order of both languages aligns (Poplack, 1980). Bilingual children are sensitive to the code-switching patterns in the mostly spoken input they receive from their interlocutors (Comeau et al., 2003), and they may implicitly infer the syntactic constraints of both linguistic systems from these patterns of switching (Chouinard and Clark, 2003).

Although code-switched input seems beneficial for bilingual proficiency, the training data for bilingual language models usually consists of parallel corpora (e.g. Kim et al., 2019; Lin et al., 2025) or interleaved monolingual datasets (e.g. Conneau et al., 2020; Zheng et al., 2021). While this strategy is effective, it also provides an implicit signal about the language identity of each training example, resulting in cleaner language boundaries than those encountered by bilingual children. To enforce cross-lingual alignment, existing approaches typically include an explicit alignment signal by using a contrastive learning objective (Shen et al., 2025) or synthetic word-level alignments (Kautsar and Koto, 2025). These approaches primarily operate at the lexical level, failing to capture how children implicitly acquire bilingual lexicons and syntactic structure during code-switching.

Recent results indicate that code-switched data can make bilingual learning more efficient, improving cross-lingual representation alignment beyond standard parallel corpora (Wang et al., 2025),

[1] Code available at GitLab repository

Proceedings of the Second BabyLM Workshop, pages 805–822
October 29, 2026 ©2026 Association for Computational Linguistics

while also benefiting downstream instruction tuning (Asano et al., 2026) and reducing the need for massive monolingual pretraining corpora (Zeng et al., 2026). Importantly, code-switched data should not simply be a consequence of mixing languages: the direction and proportion of token insertions can substantially affect performance (Wang et al., 2025), while progressively reducing code-switched content toward monolingual input has also been shown to improve cross-language accuracy and alignment (Yoo et al., 2025).

Despite the promising results, existing work has only examined different factors of code-switched training separately, but not yet how the *switch point* and the *switch rate* jointly shape bilingual representation learning. Therefore, we hypothesize that structurally and developmentally valid code-switched input can facilitate the emergence of shared, syntax-aware bilingual representations without the need for explicit alignment. To test this hypothesis, we train four different code-switching variants across two language pairs with distinct typological distances: English–Dutch (`eng-nld`) and English–Chinese (`eng-zho`), carefully controlling both the *switch rate* and the *switch point*.

To evaluate our models under developmentally plausible constraints, we use the BabyLM training data for the multilingual track (Jumelet et al., 2026a) and the 2026 evaluation pipeline (Choshen et al., 2026), assessing the models' performance on linguistic knowledge and downstream tasks. We show that training on code-switched data generally improves cross-lingual alignment and downstream performance for `eng-nld` models. For `eng-zho` models, however, code-switched contexts negatively influence performance in Chinese. To further understand how training with code-switched data affects the learned representation, we examine the geometry of the embedding space and find that code-switching models lead to better cross-lingual alignments for `eng-nld` models.

2 Related Work

While human bilinguals naturally acquire shared semantic concepts through mixed-language input, current bilingual modeling often ignores this aspect. The following sections review the psycholinguistic principles governing human code-switching and present how these insights are currently being integrated into language modeling.

2.1 Code-Switching in Bilingual Children

During the language acquisition process, exposure to mixed-language input is a common feature of many bilingual environments (Genesee, 2008; Bail et al., 2015; Kremin et al., 2019; Huang, 2025). Despite this input variability, studies show that code-switching carries minimal processing cost in natural settings (Blanco-Elorrieta and Pylkkänen, 2017; Adamou and Shen, 2019), allowing bilingual children to achieve key linguistic milestones within the same overall time frame as monolingual peers (Hambly et al., 2013). Crucially, Bilson et al. (2015) suggest that learning a word in one language supports its acquisition in the other, presumably because the underlying semantic representation has already been established, allowing bilingual learners to map an additional lexical form onto an existing concept. As proficiency in both languages stabilizes, code-switching gradually loses its role as an essential scaffolding mechanism, leading to a corresponding increase in the monolingual utterance (Smolak et al., 2020; Kapantzoglou et al., 2021). Beyond lexical learning, code-switching during language acquisition is also governed by syntactic constraints. Myers-Scotton (1993) suggests that one "matrix" language constructs the morphosyntactic frame, restricting switches predominantly to open-class content words. Poplack (1980), on the other hand, argues that switches occur only at points where the surface word orders of both languages coincide, which opens up more switch points between closely related languages sharing similar syntactic structures.

2.2 Code-Switching in Bilingual Models

In language modeling, introducing code-switched contexts enhances cross-lingual representation alignment beyond standard parallel corpora (Wang et al., 2025), yielding benefits that extend to instruction-tuning (Asano et al., 2026) and enabling models to achieve strong performance without relying on massive monolingual pretraining corpora (Zeng et al., 2026). Code-switching has also been explored as a means of mitigating catastrophic forgetting in multilingual adaptation. By combining POS-based code-switching with a replay adapter strategy, K et al. (2026) shows that training with code-switched data can help preserve previously acquired knowledge while adapting large language models to low-resource languages. However, these advantages are heavily mediated by factors such

as data volume, language-pair similarity, and structural switching dynamics. For instance, Zeng et al. (2026) show that performance scales logarithmically with the volume of available monolingual data in English–Spanish models. Furthermore, structural mechanics such as insertion direction are critical. For example, in Chinese–English models, increasing Chinese token insertions into English yields continuous performance gains, whereas inserting English tokens into Chinese text is beneficial only up to a specific insertion percentage (Wang et al., 2025). To further examine these factors, Yoo et al. (2025) mirror the developmental reduction of human code-switching over time, showing that progressing from frequent intra- and intersentential switching to purely monolingual data improves cross-language accuracy and alignment.

Building on these insights, we compare intrasentential code-switching strategies within a cognitively-inspired setup for bilingual models. Using strict data ceilings as established in the BabyLM challenge (Choshen et al., 2026), we evaluate whether different code-switching variants provide consistent benefits across both typologically close (`eng-nld`) and typologically distant (`eng-zho`) language pairs.

3 Methodology

We train and evaluate one baseline and four code-switching variants for each language pair. Code-switching variants use the same architecture as the baseline, varying only *switch rate* and *switch point*. All models are trained across three random seeds.

3.1 Baseline

Our baseline models are pretrained on a balanced bilingual corpus constructed from the BabyLM 2026 multilingual data collections (Jumelet et al., 2026a; Choshen et al., 2026), summing up to 100M tokens.[2] We adopt the GPT-2-small architecture optimized with a standard causal language modeling objective. A Byte-Pair Encoding (BPE) tokenizer with a vocabulary size of $16,384$ is trained on the unmodified bilingual corpus. We train each model for 10 epochs, with a total exposure of $\sim$1B tokens. Checkpoints are saved at logarithmic intervals, namely, 1M, 10M, 100M, up to 1B words.

[2]Following the BabyLM 2026 guidelines, the Dutch and Chinese sub-corpora are scaled by byte premium factors of 1.05 and 0.93, respectively (Arnett et al., 2024).

3.2 Code-Switching Models

We train four code-switching variants that differ in *switch rate* and *switch point* as shown in Table 1. These models are optimized on the same training data as the baseline, but the languages are mixed according to the respective switching condition.[3]

Model	Switch Rate	Switch Point
Baseline	✗	✗
Static	Fixed	Random
Curriculum	Gradual	Random
POS-Static	Fixed	POS-constrained
POS-Curriculum	Gradual	POS-constrained

Table 1: Overview of code-switching variants.

Switch Rate. The *switch rate* controls the proportion of words that are switched into another language. While Static models maintain a uniform global token density, i.e., targeting 10% switches overall, Curriculum models deliver this same aggregate density through a four-phase dynamic curriculum. Inspired by the correlation between increasing language proficiency and declining code-switching rates in bilingual children (Smolak et al., 2020; Kapantzoglou et al., 2021), the curriculum progressively reduces the data proportion for code-switching across four equal word-budget phases, i.e., 90% $\rightarrow$ 70% $\rightarrow$ 40% $\rightarrow$ 0% of the data are eligible from phase 1 $\rightarrow$ 4. To match the overall 10% target density, tokens within the active switching windows are sampled at a higher rate (~20%). Documents are reshuffled within each phase to ensure variable document exposure.

Switch Point. The *switch point* controls which words are eligible for substitution. Rather than relying on unconstrained random substitution, the POS constrained models only switch at certain word classes. Our setup targets open-class words: Nouns, Verbs, and Adjectives (Urbaneja, 2020; Azuma and Meier, 1997; Prasad and Rao, 2011), motivated by switching constraints observed in human bilingual utterances (Myers-Scotton, 1993) and findings in language models (K et al., 2026). This setting aims to minimize grammatical disruption.

Bilingual Dictionary. To switch to the equivalent word in the target language, we need bilin-

[3]To match the BabyLM challenge guidelines, the word budgets for the code-switching variants are reduced by the number of tokens that are used for creating the bilingual dictionary and POS tagger.

gual dictionaries for eng–nld and eng–zho. Due to the token constraint of the BabyLM challenge (Choshen et al., 2026), we do not use pretrained models for creating the dictionaries. Instead, we extract word alignments from parallel subtitle corpora (Lison and Tiedemann, 2016) using MGIZA (Gao and Vogel, 2008), a multi-threaded implementation of GIZA++ (Och and Ney, 2003). Additionally, we train custom POS taggers on Universal Dependencies treebanks (Nivre et al., 2017)[4] using a BiLSTM architecture without pretrained embeddings. We use the bilingual dictionaries and POS taggers for controlling the *switch rate* and *switch point* of models trained with code-switched data.

4 BabyLM Challenge

We evaluate the models using the benchmark of the BabyLM Multilingual Challenge (Choshen et al., 2026), which includes tasks for English (eng), Dutch (nld), and Chinese (zho).

4.1 Benchmark Tasks

We examine both zero-shot linguistic knowledge and fine-tuned downstream performance.[5]

Zero-shot. Zero-shot evaluation measures capabilities acquired during pretraining without additional updates, focusing on grammatical competence, commonsense reasoning, discourse understanding, and semantic composition across languages. These tasks test whether models develop transferable linguistic knowledge, including sensitivity to syntactic structure, understanding of plausible relations, coherent discourse interpretation, and cross-lingual meaning representation.

Fine-tuning. Fine-tuning evaluation measures the transferability of learned representations by adapting models to downstream tasks. The tasks evaluate broader language understanding abilities, including reasoning, reading comprehension, factual knowledge retrieval, natural language inference, truthfulness, and multilingual classification.

4.2 Model Performance

In this section, we report the model performance change of the four code-switching variants compared to bilingual baselines on the BabyLM benchmark for eng–nld and eng–zho. The results shown in Table 2 are the macro-average of all the tasks, and we report the mean across three random seeds.[6]

Eval.	Model	eng–nld Δeng	eng–nld Δnld	eng–zho Δeng	eng–zho Δzho
ZS	Static	-0.08	**+0.19**	-0.52	-4.14
	Curr.	-0.32	-0.19	-0.20	-3.99
	POS-Static	**+0.38**	-0.30	-0.59	-3.66
	POS-Curr.	-0.54	-0.46	-0.68	-3.79
FT	Static	+2.11	**+5.23**	+6.09	**+0.58**
	Curr.	+1.44	+1.63	**+7.05**	-1.06
	POS-Static	-0.84	+3.31	+3.12	-1.03
	POS-Curr.	**+2.37**	+4.74	+5.73	-0.23

Table 2: Macro-averaged performance changes relative to the bilingual baseline (Δ in percentage points). The results are averaged across three random seeds. ZS: zero-shot; FT: fine-tuning. Best improvements within each language pair and evaluation setting are in **bold**.

English and Dutch. For the eng–nld models, zero-shot performance remains largely unchanged across all code-switching variants, indicating that training with code-switched data preserves general multilingual capabilities. Interestingly, the Static and POS-Static show slightly better overall grammatical knowledge compared to their Curriculum counterparts, indicating that too much code-switching at early training could be slightly harmful for the model's grammatical knowledge. For the fine-tuning tasks, we see that introducing code-switched contexts improves downstream performance for both English and Dutch, with the Static variant achieving the strongest overall results. Overall, these results indicate that training with code-switched data enhances downstream transfer without sacrificing general linguistic knowledge for this closely related language pair.

English and Chinese. For the eng–zho models, the result is more asymmetric. For the zero-shot tasks, we see that training with code-switched data is harmful for the model's linguistic knowledge, with the negative effect being particularly pronounced for Chinese. All code-switching variants outperform the bilingual baseline after fine-tuning, but the improvements are driven primarily by the English side, whereas Chinese downstream performance usually remains under the baseline. These results suggest that, for this typologically distant language pair, intra-sentence code-switching affects some aspects of Chinese knowledge acquired during pretraining.

[4] Specifically, we use English Web Treebank (EWT), Dutch LassySmall, and Chinese PUD treebanks (Silveira et al., 2014; Bouma and van Noord, 2017; McDonald et al., 2013).

[5] Detailed tasks lists are provided in Appendix F.

[6] Results for each task are presented in Appendix G.

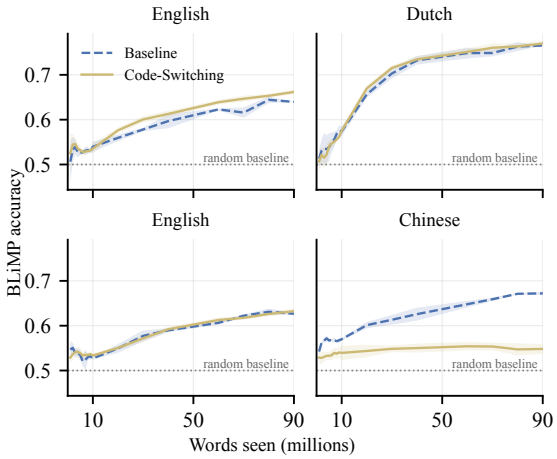


Figure 1: Early learning curve of grammatical knowledge for English–Dutch (top row) and English–Chinese (bottom row) bilingual models. Left column: English BLiMP; right column: target-language BLiMP (Dutch BLiMP for eng–nld, Chinese BLiMP for eng–zho). Each panel shows BLiMP accuracy as a function of words seen during training. The code-switching line represents the mean across four variants. Shaded bands denote standard deviation across random seeds.

4.3 Learning Curve

To inspect learning dynamics, we track learning trajectories on the BLiMP benchmark during early pretraining in Figure 1. For `eng–nld`, code-switch models show slightly faster initial progress compared to the baseline. In contrast, `eng–zho` presents a mixed picture. While English performance remains steady and largely maintains its level relative to the baseline, Chinese accuracy is negatively affected from an early stage. Rather than improving over time, Chinese performance lags behind the baseline early on and fails to recover as pretraining progresses. Overall, these trajectories show that code-switching provides an effective early learning signal for `eng–nld`, but it has an immediate negative impact on the Chinese knowledge in `eng–zho`.

5 Representational Geometry

To better understand how training with code-switched data reshapes the internal structure, we examine our models using a range of metrics comparing their global structure, relative alignment across languages, and the effect on cross-lingual transfer.

5.1 Global Structure

As a first sanity check, we measure how introducing code-switched contexts changes the representational structure of the embedding layer compared to a bilingual baseline.

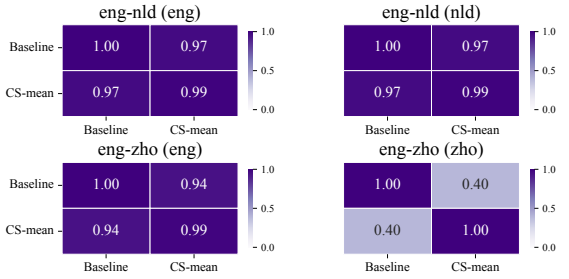


Figure 2: Similarity of the global structure of sentence representations within a language in bilingual models measured by centered kernel alignment (CKA). We compare the *Baseline* model against the mean of code-switching variants (*CS-mean*) for the English–Dutch (`eng–nld`) models in the top row and the English–Chinese (`eng–zho`) models in the bottom row. Darker cell colors indicate higher representational similarity.

Centered Kernel Alignment. We measure the similarity of sentence representations within each language using linear Centered Kernel Alignment (CKA) (Kornblith et al., 2019). Let $X \in \mathbb{R}^{n \times d_x}$ denote the representation of n sentences in the baseline model and $Y \in \mathbb{R}^{n \times d_y}$ denote the representations of the same sentences in a code-switched model. Linear CKA is defined as

$$\mathrm{CKA}(X, Y) = \frac{\|Y^\top X\|_{\mathrm{F}}^2}{\|X^\top X\|_{\mathrm{F}} \, \|Y^\top Y\|_{\mathrm{F}}},$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. CKA quantifies the global similarity of the two representational spaces and is invariant to isotropic scaling and orthogonal transformations. Sentences are represented by the average of their token embeddings.

Results. All code-switching variants lead to highly similar representational structure. We therefore focus on the comparison to the baseline in Figure 2 and average across code-switching conditions and seeds. For the pair `eng–nld`, the global representational structure remains highly similar for both languages. In contrast, the analysis of the `eng–zho` models reveals a highly asymmetric pattern. While the structure of the English representations remains relatively close to the baseline, training with code-switched data fundamentally changes the structure of the Chinese sentence representations. Despite this strong shift relative to the baseline, the structure of code-switched variants is highly similar, indicating that the overall effect is driven primarily by the introduction of

code-switched context rather than by the specific training variant.

5.2 Cross-Lingual Alignment

After assessing what code-switching changes relative to the baseline, we now examine its effect on the alignment of the two languages within a model.

Cosine Similarity of Translations. We measure the alignment between the representation an English input $\mathbf{x}_i$ and the representation of its translation in the target language $\mathbf{y}_i$ by calculating cosine similarity:

$$\cos(\mathbf{x}_i, \mathbf{y}_i) = \frac{\mathbf{x}_i^\top \mathbf{y}_i}{\|\mathbf{x}_i\|_2 \|\mathbf{y}_i\|_2}.$$

In addition to the parallel sentence pairs, we perform the same analysis on a curated set of word pairs. We use the translation pairs described in Section 3.2 and additionally examine a list of known cognates with similar form and meaning in English and Dutch (Lefever et al., 2020).

Results. We calculate the average alignment across all input elements and report the delta over the corresponding baseline model in Figure 3. A higher score indicates that semantically equivalent inputs from the two languages are embedded closer together in the shared representation space. We observe a strong increase in alignment for the `eng-nld` models both on the sentence level (top left) and the lexical level (top right). Code-switching with a static *switch rate* produces the largest increase, but the differences between code-switching variants remain small. We visualize the lexical alignment for 5,285 `eng-nld` cognate pairs when training with the POS-Static variant in Figure 4 and compare it to the baseline.[7] It shows that both approaches closely align cognates with high orthographical overlap (*material-materiaal*, *chocolate-chocolate*), but training with code-switched data increases the alignment for more diverging pairs (*answer-antwoord*, *line-lijn*). This indicates that the code-switched context alone can facilitate better cross-lingual semantic alignment beyond surface patterns without introducing an explicit alignment objective during training.

The effect of code-switched training on cross-lingual sentence alignment is generally weaker for the language pair English–Chinese. Only POS-sensitive models slightly increase the sentence-level alignment relative to the baseline. On the lexical level, the difference in cosine similarity between the baseline and the code-switching variants is even more pronounced than for English–Dutch pairs. However, similarity increases for both translation pairs and random pairs, indicating that the changes in representational structure do not necessarily lead to better bilingual understanding.

[7] Figure 6 in Appendix E presents plots for all variants.

5.3 Relative Structure

We abstract from specific changes in lexical cross-lingual alignment by conducting a second-order representational similarity analysis.

Representational Similarity Analysis. We calculate the Spearman correlation over the cosine similarities between English representations $X^{(A)}$ and target language representations $X^{(B)}$ for translation pairs. While CKA compares feature spaces via linear transformations, RSA evaluates whether two models preserve similar relative distances across lexical translation pairs.

Results. We observe a similar pattern as for the CKA analysis on sentence embeddings. Figure 5 shows that token-level relations remain similar when comparing the code-switching variant to the baseline for all `eng-nld` models. This indicates that training on code-switched data causes only limited reorganization of the embedding geometry. In contrast, we observe a substantially larger divergence for the `eng-zho` models, suggesting that code-switching has a much stronger effect for this language pair that does not share lexical overlap. However, different code-switching variants remain highly similar to one another, indicating that they converge toward a shared lexical geometry despite differences in training strategies.

Overall, training with code-switched data produces a more pronounced representational shift for `eng-zho` than for `eng-nld`, on both global structure and relational alignment.

5.4 Cross-Lingual Transfer

We now examine how the observed changes in cross-lingual alignment induced by training on code-switched data affect the performance on tasks requiring bilingual knowledge.

Translation Retrieval. We test whether the increased alignment on translation pairs suffices to effectively distinguish them from unrelated sentences. We use a similar procedure as Lample et al.

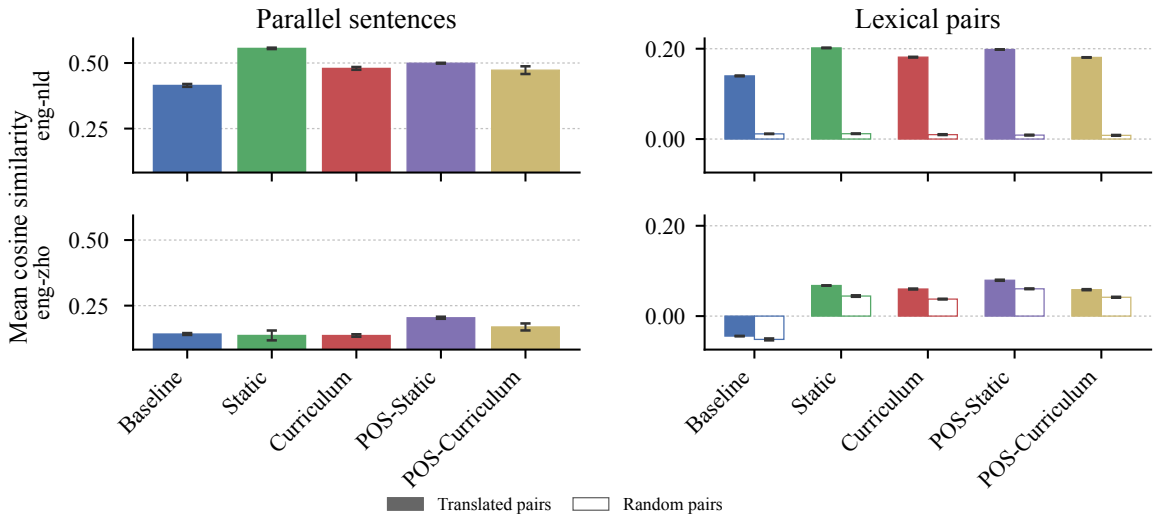


Figure 3: Cross-lingual embedding alignment across models for English–Dutch (eng–nld; top row) and English–Chinese (eng–zho; bottom row). The left column displays the mean cosine similarity for aligned parallel sentences, while the right column shows lexical alignment for curated translation pairs (filled bars) versus randomly matched word pairs (outlined bars).

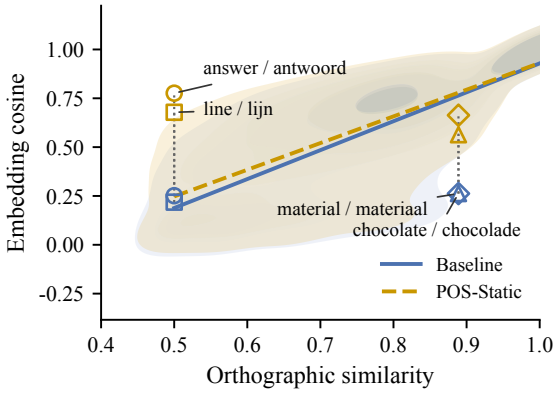


Figure 4: Relationship between orthographic similarity and word embedding cosine similarity for English–Dutch cognates ($n = 5,285$ pairs). Contours indicate kernel density estimates. Dotted connectors link the same cognate pair between the *Baseline* and *POS-Static* setups, illustrating shifts in embedding similarity at equivalent levels of orthographic overlap (e.g., *answer/antwoord*, *line/lijn*), where ideal cross-lingual representations maintain high cosine similarity regardless of distance. Pearson correlations are $r = 0.75$ for Baseline and $r = 0.72$ for POS-Static.

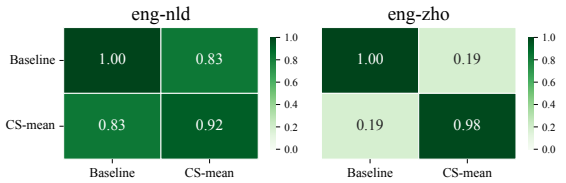


Figure 5: Lexical-level Representational Similarity Analysis (RSA) heatmaps for English—Dutch (eng–nld) and English–Chinese (eng–zho). Panels compare cross-model representational similarity between the *Baseline* model and the average of code-switching variants (*CS-mean*) over paired-token embeddings. Darker cell colors indicate higher representational similarity.

(2018) and rank 1,000 candidate sentence representations according to their cosine similarity to the representation of the source sentence.[8] The translation retrieval performance is reported using Precision@1 (P@1), which measures the percentage of the correct translation that appears among the top-1 retrieved sentences. Higher P@1 indicates stronger cross-lingual alignment.

Results. We report retrieval performance as P@1 over all candidate sentences in Table 3 and see a clear difference between the two language pairs. For eng–nld, training with code-switched data improves translation retrieval over the bilingual baseline. This indicates that code-switching increases not only the absolute similarity of translation pairs as shown in Figure 3, but also their relative distance to other candidate sentences. In contrast, the eng–zho results show a weaker retrieval pattern. All code-switching variants further decrease the already low retrieval precision. While retrieval success remains above the chance level, it indicates that code-switching does not reliably position Chinese sentences closer to their English translations relative to other candidate sentences.

[8] Sentences are represented by the average of their token embeddings.

Language pair	Model	eng → X	X → eng
eng–nld	Baseline	13.9	14.9
	Static	30.5	32.0
	Curr.	20.2	22.1
	POS-Static	21.5	22.4
	POS-Curr.	17.2	19.4
eng–zho	Baseline	5.6	6.9
	Static	2.7	3.0
	Curr.	2.7	3.1
	POS-Static	2.8	3.1
	POS-Curr.	2.6	2.9

Table 3: Sentence-level translation retrieval measured by P@1 (%), showing the percentage of the correct translation that appears among the top-1 retrieved sentences. The results are averaged across three random seeds. The P@1 for random retrieval is 0.1%.

6 Discussion

Our complementary analysis of performance aspects and representational structure provides a coherent picture of how training with code-switched data impacts bilingual language models. Our contrasting findings for `eng-nld` and `eng-zho` models suggest that the benefit of intra-sentence code-switching depends on how easily the two languages can be integrated within the same training context. For the typologically close `eng-nld` pair, the bilingual baseline already leads to a reasonably aligned representation space, and code-switching further improves both the accuracy on linguistic minimal pairs and downstream performance. The strongest results are obtained with the unconstrained Static variant. This may indicate that further constraining the *switch point* using linguistic constraints is not necessary if the surface structure of the two languages is highly similar (as is the case for Dutch and English). These findings are compatible with Poplack's (1980) Equivalence Constraint, under which switching is more likely at points where the surface structures of the two languages can be combined without violating either grammar.

For the `eng-zho` models, training with code-switched data fails to improve meaningful bilingual alignment and also leads to weaker grammatical knowledge in Chinese. At the same time, some code-switched models produce higher absolute similarity between `eng-zho` translation pairs than the bilingual baseline. This suggests that the problem is not simply that the two languages become less similar in the representation space. Rather, the increase in similarity does not translate into better relative alignment, as non-translation pairs may become similarly close. The CKA and RSA results provide further evidence that code-switching substantially reorganizes the Chinese representation space while leaving the English space comparatively stable. These findings suggest that intra-sentence code-switching may alter the organization of Chinese representations without producing the translation-relevant structure needed for effective cross-lingual transfer.

The difficulty of the `eng-zho` setting may partly arise from the substantially different linguistic and orthographic properties of the two languages, which provide fewer straightforward *switch points* for intra-sentence integration. However, this interpretation should be considered alongside the training constraints of the BabyLM challenge (Choshen et al., 2026). The limited token budget requires the POS tags and bilingual dictionary to be self-trained from relatively little data, making the resources used to identify and construct switching points relatively noisy. Such noise may have a greater impact when the languages are more distant, where reliable lexical and structural correspondences are already harder to establish. Thus, the observed difference between `eng-nld` and `eng-zho` may reflect an interaction between properties of the language pair and the quality of the resources available for code-switching, rather than typological distance alone.

7 Conclusion

We tested four cognitively-inspired code-switching variants and examined their effect on bilingual language models for two language pairs. Our results show that training with code-switched data can improve cross-lingual transfer and representational alignment, but its effectiveness seems to depend on the compatibility of the languages.

For the closely related `eng-nld` pair, training with code-switched data generally improves linguistic proficiency, downstream performance, and internal cross-lingual alignment compared to the bilingual baseline. The effects of manipulating the *switch rate* and the*switch point* are relatively small, with the Static variant showing slightly better overall performance. This suggests that the introduction of code-switching itself is more influential than the precise choice of when or how frequently language switching occurs for `eng-nld`. The results for `eng-zho`, however, show that these benefits do not necessarily generalize to all languages. Code-switching variants perform worse than the bilin-

gual baseline on sentence-level translation retrieval and on grammatical tasks, especially for Chinese. Within the eng–zho code-switching variants, POS-Static tends to perform marginally better than the ones with random *switch point* and the ones with high *switch rate*, suggesting that constraining the switching location and reducing switched content at early training stages may better preserve grammatical structure when code-switching between more distant languages.

Overall, our findings suggest that training with code-switched data is a promising mechanism for improving cross-lingual transfer, particularly for languages that have high lexical overlap. Future work could therefore investigate whether typological distance remains a significant factor when larger token budgets, higher-quality POS annotations, and more reliable bilingual dictionaries are available.

Limitations

Although training with code-switched data appears to support cross-lingual transfer, particularly for eng–nld even under the limited-token constraints, several limitations remain regarding the cognitive plausibility of these models.

Importantly, our approach does not strictly mirror the cognitive and interactional realities of bilingual child language acquisition. In naturalistic settings, parent-child code-switching is highly contextualized and communicatively driven; for instance, bilingual parents frequently utilize one language to instruct an infant about a novel object or concept associated with another language (García, 1980; Bail et al., 2015). Our word- and phrase-level substitution framework lacks this semantic and situational grounding, as it relies on a blind stochastic process rather than communicative intent. In addition, the code-switched data we use in this study is synthetic. While we try to introduce high-quality code-switched context, the data are inherently noisy and occasionally invalid, as they are generated via a probabilistic pipeline applied to translated parallel corpora. Another limitation in cognitive plausibility stems from the modeling design: our current pipeline exposes the model to a fixed architectural design from the onset of training, while in human language acquisition, children dynamically expand their vocabulary size, lexical processing speed, and syntactic complexity over time. Incorporating a cognitively plausible architecture remains an important direction for future work.

Acknowledgments

We thank the reviewers for the thoughtful and detailed feedback. We thank Irantzu Elcoro Alberdi for the insightful discussion of the literature on code-switching. This research is partially supported by the zukunft.niedersachsen program of the VolkswagenStiftung (L.B., Z.H., L.P.) and by a VENI grant (Vl.Veni.211C.039) from the Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO) (L.B.).

Author Contributions

Z.H.: Conceptualization, Analysis, Literature review, Visualization, Writing—original draft, Writing—review & editing. **L.P.:** Conceptualization, Literature review, Writing—original draft (parts of Introduction and Related Work). **L.B.:** Supervision, Conceptualization, Analysis, Writing—review & editing.

References

Evangelia Adamou and Xingjia Rachel Shen. 2019. There are no language switching costs when codeswitching is frequent. *International Journal of Bilingualism*, 23(1):53–70.

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O. Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Catherine Arnett, Tyler A. Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 1–9, Torino, Italia. ELRA and ICCL.

Shunta Asano, Jeonghun Baek, and Toshihiko Yamasaki. 2026. Beyond bilingual transfer: Multilingual code-switching in instruction tuning. *Preprint*, arXiv:2605.29414.

Shoji Azuma and Richard P. Meier. 1997. Open class and closed class: Sentence-imitation experiments on intrasentential code-switching. *Applied Psycholinguistics*, 18(3):257–276.

Amelie Bail, Giovanna Morini, and Rochelle S. Newman. 2015. Look at the gato! Code-switching in speech to toddlers. *Journal of Child Language*, 42(5):1073–1101.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

Samuel Bilson, Hanako Yoshida, Crystal D. Tran, Elizabeth A. Woods, and Thomas T. Hills. 2015. Semantic facilitation in bilingual first language acquisition. *Cognition*, 140:122–134.

Esti Blanco-Elorrieta and Liina Pylkkänen. 2017. Bilingual Language Switching in the Laboratory versus in the Wild: The Spatiotemporal Dynamics of Adaptive Language Control. *The Journal of Neuroscience*, 37(37):9022–9036.

Gosse Bouma and Gertjan van Noord. 2017. Increasing return on annotation investment: The automatic construction of a Universal Dependency treebank for Dutch. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 19–26, Gothenburg, Sweden. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Krista Byers-Heinlein. 2013. Parental language mixing: Its measurement and the relation of mixed input to young bilingual children's vocabulary size. *Bilingualism: Language and Cognition*, 16(1):32–48.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Michelle M. Chouinard and Eve V. Clark. 2003. Adult reformulations of child errors as negative evidence. *Journal of Child Language*, 30(3):637–669.

Liane Comeau, Fred Genesee, and Lindsay Lapaquette. 2003. The Modeling Hypothesis and child bilingual codemixing. *International Journal of Bilingualism*, 7(2):113–126.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Margaret Deuchar and Suzanne Quay. 1994. Language choice and code-switching in a young bilingual child. In *Proceedings of the 10th Annual Sociolinguistics Symposium*, Lancaster, England, United Kingdom. 18-page conference paper.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57, Columbus, Ohio. Association for Computational Linguistics.

Eugene E. García. 1980. The function of language switching during bilingual mother-child interactions. *Journal of Multilingual and Multicultural Development*, 1(3):243–252.

Dorota Gaskins, Antje Endesfelder Quick, Anna Verschik, and Ad Backus. 2022. Usage-based approaches to child code-switching: State of the art and ways forward. *Cognitive Development*, 64:101269.

Fred Genesee. 2008. Bilingual First Language Acquisition: Evidence from Montreal. *Diversité urbaine*, pages 9–26.

Helen Hambly, Yvonne Wren, Sharynne McLeod, and Sue Roulstone. 2013. The influence of bilingualism on speech production: A systematic review. *International Journal of Language & Communication Disorders*, 48(1):1–24.

Wenhan Han, Yifan Zhang, Zhixun Chen, Binbinliu, Mykola Pechenizkiy, Meng Fang, and Yin Zheng. 2026. MuBench: Assessment of multilingual capabilities of large language models across 61 languages. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 16163–16192, San Diego, California, United States. Association for Computational Linguistics.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, Helmut Schmid, Hinrich Schütze, and Nima Mesgarani. 2025. XCOMPS: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81, Vienna, Austria. Association for Computational Linguistics.

Qiaoya Huang. 2025. Bilingual children's code-switching: Types, approaches, and functions. *International Journal of Social Science and Education Research*, 7(1):38–46.

Jaap Jumelet, Abdellah Fourtassi, Akari Haga, Bastian Bunzeck, Bhargav Shandilya, Diana Galvan-Sosa, Faiz Ghifari Haznitrama, Francesca Padovani, Francois Meyer, Hai Hu, Julen Etxaniz, Laurent Prevot, Linyang He, María Grandury, Mila Marcheva, Negar Foroutan, Nikitas Theodoropoulos, Pouya Sadeghi, Siyuan Song, and 7 others. 2026a. BabyBabelLM: A multilingual benchmark of developmentally plausible training data. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3297–3329, Rabat, Morocco. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026b. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Santosh Srinath K, Mudit Somani, Varun Reddy Padala, Prajna Devi Upadhyay, and Abhijit Das. 2026. Continual-learning for modelling low-resource languages from large language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6258–6275, Rabat, Morocco. Association for Computational Linguistics.

Maria Kapantzoglou, Julie Esparza Brown, Lauren M. Cycyk, and Gerasimos Fergadiotis. 2021. Code-switching and language proficiency in bilingual children with and without developmental language disorder. *Journal of Speech, Language, and Hearing Research*, 64(5):1605–1620.

Margarita Kaushanskaya, Kimberly Crespo, and Anne Neveu. 2023. Does code-switching influence novel word learning? *Developmental Science*, 26(2):e13292.

Muhammad Dehan Al Kautsar and Fajri Koto. 2025. Parallel tokenizers: Rethinking vocabulary design for cross-lingual transfer. *Preprint*, arXiv:2510.06128.

Yunsu Kim, Hendrik Rosendahl, Nick Rossenbach, Jan Rosendahl, Shahram Khadivi, and Hermann Ney. 2019. Learning bilingual sentence embeddings via autoencoding and computing similarities with a multilayer perceptron. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 61–71, Florence, Italy. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR.

Lena V. Kremin, Adriel John Orena, Linda Polka, and Krista Byers-Heinlein. 2019. Investigating naturalistic code-switching directed towards infants. *The Journal of the Acoustical Society of America*, 145(3_Supplement):1766–1766.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Els Lefever, Sofie Labat, and Pranaydeep Singh. 2020. Identifying cognates in English-Dutch and French-Dutch by means of orthographic information and cross-lingual word embeddings. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4096–4101, Marseille, France. European Language Resources Association.

Peiqin Lin, Andre Martins, and Hinrich Schuetze. 2025. A recipe of parallel corpora exploitation for multilingual large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4038–4050, Albuquerque, New Mexico. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Yikang Liu, Yeting Shen, Hongao Zhu, Lilong Xu, Zhiheng Qian, Siyuan Song, Kejia Zhang, Jialong Tang, Pei Zhang, Baosong Yang, Rui Wang, and Hai Hu. 2025. A systematic assessment of language models with linguistic minimal pairs in chinese. *Preprint*, arXiv:2411.06096.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Jürgen M. Meisel. 2019. *Keeping Languages Apart: Mixing, Interference and Interaction of Languages*, page 93–122. Cambridge University Press.

Carol Myers-Scotton. 1993. *Duelling Languages: Grammatical Structure in Codeswitching*. Oxford University PressOxford.

Joakim Nivre, Daniel Zeman, Filip Ginter, and Francis Tyers. 2017. Universal Dependencies. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Valencia, Spain. Association for Computational Linguistics.

Franz Josef Och. 1999. An efficient method for determining bilingual word classes. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 71–76, Bergen, Norway. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Johanne Paradis, Elena Nicoladis, and Fred Genesee. 2000. Early emergence of structural constraints on code-mixing: evidence from french–english bilingual children. *Bilingualism: Language and Cognition*, 3(3):245–261.

Shana Poplack. 1980. Sometimes I'll start a sentence in Spanish Y TERMINO EN ESPAÑOL: toward a typology of code-switching1. *Linguistics*, 18(7-8).

B. A. Mahalakshmi Prasad and Prema K. S. Rao. 2011. Code Switching in the Lexical Corpora of Children. *Theory and Practice in Language Studies*, 1(5):471–477.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A Haggag, Imanol Schlag, Marzieh Fadaee, Sara Hooker, Antoine Bosselut, and 1 others. 2024. Include: Evaluating multilingual language understanding with regional knowledge. *ICLR*.

Julie M. Schneider, Giovanna Morini, Veronica De Jesus, and Aquiles Iglesias. 2026. Development of intrasentential code-switching patterns in spanish-speaking english learners. *Child Development*, 97(4):1013–1027.

Yingli Shen, Wen Lai, Shuo Wang, Ge Gao, Kangyang Luo, Alexander Fraser, and Maosong Sun. 2025. From unaligned to aligned: Scaling multilingual LLMs with multi-way parallel corpora. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7357–7379, Suzhou, China. Association for Computational Linguistics.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel R. Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904, Reykjavik, Iceland. European Language Resources Association (ELRA).

Erin Smolak, Stephanie De Anda, Bianka Enriquez, Diane Poulin-Dubois, and Margaret Friend. 2020. Code-switching in young bilingual toddlers: A longitudinal, cross-language investigation. *Bilingualism: Language and Cognition*, 23(3):500–518.

Michelle Suijkerbuijk, Zoë Prins, Marianne de Heer Kloots, Willem Zuidema, and Stefan L. Frank. 2025. BLiMP-NL: A corpus of Dutch minimal pairs and acceptability judgments for language model evaluation. *Computational Linguistics*, 51(4):1267–1301.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*.

Desirée Ramírez Urbaneja. 2020. "¿Tú tienes una little pumpkin?" Mixed noun phrases in Spanish-English bilingual children and adults. *International Journal of Bilingualism*, 24(4):824–839. _eprint: https://doi.org/10.1177/1367006919888580.

Zhijun Wang, Jiahuan Li, Hao Zhou, Rongxiang Weng, Jingang Wang, Xin Huang, Xue Han, Junlan Feng, Chao Deng, and Shujian Huang. 2025. Investigating and scaling up code-switching for multilingual language model pre-training. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11032–11046, Vienna, Austria. Association for Computational Linguistics.

Janet F. Werker and Krista Byers-Heinlein. 2008. Bilingualism in infancy: first steps in perception and comprehension. *Trends in Cognitive Sciences*, 12(4):144–151.

Haneul Yoo, Cheonbok Park, Sangdoo Yun, Alice Oh, and Hwaran Lee. 2025. Code-Switching Curriculum Learning for Multilingual Transfer in LLMs. *arXiv preprint*. ArXiv:2411.02460 [cs.CL].

W. Quin Yow, Jessica S. H. Tan, and Suzanne Flynn. 2018. Code-switching as a marker of linguistic competence in bilingual children. *Bilingualism: Language and Cognition*, 21(5):1075–1090.

Linda Zeng, Steven Y. Feng, and Michael C. Frank. 2026. Bringing Up a Bilingual BabyLM: Investigating Multilingual Language Acquisition Using Small-Scale Models. *arXiv preprint*. ArXiv:2603.29552 [cs.CL].

Francis Zheng, Machel Reid, Edison Marrese-Taylor, and Yutaka Matsuo. 2021. Low-resource machine translation using cross-lingual language model pretraining. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 234–240, Online. Association for Computational Linguistics.

A Building the Bilingual Dictionaries

The bilingual dictionaries utilized in our code-switching pipeline are built from parallel subtitle corpora from OpenSubtitles (Lison and Tiedemann, 2016), obtained via the OPUS platform (Tiedemann, 2012). Prior to statistical alignment, all sentence pairs were filtered to enforce a maximum length constraint of 80 tokens per side and a length ratio of $\leq$ 9:1. The specific dataset parameters, release configurations, and resulting sentence pair counts utilized during the alignment process are detailed in Table 4.

Languages	Release	Aligned Sentence Pairs
eng-nld	v2018	325,874
eng-zho	v2024	504,020

Table 4: Parallel Corpus Configurations for Bilingual Lexicon Extraction

To learn word-level translation mappings, we perform unsupervised word alignment using MGIZA (Gao and Vogel, 2008), a multi-threaded extension of GIZA++ (Och and Ney, 2003). The preprocessing pipeline first generates internal source vocabularies and sentence co-occurrence structures via `snt2cooc`, while `mkcls` (Och, 1999) clusters words into statistical frequency classes. Word alignments are trained using the Expectation–Maximization (EM) algorithm (Brown et al., 1993). To bootstrap baseline lexical translation probabilities, the optimization routine begins with 5 iterations of IBM Model 1. The alignment pipeline then transitions to 3 iterations each of IBM Models 3 and 4 to account for word-order reordering (distortion) and one-to-many word mappings (fertility). Models are trained symmetrically in both directions (source $\rightarrow$ target and target $\rightarrow$ source). Finally, the resulting directional alignments are merged using the standard `grow-diag-final-and` symmetrization heuristic (Koehn et al., 2003) to extract high-confidence word and multi-word alignment links.

English–Dutch Lexicon Extraction Prior to statistical alignment, the English and Dutch text is standardized by lowercasing and applying whitespace tokenization. Once the unified bidirectional alignments are produced by the Pharaoh-formatted alignment map, a unigram extraction layer processes the aligned sentences to count every valid one-to-one mapping across the corpus. To eliminate structural alignment noise, spurious translations, and low-frequency accidental links, we enforce a strict minimum co-occurrence threshold where a word translation pair must appear ≥ 5 times in the corpus to be retained. Punctuation marks are automatically stripped, and the final filtered lexicon maps individual source unigrams directly to their most probable target counterparts.

English–Chinese Lexicon Extraction Prior to running the alignment models, the text is processed without relying on any pretrained segmenters or tokenizers: the English data is lowercased and whitespace-tokenized, whereas the Chinese sentences are split strictly at the character level, treating each individual character as an autonomous token. From the unified bidirectional alignments, the lexicon is constructed via a dual-layer extraction architecture to properly handle structural differences:

1. **Unigram Layer:** Captures and counts every direct one-to-one alignment link across the corpus (e.g., mapping individual English words to single Chinese characters).

2. **Phrase Layer:** Extracted via standard Moses-style phrase extraction to recover compound multi-character concepts. For each alignment anchor, the algorithm identifies all contiguous English and Chinese spans (up to a maximum boundary constraint of 5 English tokens and 6 Chinese characters), where every internal alignment link is strictly consistent, and no links cross outside the candidate span boundaries.

Once multi-word English expressions and multi-character Chinese compounds are recovered, contiguous Chinese characters within an extracted phrase are joined into a unified, non-whitespace string (e.g., fusing character-level outputs to form a multi-character word). The unigram and phrase layers are subsequently merged into a master dictionary. Mappings are filtered out if they have fewer than a minimum co-occurrence frequency threshold (≥ 2 occurrences), punctuation marks are stripped, and English-only function-word phrases are discarded. Finally, a subsumption pruning pass removes shorter dictionary entries if they are strictly dominated by a longer phrase that possesses an equal or higher co-occurrence frequency.

Runtime Dictionary Utilization Each generated entry in both lexicons tracks total occurrence

counts alongside the empirical translation probability $P(\text{target} \mid \text{source})$, representing how frequently a specific translation appears among all candidate options for a given source term. At runtime, the data pipeline queries these ranked candidates directly, choosing the highest-probability option.

B POS tagger Trainer

In this work, we train part-of-speech (POS) taggers for English, Chinese, and Dutch using the Universal Dependencies treebanks (Nivre et al., 2017). Each dataset is provided in CoNLL-U format and is split into standard training, development, and test partitions.

We employ a BiLSTM-CRF architecture implemented in PyTorch. The model is trained from scratch without the use of pretrained word embeddings. Instead, it learns both word-level and character-level representations directly from the training data. Each token is represented as the concatenation of a learned word embedding and a character-level representation obtained via a convolutional neural network over characters. This allows the model to better handle rare and out-of-vocabulary words. The sequence of token representations is then processed by a bidirectional LSTM, which captures contextual information from both the left and right contexts of each word. The resulting contextualized representations are passed through a linear layer to produce emission scores over POS tags. A conditional random field (CRF) layer is used on top to model dependencies between adjacent tags and to decode the globally optimal tag sequence.

For data preparation, sentences are extracted from the CoNLL-U files as sequences of tokens paired with their gold POS tags. We construct vocabularies for words, characters, and POS tags based on the training set. During inference, unseen words are mapped to a special `UNK` token.

The model is trained using the Adam optimizer with a learning rate of 10^{-3}, a batch size of 32, and gradient clipping to improve training stability. The training objective is the negative log-likelihood of the correct tag sequence under the CRF layer. After each epoch, performance is evaluated on the development set using token-level accuracy. The best-performing model on the development set is saved separately for each language. Early stopping is applied if development accuracy does not improve for three consecutive epochs.

At evaluation time, the saved model checkpoints are used to compute performance on both the development and test sets. We report overall token-level accuracy as well as per-tag accuracy for major POS categories. The models achieve approximately 82% accuracy on English, 85% on Chinese, and 83% on Dutch test sets. Performance is generally strongest on frequent and syntactically regular tags, while lower performance is observed on rare categories such as interjections and foreign words.

Once trained, the models assign a POS tag to each token in an input sentence. For example, in English, *orange* may be labeled as an ADJ and *cat* as a NOUN. However, in Chinese, as a word can be formed by several characters, the POS tagger can assign one tag to a compound word. For example, 橘猫 (orange cat) can be labeled as NOUN.

C Data Licensing

The training data used for our models follow the licensing conditions described by Jumelet et al. (2026a), who report that, during their data collection effort, they verified that all data were released with licenses permitting academic research, such as Creative Commons licenses or Public Domain.

For the construction of the bilingual dictionaries, we used the OpenSubtitles2016 corpus (Lison and Tiedemann, 2016), which is distributed through the OPUS collection and was made freely available to the research community. For training the POS tagger, we used the Universal Dependencies (UD) English Web Treebank (EWT) (Silveira et al., 2014), Dutch LassySmall (Bouma and van Noord, 2017), and Chinese PUD treebanks (McDonald et al., 2013). The English-EWT and Dutch-LassySmall treebanks are licensed under CC BY-SA 4.0, while the Chinese-PUD treebank is licensed under CC BY-SA 3.0. We used these resources in accordance with their respective licensing terms.

Finally, the Dutch–English cognate dataset used for cross-lingual alignment is released under the Creative Commons Attribution–NonCommercial (CC BY-NC) license, which permits sharing and adaptation with appropriate attribution but prohibits commercial use.

D Hyperparameter

The hyperparameters of model training are described in Table 5.

Parameter	Value
Model architecture	12-layer GPT-2 with hidden size 768 and 12 attention heads.
Tokenizer	16,384-token shared bilingual BPE vocabulary.
Optimizer	AdamW.
Learning Rate	5×10^{-5}. Cosine decay with 1% linear warmup.
Batch size	16.
Sequence length	512 tokens.
Randomization	Independent random seeds (13, 42, 123) for model initialization and code-switch decisions.

Table 5: Training hyperparameters and configuration.

E Cognates Alignment

Figure 6 shows the Pearson correlation between orthographic similarity and word embedding cosine similarity for English–Dutch cognates. The figure shows the results for the bilingual baseline and all four code-switching variants.

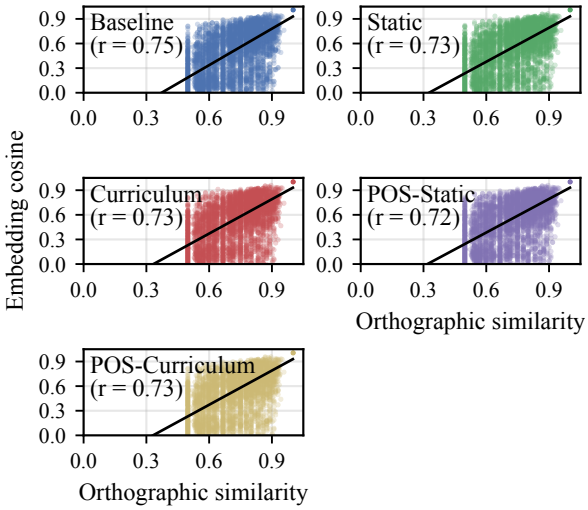


Figure 6: Relationship between orthographic similarity and word embedding cosine similarity for English–Dutch cognates ($n = 5,285$ pairs). Lower Pearson correlation indicates better cross-lingual alignment, especially for cognates with less similar surface forms.

F BabyLM Benchmark

Table 6 and Table 7 list all the tasks being evaluated for `eng-nld` and `eng-zho` models.

Evaluation Dimension	eng–nld Tasks	eng–zho Tasks
Grammatical competence	blimp_babylm_filtered[1], multiblimp_eng[2], blimp_nl[3], multiblimp_nld[2]	blimp_babylm_filtered[1], multiblimp_eng[2], zhoblimp[4]
Commonsense reasoning	hellaswag_en_mubench[5], winogrande_en_mubench[5], hellaswag_nl_mubench[5], winogrande_nl_mubench[5]	hellaswag_en_mubench[5], winogrande_en_mubench[5], hellaswag_zh_mubench[5], winogrande_zh_mubench[5]
Discourse understanding	xstorycloze_en_mubench[5], xstorycloze_nl_mubench[5]	xstorycloze_en_mubench[5], xstorycloze_zh_mubench[5]
Compositional semantics	xcomps_nl[6]	xcomps_zh[6]

Table 6: Zero-shot evaluation benchmarks grouped by evaluation dimension. [1](Choshen et al., 2026); [2](Jumelet et al., 2026b); [3](Suijkerbuijk et al., 2025); [4](Liu et al., 2025); [5](Han et al., 2026); [6](He et al., 2025);

Evaluation Dimension	eng–nld Tasks	eng–zho Tasks
Science and general reasoning	arc_mubench[1]	arc_mubench[1]
Reading comprehension	belebele[2]	belebele[2]
Factual knowledge	bmlama_mubench[1]	bmlama_mubench[1]
Natural language inference	mnli_mubench[1], xnli (eng)[3], mnli_mubench (nld)[1]	mnli_mubench[1], xnli[3]
Topic classification	sib200[4]	sib200[4]
Truthfulness and calibration	truthfulqa_mubench[1]	truthfulqa_mubench[1]
Language-specific understanding	include (nld)[5]	include (zho)[5]

Table 7: Fine-tuning evaluation benchmarks grouped by evaluation dimension. [1](Han et al., 2026); [2](Bandarkar et al., 2024); [3](Conneau et al., 2018); [4](Adelani et al., 2024); [5](Romanou et al., 2024);

G Per-Category Results

Table 8 and Table 9 present the results of all models across each task.

Task	Setting	Baseline	Static	Curriculum	POS-Static	POS-Curriculum
Zero-Shot						
BLiMP	eng	72.6	**73.1**	72.6	72.7	72.4
MultiBLiMP	eng	87.7	**88.8**	87.9	88.1	88.3
Hellaswag	eng	26.5	26.5	26.5	26.4	**26.6**
Winogrande	eng	50.8	50.3	50.3	**51.1**	50.1
XStoryCloze	eng	49.1	49.0	48.6	**49.2**	48.6
PIQA parallel	eng	22.8	20.4	20.1	**23.3**	19.7
PIQA nonparallel	eng	49.5	51.0	**51.3**	50.7	50.0
BLiMP	nld	80.8	80.9	80.6	**81.1**	80.1
MultiBLiMP	nld	93.8	**94.0**	93.5	93.5	93.1
Hellaswag	nld	26.5	26.5	**26.6**	26.3	**26.6**
Winogrande	nld	49.8	**50.2**	49.6	49.8	**50.2**
XCOMPS	nld	53.8	54.0	**54.3**	53.9	**54.3**
XStoryCloze	nld	48.4	**48.6**	47.7	48.2	48.3
PIQA parallel	nld	22.7	22.3	22.3	18.8	**23.6**
PIQA nonparallel	nld	55.0	55.7	54.7	**56.7**	50.7
Fine-Tuned						
ARC	eng	**25.6**	24.7	25.1	25.1	24.4
ARC	nld	**24.9**	23.8	**24.9**	**24.9**	24.1
Belebele	eng	25.0	23.9	23.9	23.9	**25.6**
Belebele	nld	23.9	**28.2**	25.0	25.0	28.1
BM-LAMA	eng	9.7	10.2	11.3	9.7	**11.8**
BM-LAMA	nld	11.5	**12.0**	10.3	10.2	10.0
MNLI	eng	**50.3**	**50.3**	48.5	49.2	49.7
MNLI	nld	49.6	49.8	48.9	51.0	**51.3**
SIB-200	eng	59.8	**78.3**	76.2	58.8	77.7
SIB-200	nld	39.8	72.2	55.8	56.7	**72.7**
TruthfulQA	eng	**27.7**	**27.7**	24.7	25.6	25.4
TruthfulQA	nld	26.2	22.3	23.2	**27.1**	25.0
XNLI	eng	47.3	45.2	45.9	47.2	**47.4**
INCLUDE	nld	29.5	**33.6**	28.6	**33.6**	27.2

Table 8: Evaluation results for `eng-nld`. The results are averaged across three random seeds.

Task	Setting	Baseline	Static	Curriculum	POS-Static	POS-Curriculum
Zero-Shot						
BLiMP filtered	eng	71.3	**71.8**	71.7	70.0	71.5
MultiBLiMP	eng	86.0	**87.2**	86.9	85.2	86.5
Hellaswag	eng	**26.5**	**26.5**	26.4	26.2	26.4
Winogrande	eng	50.9	50.6	50.8	48.9	**51.1**
XStoryCloze	eng	**50.2**	48.8	49.0	47.5	48.6
PIQA parallel	eng	25.6	25.9	27.2	**28.8**	26.9
PIQA nonparallel	eng	**52.3**	49.3	50.0	51.3	48.0
Hellaswag	zho	**26.9**	25.9	25.9	25.8	26.0
Winogrande	zho	49.3	51.0	50.6	50.8	**51.1**
XCOMPS	zho	**53.4**	50.3	50.3	50.8	50.7
XStoryCloze	zho	**48.8**	46.2	46.3	46.6	46.6
ZhoBLiMP	zho	**76.0**	54.9	55.1	54.5	56.0
Hanzi structure	zho	**56.2**	50.1	50.6	50.6	50.9
Hanzi pinyin	zho	49.2	50.9	**51.4**	50.0	51.0
PIQA parallel	zho	**21.8**	19.7	18.8	20.7	17.5
PIQA nonparallel	zho	**50.5**	46.0	47.3	49.3	48.3
Fine-Tuned						
ARC	eng	24.8	23.8	**26.2**	24.4	24.5
ARC	zho	24.6	24.2	**25.5**	25.1	25.1
Belebele	eng	25.0	25.9	25.0	**26.1**	**26.1**
Belebele	zho	25.0	22.5	24.2	23.9	**26.9**
BM-LAMA	eng	10.1	10.3	10.9	10.2	**11.9**
BM-LAMA	zho	9.5	10.2	10.1	**10.9**	10.5
MNLI	eng	46.4	49.8	**56.0**	49.4	50.3
MNLI	zho	**45.8**	38.8	42.2	39.1	41.9
SIB-200	eng	36.8	**76.8**	76.2	56.8	74.7
SIB-200	zho	40.0	**51.3**	29.3	37.0	36.8
TruthfulQA	eng	21.7	**26.5**	24.4	23.2	25.0
TruthfulQA	zho	22.3	25.3	**27.7**	**27.7**	23.5
XNLI	eng	45.2	45.6	**47.7**	44.6	43.3
XNLI	zho	45.1	**46.1**	45.6	45.4	45.1
INCLUDE	zho	26.8	27.1	26.5	22.0	**28.3**

Table 9: Evaluation results for eng−zho. The results are averaged across three random seeds.

BabySteps at BabyLM 2026: Stronger Optimization and Causal Baselines for Data-Constrained Pretraining

Siddhi Satheesh* **Jorge Murillo*** **Yubei Chen**
University of California, Davis
{svsatheesh, joemurillo, ybchen}@ucdavis.edu

Abstract

We present three BabySteps systems that strengthen baselines for data-constrained pretraining in the strict-small (10M words) and strict (100M words) NLP tracks of the 2026 BabyLM Challenge. Across our GPT-BERT experiments, optimizer choice and learning-rate tuning produced our largest gains, highlighting the importance of well-tuned baselines for evaluating progress in sample-efficient pretraining. The curriculum, synthetic-data, teacher-supervision, and corpus-mixing interventions that we tested did not improve the final aggregate score.

We also test whether the official GPT-2 checkpoint is a sufficiently strong reference for causal language modeling. A 127.6M-parameter Llama-style decoder trained with Muon and Liger-optimized kernels obtains a seven-suite NLP average of 56.18, compared with 52.79 for our completed reevaluation of the official BabyLM GPT-2 strict checkpoint, and scores higher on all seven categories. This is not a controlled architectural ablation: architecture, optimizer, implementation, and hyperparameters change together. Instead, it provides a stronger practical causal baseline. In addition, we report failed data interventions, a numerical failure caused by batches with no supervised target tokens, and substantial variance between small evaluation suites.

1 Introduction

The BabyLM Challenge (Warstadt et al., 2023; BabyLM Organizing Committee, 2026) asks whether language models can be made dramatically more sample-efficient by restricting pretraining to developmentally plausible budgets of 10M (strict-small) or 100M (strict) words. Plausible approaches to sample-efficient pretraining include: curate a better corpus, order it like a curriculum, generate synthetic text targeted at the evaluations, or distill knowledge from a teacher. We pursued those intuitions, and this paper is in part a report on how they failed for us.

Our results point to a broader lesson about baseline quality. Within GPT-BERT (Charpentier and Samuel, 2024), we obtained larger and more reliable gains from Muon-family optimization and learning-rate tuning (Jordan et al., 2024; Si et al., 2025) than from any of the data-centric interventions we tested. Separately, we found that the official GPT-2 baseline was not a strong representation of what a modern causal decoder could achieve. We therefore trained a Llama-style causal model (Touvron et al., 2023) using Muon and a Liger-optimized implementation (Hsu et al., 2024), primarily to provide a stronger practical control rather than to propose an architectural innovation.

We make three contributions:

- Across our experiments, Muon-family optimization and learning-rate tuning produced our strongest results; none of the tested data interventions improved the final scores.
- We introduce a modern Llama-style causal baseline that improves the completed official GPT-2 strict result from 52.79 to 56.18 NLP average and outperforms it on every individual NLP benchmark.
- We document evaluation and reproducibility pitfalls that can materially distort conclusions at BabyLM scale.

We submit **BabySteps MurphysLaw-10M-mixed**, a strict-small GPT-BERT trained with AdaMuon; **BabySteps SidBert-100M-mixed**, a strict GPT-BERT trained with Muon; and **BabySteps Chatty BERT**, a strict Llama-style causal decoder trained with Muon using memory-efficient fused kernels.

* Equal contribution.

Proceedings of the Second BabyLM Workshop, pages 823–830
October 29, 2026 ©2026 Association for Computational Linguistics

2 Method

2.1 Model families

2.1.1 GPT-BERT reconstruction

Two of our three submissions use GPT-BERT (Charpentier and Samuel, 2024), which trains a single transformer under a mixture of masked-language-modeling and causal objectives and can subsequently be evaluated in either causal or masked-next-token-prediction (MNTP) modes. We reimplemented the training loop—on both single and multi-GPU runs—using the published baseline configurations (the "reconstruction" in our internal naming), matched the BabyLM baseline model sizes for each track, and verified the implementation by reproducing baseline-level zero-shot scores before changing the optimization recipe.

Both GPT-BERT submissions use the *mixed* objective family and are evaluated in MNTP mode. In our sweeps, this combination provided the strongest overall balance across the 2026 evaluation suites, although causal-focus configurations remained competitive on several individual benchmarks.

2.1.2 Modern causal baseline: Llama with Liger

The third system, *BabySteps Chatty BERT*, is a decoder-only Llama-style model (Touvron et al., 2023). Despite its leaderboard name, it is not a BERT-family model, and we added a disguise emoji to point to that. We trained it from scratch on the 100M-word BabyLM 2026 Strict corpus to test whether the official GPT-2 baseline accurately represents the performance attainable by a modern causal decoder under the same data restriction. Our goal with Chatty BERT was to provide a stronger practical baseline rather than to propose a novel architecture.

The model contains 127.6M parameters, 12 transformer layers, a hidden dimension of 896, an intermediate dimension of 2,304, 14 attention heads, and a maximum context length of 1,024 tokens. It uses a 16,384-entry byte-level BPE vocabulary with tied input and output embeddings.

We used the Hugging Face Llama patch provided by Liger Kernel (Hsu et al., 2024), enabling its optimized implementations of rotary positional embeddings, SwiGLU, and RMSNorm. We did not enable Liger's cross-entropy or fused linear cross-entropy kernels. This implementation made the Llama–Muon configuration a practical 100M-word baseline on a single GPU by reducing the memory and execution overhead associated with these operations.

The single largest and most reliable improvement over the AdamW-trained baselines came from the optimizer. We use Muon (Jordan et al., 2024), which orthogonalizes momentum updates for hidden weight matrices, and AdaMuon (Si et al., 2025), an adaptive variant, with a separate auxiliary learning rate for the non-matrix parameters (embeddings, norms, head). Track-level winners differed: at 10M words, AdaMuon with learning rate 0.02 and auxiliary rate 0.006 was best; at 100M words, Muon with learning rate 0.005 and auxiliary rate 0.007 was best, and the 100M scale showed a sharp cliff at higher rates (a 3×10^{-5} finetuning learning rate was already enough to partially degenerate some GLUE tasks on a sibling run). All final models use seed 42 and identical data ordering to their development runs.

2.2 Tail-weight averaging

For the strict-small entry we submit the average of the weights over the final 20% of training steps rather than the last checkpoint. On the twin development run this left zero-shot scores essentially unchanged (± 0.3 BLiMP) but improved finetuned SuperGLUE by $+1.6$ points (70.1 vs. 68.5 average), and the improvement was stable across finetuning learning rates and seeds. The 100M entry uses final weights, as tail averaging did not help at that scale in our sweeps.

2.3 Training configuration and checkpoints

MurphysLaw trains for 9,914 steps at 16,384 tokens per step ($\approx$10 epochs of the official strict-small corpus); SidBert trains for 12,330 steps at 131,072 tokens per step on the official strict corpus. The Llama submission trains for 13,510 steps at 131,072 tokens per step ($\approx$10 epochs of the official strict corpus). The models save the word-count milestone checkpoints required by the 2026 challenge: 19 for strict-small and 28 for strict. We publish these as revisions of the submission repositories, with the final weights on `main`, giving 20 and 29 published model states, respectively. We evaluate them with the official fast pipeline, and the same milestone checkpoints drive the Age-of-Acquisition analysis (Chang and Bergen, 2022; Frank et al., 2017). Finetuning for SuperGLUE follows the official pipeline; the best strict-small recipe was learning rate 8–9 $\times 10^{-5}$ with seed 44,

while SidBert used 2–3 $\times 10^{-5}$ with seed 42 and Llama used 3×10^{-5} with seed 42.

3 What did not work

Because the negative results consumed most of the project, we summarize them here; Table 2 gives representative numbers. All were evaluated on BLiMP (Warstadt et al., 2020) and BLiMP-Supplement under the official pipelines of their respective years.

Alternative and curriculum corpora. TinyStories (Eldan and Li, 2023) substantially underperformed the standard corpus at 10M words (47–51 BLiMP vs. ≈70). A hand-built Gutenberg corpus ordered by U.S. reading level (grades 1–4 → 5–8 → 9–adult), motivated by curriculum learning, performed no better, and grammar-textbook corpora were worse still. Reordering the corpus or looping epochs within grade bands did not change the result. This is consistent with previous BabyLM findings, in which curriculum learning was widely explored but generally did not improve aggregate performance over non-curriculum baselines (Oba et al., 2023; Hu et al., 2024; Charpentier et al., 2025).

Synthetic data generation. Generating corpus text targeted toward evaluation-like linguistic phenomena, including minimal-pair-style sentences and prompt variations, reached at best 55.7 BLiMP, well below training on the standard corpus. A smaller 1% synthetic augmentation produced a more localized benefit: it improved BLiMP and BLiMP-Supplement to 72.5 and 70.3, respectively, but reduced entity tracking from 28.0 to 22.1 and SuperGLUE from 69.9 to 64.6, lowering the overall NLP average from 53.6 to 52.1. This result should not be interpreted as evidence that synthetic data is uniformly ineffective. Prior work indicates that its value depends strongly on the generation procedure, linguistic diversity, and proportion of natural text (Charpentier et al., 2025; Kang et al., 2025).

In our experiments, targeted synthetic augmentation improved the linguistic phenomena it most closely resembled, but these gains did not generalize across the broader evaluation suite.

Teacher-supervised objectives. A distillation-adjacent scheme in which a teacher reweights the student's loss toward syntactically informative tokens plateaued at 66–67 BLiMP, a few points below the plain GPT-BERT reconstruction, and hybridizing it with nanochat-style (Karpathy, 2025) training schedules did not close the gap (best 65.8/60.0).

Corpus mixing trade-offs. Dialogue-rich custom mixtures (our "Benchmark Target Mix" family) did improve zero-shot suites (+1.7 selective zero-shot average) but cost ≈3.5 points of finetuned SuperGLUE relative to the official corpus at 10M—a trade that loses under the challenge's aggregate. This was the closest any data intervention came to helping, and it still lost to the official corpus.

4 Results

4.1 Strict-small (10M): candidate selection

Table 3 compares the checkpoint-eligible finalists from our development sweeps under MNTP evaluation. The submitted model wins on the challenge-relevant aggregate despite not winning every column: the Benchmark Target Mix-family models buy zero-shot supplement and EWoK (Ivanova et al., 2024) points at the cost of SuperGLUE (Wang et al., 2019), while the official-corpus AdaMuon model has the best Global-PIQA and finetuning behavior. Its Age-of-Acquisition correlation with child CDI data is -0.151 ($p = 0.039$) over the 19 checkpoints.

4.2 GPT-BERT Strict (100M)

Table 4 reports the 100M sweep for GPT-BERT Mixed Focus. The submitted configuration (Muon, lr 0.005) dominates its siblings on every aggregate: +2 finetuned SuperGLUE over the next-best run and the best zero-shot scores of the entire campaign (80.2 BLiMP, 73.6 Supplement). The finetuning learning-rate cliff observed on the sibling run (partial task collapse at 3×10^{-5}) did not appear for this model, whose 2×10^{-5} and 3×10^{-5} recipes tie at 73.4–73.5. Table 4 also shows variant SidBert models: identically-configured, identically-seeded retrains with milestone checkpointing enabled; and Section 5 quantifies the reproduction fidelity we expect from such retrains.

4.3 A stronger causal baseline

The official leaderboard row for `gpt2-baseline-strict` was incomplete, with several missing evaluations represented as zero. We therefore reevaluated the official checkpoint using the BabyLM 2026 strict full-evaluation data, the causal backend, batch size 64, and the `main` model revision. Table 5 compares this

	MurphysLaw	SidBert	Llama
Track	strict-small	strict	strict
Words	10M (official)	100M (official)	100M (official)
Objective	mixed	mixed	causal
Optimizer	AdaMuon	Muon + AdamW	Muon + AdamW
LR / aux LR	0.02 / 0.006	0.005 / 0.007	0.01 / 1e-4
Steps × tok/step	9,914 × 16,384	12,330 × 131,072	13,510 × 131,072
Final weights	tail-avg (20%)	last step	last step
Milestones	19 (1M–100M)	28 (1M–1000M)	28 (1M–1000M)
Eval mode	MNTP	MNTP	causal
FT recipe	8–9e-5, s44	2–3e-5, s42	3e-5, s42

Table 1: Submission configurations. Milestone counts exclude the final model state published on `main`.

Intervention (10M)	BLiMP	Supp.
TinyStories corpus (20 ep)	47.3	45.3
Gutenberg curriculum corpus	≈66	≈55
Synthetic generation (best)	55.7	51.4
Teacher-supervised (best)	67.3	53.0
nanochat schedule (best)	57.0	41.0
Hybrid teacher+nanochat	65.8	60.0
GPT-BERT recon. + AdaMuon	**71.6**	**63.9**

Table 2: Representative development results for data- and objective-centric interventions versus the submitted optimizer-centric recipe (official corpus). Earlier rows use the evaluation pipeline of their year; the final row is the submitted strict-small model under the 2026 pipeline.

completed evaluation with the official Chatty BERT submission.

Chatty BERT improves the seven-suite NLP average by 3.39 points and outperforms GPT-2 on every benchmark category. The largest gains occur on BLiMP, BLiMP-Supplement, and SuperGLUE, suggesting that the modern training recipe particularly improves linguistic competence and downstream transfer. Improvements on EWoK and entity tracking are smaller, and the absolute entity-tracking scores remain low for both models. Thus, strengthening the causal baseline does not remove the broader difficulty of learning persistent entities and world structure from limited text.

The comparison should not be interpreted as isolating a uniquely Llama-specific architectural advantage. Chatty BERT jointly changes the model architecture, normalization and feed-forward design, positional encoding, optimizer, implementation stack, and tuned hyperparameters. Its role is instead to demonstrate that the official GPT-2 result is not a sufficient control for claims about the limits of causal modeling.

5 Evaluation pitfalls

Validation loss does not rank configurations well. We selected hyperparameters by benchmark score, not held-out MLM loss: across a 10M learning-rate grid, validation loss correlates with BLiMP only through broken extremes (Spearman -0.91 overall, -0.40 among the four best), preferring lr 0.005 over the lr 0.02 recipe we submitted, and is uninformative among identically-configured replicas.

Identical retrains are not bit-identical. Our submission process required retraining models with checkpointing enabled, holding configuration, data order, and seed fixed. The retrain reproduced BLiMP to within 0.1 points, but the 82-item BLiMP-Supplement moved by 4.5 points and entity tracking (Kim and Schuster, 2023) by 1.9—consistent with GPU nondeterminism amplified by small suite sizes. Repeating this across 18 identical runs, the seven-suite NLP average spans 56.6–57.9, while evaluation itself is deterministic given fixed weights. Small-suite deltas under a few points should not be read as real effects at these scales, in our submission or anyone else's.

Training and evaluation code, exact configuration files, and launch scripts for the submitted models are available in our public repositories. We note that this repository is designed to run from PyTorch, while the original GPT-BERT code is designed to run on Slurm, so this is another contribution that may be helpful for future competition participants. [1] Because the repository also contains exploratory experiments conducted before and after model selection, we provide a paper-specific reproduction index that maps each reported model to its corresponding training configuration, evaluation recipe, and released weights.

[1] https://github.com/JorgeCMurillo/babylm-baby-steps

GPT-BERT Mixed Focus Model (10M, MNTP)	BLiMP	Supp.	EWoK	Entity	PIQA	SG	Avg.
MurphysLaw (AdaMuon, official)	71.6	63.9	51.9	**28.0**	36.2	69.9†	**53.6**
AdaMuon, MixBlend corpus	71.5	**70.2**	53.5	20.8	33.7	65.4	52.5
Muon, Benchmark Target Mix corpus	70.7	68.5	**56.1**	19.3	34.3	65.2	52.4
Muon, 1%-augmented corpus	**72.5**	70.3	51.2	22.1	31.7	64.6	52.1

Table 3: Strict-small finalists (development sweep, MNTP). SG = finetuned SuperGLUE; Avg. is the mean of the six columns, not including COMPS. †SuperGLUE for the submitted configuration measured on its identically-configured development twin (69.9–70.1 across the 8–9e-5 / seed-44 recipes); official-pipeline finetuning of the submitted weights uses the same recipe.

GPT-BERT Mixed Focus Model (100M)	BLiMP	Supp.	SG	NLP
Muon lr .005 (sub.)	**80.2**	**73.6**	**73.5**	57.3
Muon lr .005 (var. 1)	79.5	71.8	73.0	**57.9**
Muon lr .005 (var. 2)	79.4	70.9	71.6	56.0
Muon lr .01	76.6	62.4	70.3	53.8
AdaMuon lr .02	76.6	64.8	70.1	53.9

Table 4: Strict-track sweep (MNTP; SG = finetuned SuperGLUE average; NLP = six-suite mean). The AdaMuon recipe that wins at 10M loses at 100M, where lower-rate Muon dominates. Additionally, we show identical-training submission variants, displaying the GPU-based nondeterminism inherently present in training and evaluating at this scale (var 1. = identical training process, var. 2 = two-GPU distributed training).

6 Discussion and Limitations

6.1 The importance of strong baselines

Our results support a broader but still qualified conclusion: BabyLM comparisons are highly sensitive to the strength of the baseline. Within GPT-BERT, optimizer choice and learning-rate tuning produced larger and more consistent gains than the multiple data interventions we tested. Across causal decoders, replacing the official GPT-2 reference with a modern Llama–Muon implementation raised the strict NLP average by 3.39 points and improved every individual benchmark.

These findings do not imply that data design or architectural innovation is unimportant. Rather, they show that gains attributed to novelty can be overstated when the comparison is made only against an outdated or undertuned reference implementation. A strong BabyLM control should ideally include both the official baseline and a tuned modern baseline trained under the same data and token restrictions.

6.2 Systems efficiency versus sample efficiency

It is also useful to distinguish statistical sample efficiency from systems efficiency. Liger kernels do not change the number of words observed by the model, and our experiment does not establish that they improve learning per token. Their benefit is practical: reduced memory pressure and improved throughput make Muon training and broader hyperparameter exploration easier under a fixed hardware budget. Because the Llama and GPT-2 systems differ in several respects, a clean architectural ablation would need to hold parameter count, tokenizer, data order, optimizer, token batch, learning-rate schedule, and total training compute constant.

6.3 Limitations

Our exploration of data interventions, though broad, was not exhaustive, and most sweeps used a single seed. The Llama–GPT-2 comparison is a comparison between practical training recipes rather than a controlled architectural ablation: architecture, optimizer, hyperparameters, and implementation differ simultaneously. Small evaluation suites also show substantial run-to-run variation, so differences of only a few points should be interpreted cautiously. We primarily optimized the learning rates. Finally, Age-of-Acquisition correlations are reported as produced by the official pipeline, and we do not attempt to interpret their sign.

Acknowledgments

We thank the BabyLM organizers for providing the evaluation pipeline and baseline models. We thank Zaishuo Xia for computation infra support. We also thank Shawheen Naderi and many other members of Yubei Chen's self-supervised learning group at UC Davis for their helpful discussions and continued support.

Model	BLiMP	Supp.	EWoK	Entity	COMPS	PIQA	SG	NLP avg.
GPT-2 strict	74.73	65.00	54.37	16.91	55.85	36.62	66.06	52.79
Chatty BERT (Llama)	**80.57**	**69.88**	**55.39**	**19.60**	**58.01**	**39.62**	**70.19**	**56.18**
Difference	+5.84	+4.88	+1.02	+2.69	+2.16	+3.00	+4.13	+3.39

Table 5: Completed strict-track comparison between the official BabyLM strict GPT-2 checkpoint and Chatty BERT. NLP average is the unweighted mean of the seven displayed benchmark categories.

References

BabyLM Organizing Committee. 2026. The BabyLM challenge 2026: Call for papers. https://babylm.github.io. Accessed July 2026.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *Proceedings of the BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Ronen Eldan and Yuanzhi Li. 2023. TinyStories: How small can language models be and still speak coherent English? *Preprint*, arXiv:2305.07759.

Michael C. Frank, Mika Braginsky, Daniel Yurovsky, and Virginia A. Marchman. 2017. Wordbank: An open repository for developmental vocabulary data. *Journal of Child Language*, 44(3):677–694.

Pin-Lun Hsu, Yun Dai, Vignesh Kothapalli, Qingquan Song, Shao Tang, Siyu Zhu, Steven Shimizu, Shivam Sahni, Haowen Ning, and Yanning Chen. 2024. Liger Kernel: Efficient Triton kernels for LLM training. *arXiv preprint arXiv:2410.10989*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyurek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Preprint*, arXiv:2405.09605.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/. Blog post; see also the NanoGPT speedrun records.

Feiyang Kang, Newsha Ardalani, Michael Kuchnik, Youssef Emad, Mostafa Elhoushi, Shubhabrata Sengupta, Shang-Wen Li, Ramya Raghavendra, Ruoxi Jia, and Carole-Jean Wu. 2025. Demystifying synthetic data in LLM pre-training: A systematic study of scaling laws, benefits, and pitfalls. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10739–10758, Suzhou, China. Association for Computational Linguistics.

Andrej Karpathy. 2025. nanochat: The best ChatGPT that $100 can buy. https://github.com/karpathy/nanochat. GitHub repository.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. *Preprint*, arXiv:2305.02363.

Miyu Oba, Akari Haga, Akiyo Fukatsu, and Yohei Oseki. 2023. BabyLM challenge: Curriculum learning based on sentence complexity approximating language acquisition. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 290–297, Singapore. Association for Computational Linguistics.

Chongjie Si, Debing Zhang, and Wei Shen. 2025. AdaMuon: Adaptive muon optimizer. *Preprint*, arXiv:2507.11005. ArXiv preprint.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

A Reproduction details

All models were trained on NVIDIA RTX 5090 GPUs. MurphysLaw and Chatty BERT were trained on a single GPU, while SidBert was trained on two GPUs using data parallelism. Submission repositories publish the final weights on `main` and each word-count milestone as a branch named `chck_{N}M`, as required for the challenge's fast checkpoint evaluation and AoA analysis. Finetuning used the official evaluation pipeline's scripts with per-task defaults, batch size 4 at 100M scale to fit memory alongside concurrent jobs, and the learning rates and seeds given in Table 1.

Objective focus and evaluation mode. The sweeps reveal an interaction between training objective, evaluation mode, and scale rather than a uniformly best objective family. At strict-small scale, mixed-focus models evaluated with MNTP generally provided the strongest balance across the zero-shot suites, although the best causal-focus configuration was competitive. At strict scale, the result was less decisive: causal-focus models evaluated with MNTP performed particularly well on EWoK, entity tracking, and COMPS, whereas mixed-focus models evaluated with MNTP attained the strongest BLiMP and BLiMP-Supplement results and performed well under the broader submission evaluation. We therefore selected mixed-focus training with MNTP evaluation for both submitted models. This choice should be interpreted as a submission-level model-selection decision rather than evidence that mixed-focus training uniformly dominates causal-focus training.

B Hyperparameter sweeps for 100M track

Model Family	LR	Mode	Blimp	Blimp SUPP	Ewok	Entity Tracking	Comps	SelfPace R	Eye Tracking
muon-10m-CAUSAL-auxlr0.001	0.001	causal	71.64	61.97	52.43	33.92	53.33	3.92	8.84
muon-10m-CAUSAL-auxlr0.001	0.001	mntp	71.50	64.15	52.99	41.38	53.08	4.40	9.55
muon-10m-CAUSAL-auxlr0.002	0.002	causal	72.20	62.52	51.84	19.57	53.03	3.52	9.06
muon-10m-CAUSAL-auxlr0.002	0.002	mntp	72.01	65.02	53.80	30.80	53.28	3.86	9.72
muon-10m-CAUSAL-auxlr0.003	0.003	causal	71.55	60.93	52.76	24.15	53.38	3.66	9.39
muon-10m-CAUSAL-auxlr0.003	0.003	mntp	70.40	63.07	53.70	30.72	53.28	4.03	10.03
muon-10m-CAUSAL-auxlr0.005	0.005	causal	73.73	59.91	52.39	14.63	52.89	3.53	9.52
muon-10m-CAUSAL-auxlr0.005	0.005	mntp	71.50	62.88	50.79	26.67	52.85	3.67	9.43
muon-10m-CAUSAL-auxlr0.01	0.01	causal	73.51	63.47	52.11	17.34	52.66	4.05	9.82
muon-10m-CAUSAL-auxlr0.01	0.01	mntp	71.57	61.71	51.85	32.77	53.05	4.80	10.26
muon-10m-mixed-auxlr0.001	0.001	causal	69.85	59.51	51.25	27.45	52.58	4.03	9.66
muon-10m-mixed-auxlr0.001	0.001	mntp	72.22	60.74	50.14	40.49	52.99	4.38	10.21
muon-10m-mixed-auxlr0.002	0.002	causal	70.50	56.06	50.93	27.89	53.14	4.38	9.21
muon-10m-mixed-auxlr0.002	0.002	mntp	71.59	59.38	52.08	36.88	53.00	4.43	9.74
muon-10m-mixed-auxlr0.003	0.003	causal	69.82	57.89	51.51	29.10	53.25	3.95	9.74
muon-10m-mixed-auxlr0.003	0.003	mntp	71.72	61.52	52.99	38.29	53.34	4.51	10.49
muon-10m-mixed-auxlr0.005	0.005	causal	69.31	58.59	50.68	29.77	52.78	4.25	9.84
muon-10m-mixed-auxlr0.005	0.005	mntp	72.15	62.75	51.89	43.03	53.77	4.60	10.54
muon-10m-mixed-auxlr0.01	0.01	causal	70.04	60.19	51.46	33.35	52.56	3.68	9.74
muon-10m-mixed-auxlr0.01	0.01	mntp	71.33	62.84	52.86	35.27	52.93	4.01	10.50

Table 6: In the Strict Small 10M track, Hyperparameter sweep across different learning rates, before entity tracking was updated to remove the non-nothing , and the addition of hidden task GlobalPiqa. We retain the original results because they were the measurements available when the learning rates and model configurations were selected.

LR	BLiMP	BLiMP Supp.	EWoK	Entity Tracking	COMPS	SelfPace R	Eye Tracking
Model focus: Causal | `eval=causal`							
0.00025	75.53	62.69	53.21	31.98	55.92	3.52	10.17
0.0025	78.13	70.22	56.75	32.98	58.14	3.39	9.08
0.005	78.74	69.62	56.60	29.38	58.15	3.18	8.69
0.01	79.95	68.75	57.89	34.82	58.65	3.01	8.16
0.02	79.45	67.88	56.45	30.92	58.39	3.36	8.70
Model focus: Mixed | `eval=causal`							
0.00025	73.41	58.84	52.47	40.22	54.91	3.04	10.06
0.0025	77.43	65.98	53.25	28.34	56.68	3.54	9.33
0.005	78.62	67.38	56.24	33.08	56.92	3.12	9.35
0.01	78.52	63.82	55.59	31.88	57.51	3.26	9.27
0.02	78.06	63.95	54.36	27.99	56.83	3.26	8.83
Model focus: Causal | `eval=mntp`							
0.00025	73.25	62.25	53.62	30.36	55.29	3.52	9.86
0.0025	77.67	69.13	56.31	41.57	58.28	3.99	9.81
0.005	78.76	70.45	57.27	36.46	58.44	3.23	9.23
0.01	79.43	70.65	59.09	44.13	59.47	3.47	9.11
0.02	78.77	71.26	57.38	42.39	59.01	3.62	9.58
Model focus: Mixed | `eval=mntp`							
0.00025	75.32	62.93	52.75	37.00	54.73	3.17	10.18
0.0025	79.50	73.26	54.72	40.81	57.52	3.27	9.43
0.005	80.23	73.62	56.25	41.46	58.72	3.26	9.45
0.01	79.90	73.63	57.48	41.00	59.01	3.14	9.63
0.02	79.16	71.33	55.26	44.59	59.40	3.61	10.33

Table 7: GPT-BERT Learning-rate sweep results for the BabyLM 100M track, before the entity tracking update and addition of the Global PIQA task. We retain the original results because they were the measurements available when the learning rates and model configurations were selected.

Learning What Can Combine and What Can Substitute: Relational Inductive Biases for BabyLM Pretraining

Yuwei Yang[1] **Shihao Pu**[1] **Gefei Feng**[1,2*] **Yiming Yang**[1,2*]
[1]School of Linguistic Sciences and Arts, Jiangsu Normal University, Xuzhou, Jiangsu, China
[2]Collaborative Innovation Center for Language Ability, Jiangsu Key Laboratory of Brain Cognition and Language Rehabilitation, Jiangsu Normal University, Xuzhou, China
{2020240930, ausar_pu, fspeed, yangym}@jsnu.edu.cn

Abstract

Language can produce many expressions from a limited set of units because linguistic units participate in two basic relations. Syntagmatic relations determine which units can combine, while paradigmatic relations determine which units can substitute for one another in the same position. We study whether explicitly learning these relations helps language models generalize from limited data. We introduce two auxiliary objectives for masked language modeling. The Syntagmatic objective aligns a masked representation with relevant context selected by attention, while the Paradigmatic objective aligns it with the embedding neighbors of the target token. Both objectives are generated by the model itself and require no additional data, annotation, or trainable parameters. Experiments on BabyLM 2026 Strict-Small show that both single-relation systems improve mean Overall performance over Standard MLM, with Paradigmatic giving the most consistent gains across three seeds. Relay Joint combines the two objectives and shows a strong neighborhood-alignment signal at its submitted checkpoint, although its downstream gains vary across seeds. These results suggest that relational objectives can provide useful learning signals under limited data, while stronger internal alignment does not always lead to more stable downstream performance. Code is now publicly available on GitHub.

1 Introduction

Language is an economical, efficient, and flexible hierarchical symbolic system. A finite inventory of linguistic units supports a virtually unbounded range of expressions because these units are not isolated: they participate together in syntagmatic relations of combination and paradigmatic relations of choice. Syntagmatic relations determine which units can combine to form larger structures (Saussure, 1916; Firth, 1957), whereas paradigmatic relations organize the alternatives that may occupy the same structural position (Saussure, 1916; Halliday, 1961; Harris, 1954). Together, these axes support the hierarchical organization, productivity, and generalization capacity of language.

This organization suggests a hypothesis for learning from limited input. Learners cannot memorize every possible expression; they must abstract structures and categories that transfer beyond observed instances (Tomasello, 2003; Bybee, 2010). Combination relations provide regularities over how units form larger patterns, while choice relations organize categories of fillers for a shared position. If a language model is explicitly encouraged to capture both kinds of regularity, it may rely less on memorizing individual expressions and instead form reusable structures and categories. We refer to this proposal as the **Relational Efficiency Hypothesis**. It is a research hypothesis about fixed-exposure generalization, not a claim of computational speed.

Masked language modeling (MLM) (Devlin et al., 2019) already offers implicit opportunities to learn both relations: predicting a missing token requires sensitivity to context and lexical alternatives. However, token-level cross-entropy directly supervises only the identity of the ground truth token. It does not explicitly require the masked representation to retain which contextual units support combination at that position or which lexical units approximate a potential choice class. We do not assume that standard MLM lacks these properties; rather, we test whether explicit relational representation targets provide a useful inductive bias when input is limited (Warstadt et al., 2023; Hu et al., 2024).

We operationalize the two axes through detached, model-internal targets at MLM-selected positions. Syntagmatic uses last-layer attention to construct an attention-weighted target from eligible visible context, approximating contextual support.

*Corresponding authors.

Proceedings of the Second BabyLM Workshop, pages 831–842
October 29, 2026 ©2026 Association for Computational Linguistics

Paradigmatic uses the current input-embedding neighbors of the ground truth token to construct a neighborhood centroid, approximating substitutional choice. Neither objective adds corpus data, external annotation, or trainable parameters. We compare the two single-relation systems and Relay Joint across seeds 0, 1, and 42 under the same backbone, corpus, masking process, and exposure budget. Paradigmatic achieves the highest and most consistent mean Overall result. The submitted Relay checkpoint is the strongest individual checkpoint among the four primary systems, but its advantage does not persist across seeds. The results are therefore task- and seed-dependent and do not establish universal synergy.

Our contributions are:

1. **Theory.** The Relational Efficiency Hypothesis connects syntagmatic–paradigmatic organization to language generalization under limited exposure.

2. **Method.** Two self-generated, detached, and parameter-neutral relational targets approximate contextual combination and substitutional choice at MLM-selected positions, together with the staged Relay Joint schedule used by the submitted checkpoint.

3. **Evidence.** We provide a three-seed, seed-matched downstream comparison, together with matched checkpoint trajectories for seed 0 and a 2,000-item representation analysis over 31 target-token-ID clusters. The latter uses clustered-bootstrap uncertainty and should not be interpreted as across-seed variance.

2 Related Work

2.1 Syntagmatic and Paradigmatic Organization

Saussure (1916) distinguished syntagmatic relations among units combined in sequence from associative, later paradigmatic, relations among alternatives *in absentia*. Halliday (1961) recast this contrast as structure (chain) and system (choice), while Harris (1954) used distributional equivalence to characterize substitution classes. Sahlgren (2008) connected the distinction to distributional representation: direct co-occurrence tends to emphasize syntagmatic association, whereas similarity of contexts tends to emphasize paradigmatic association.

Usage-based and construction-based theories further explain how repeated instances can support schemas with structurally constrained slots (Tomasello, 2003; Bybee, 2010; Goldberg, 2006). Word-association studies document related contrasts in human responses (Brown and Berko, 1960; Ervin, 1961; Entwisle et al., 1964). These traditions motivate the relational axes, but our continuous targets are model-internal approximations rather than symbolic linguistic structures. The unresolved question is whether both axes can be operationalized as useful inductive biases for pretraining from limited exposure.

2.2 Relational Signals in Language Models

Sun et al. (2015) modeled syntagmatic and paradigmatic relations in static embeddings, while Mickus (2024) interpreted MLM through the paradigmatic axis. Lexical-substitution work instead generates or ranks context-appropriate substitutes with contextual models (Melamud et al., 2016; Zhou et al., 2019) or human annotations (Lee et al., 2021). Our objective neither generates substitutes nor uses labeled substitute sets; it treats the current input-embedding neighborhood as a distributional approximation.

Attention analyses provide another source of relational evidence: individual heads can correlate with syntactic relations (Clark et al., 2019), although attention is not a guaranteed explanation or syntactic ground truth (Jain and Wallace, 2019). Similarly, distributional co-occurrence explains much but not all MLM generalization (Sinha et al., 2021; Chiang and Yogatama, 2023). Unlike post-hoc analyses, both objectives here are generated from internal model signals and train MLM-selected contextual hidden states. The gap we address is whether these approximate signals can serve as training biases rather than merely describe representations after training.

2.3 Data-Constrained Pretraining

The BabyLM challenge (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025) tests language learning under tightly constrained input. Existing work has improved such training through architectural changes (Huebner et al., 2021), alternative or combined training objectives (Charpentier and Samuel, 2024), adaptive masking strategies (Edman and Fraser, 2025), and data organization (Haga et al., 2024). These approaches show several routes to stronger limited-data models, but do not isolate

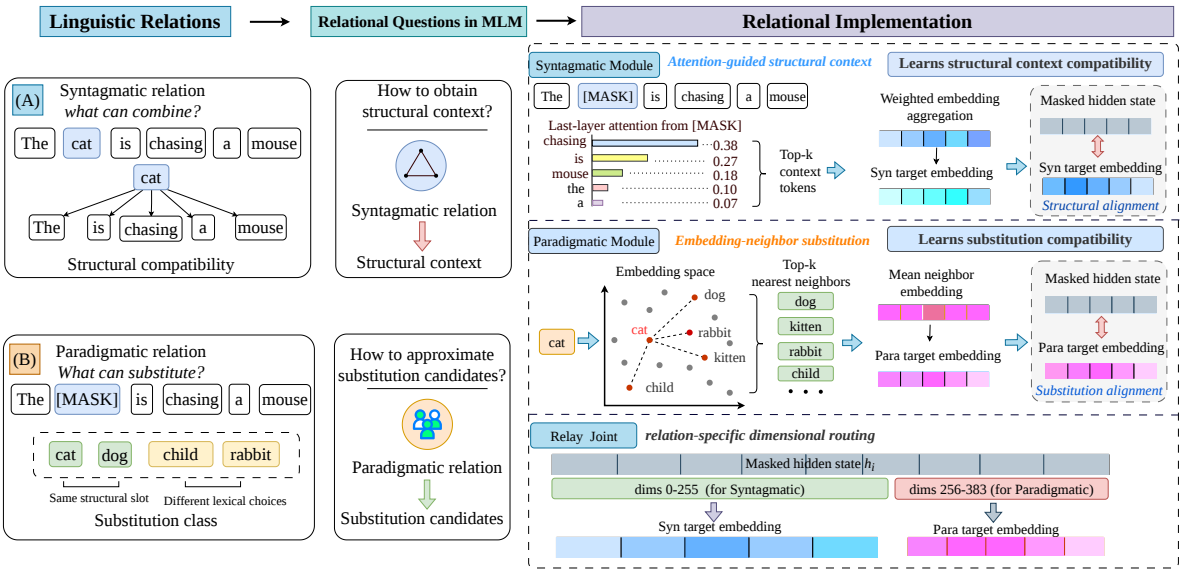


Figure 1: Overview of the relational pretraining framework. The four systems share the backbone, corpus, MLM corruption process, optimizer, and exposure budget, while differing in their relational objectives and how these objectives are integrated during training.

the effect of the two relational axes on the representation at a masked position. Our comparison therefore fixes the corpus, corruption process, backbone, and cumulative exposure, changing only the detached relational targets applied to MLM-selected hidden states.

3 Method

Let x_i be the ground truth token at an MLM-selected position i, and let $\mathbf{h}_i \in \mathbb{R}^{384}$ be its final-layer contextual hidden state before the MLM prediction head. Standard MLM minimizes token cross-entropy $\mathcal{L}_{\text{MLM}}$. Both relational objectives construct a target from signals already available inside the model and apply stop-gradient to that target. Consequently, an auxiliary target cannot move simply to reduce its own loss; the relational gradient updates the model through the contextual hidden-state branch. Neither objective requires new training data or introduces additional trainable parameters.

3.1 Syntagmatic Alignment

Syntagmatic relations concern how linguistic units combine within an expression. We approximate this relation by asking which visible context positions the model's final attention layer treats as support for an MLM-selected position. We average the last-layer attention weights across all 12 heads. After excluding the target position, padding, `[CLS]`, `[SEP]`, `[MASK]`, and every other MLM-selected position, we select the top $K = 8$ remaining positions. Let $\mathcal{A}_i$ denote this set. We apply a softmax to the selected attention values to obtain normalized weights $\widetilde{\alpha}_{ij}$ over $\mathcal{A}_i$. Using embeddings of the original, uncorrupted input tokens, the target and loss are

$$\mathbf{s}_i = \text{sg}\left(\sum_{j \in \mathcal{A}_i} \widetilde{\alpha}_{ij} \mathbf{e}(x_j)\right), \qquad (1)$$
$$\ell_{\text{syn},i} = 1 - \cos(\mathbf{h}_i, \mathbf{s}_i).$$

where sg denotes stop-gradient.

The loss acts on the complete 384-dimensional hidden state. It is weighted by $\lambda_{\text{syn}} = 5 \times 10^{-4}$ and becomes active after approximately 10% of training. MLM-selected entity tokens and tokens deemed invalid by the fixed vocabulary lookup do not receive this target. This target approximates prediction-relevant contextual support; it is not an explicit dependency representation.

3.2 Paradigmatic Alignment

Paradigmatic relations concern the alternatives that can occupy a shared structural position. Because the training data provide no explicit substitution sets, we approximate this relation using the ground truth token's neighborhood in the model's current input embedding space. For each eligible MLM-selected position, we select the $K = 8$ vocabu-

lary items with the highest cosine similarity to the ground truth token in the current input-embedding space, excluding the ground truth token itself. We denote this set by $\mathcal{N}_i$.

The target and loss are

$$\mathbf{p}_i = \mathrm{sg}\left(\frac{1}{K}\sum_{v\in\mathcal{N}_i}\mathbf{e}(v)\right), \qquad (2)$$
$$\ell_{\mathrm{para},i} = 1 - \cos(\mathbf{h}_i, \mathbf{p}_i).$$

The neighbor centroid is aligned with the full 384-dimensional hidden state; no content projection is used. For syntax tokens, the loss weight is 5×10^{-5} and activation begins after approximately 55% of training. For content tokens, the weight is 10^{-6} and activation begins at 99%, so the loss is active only during the final 1%. Entity and reading-sensitive tokens do not receive a Paradigmatic loss. The fixed vocabulary categories and their lookup priority are detailed in Appendix A. Embedding neighborhoods are distributional approximations to potential alternatives, not guaranteed grammatical substitutes in a particular context.

3.3 Relay Joint Objective

The formal submission model combines the two targets with a staged, fixed-subspace schedule. The Syntagmatic loss is applied to dimensions 0–255 of the contextual state and target, while the Paradigmatic loss is applied to dimensions 256–383. These are fixed coordinate slices, not learned projections. Standard MLM continues to use the complete hidden state, and the auxiliary design adds no trainable parameters.

Syntagmatic supervision begins after approximately 10% of optimization with weight 5×10^{-4}. Syntax-token Paradigmatic supervision begins after approximately 20% with weight 5×10^{-5}. From that point onward, the Syntagmatic per-position weight on syntax tokens is retained at 10% of its earlier strength rather than removed. Content-token Paradigmatic supervision has weight 10^{-6} and begins at 99%, so it is active only in the final 1% of training. Entity and reading-sensitive tokens receive no Paradigmatic loss, and entity tokens receive no Syntagmatic loss.

Relay Joint also applies an entity representation anchor to eligible MLM-selected entity tokens from approximately 10% to 60% of training, using the same cosine-alignment form, $1 - \cos(\mathbf{h}_i, \mathrm{sg}(\mathbf{e}(x_i)))$, between the contextual state and the detached ground truth token embedding. The anchor uses the complete 384-dimensional hidden state and has weight 2.5×10^{-4}.

Method	Syn	Para	Rel. rep.	Ent. anchor
Standard MLM	No	No	–	No
Syntagmatic	Yes	No	Full 384	No
Paradigmatic	No	Yes	Full 384	No
Relay Joint	Yes	Yes	Syn 256 / Para 128	Yes

Table 1: The four primary systems. Relay Joint uses fixed slices 0–255 for Syntagmatic alignment and 256–383 for Paradigmatic alignment; MLM prediction always uses the full state.

In Relay Joint, $\mathcal{L}_{\mathrm{syn}}$ uses dimensions 0–255 and both Paradigmatic losses use dimensions 256–383. Each loss is averaged over eligible positions; $g(t)$ is its progress gate. The Relay Joint objective is

$$\begin{aligned}\mathcal{L}(t) = \mathcal{L}_{\mathrm{MLM}} &+ \lambda_{\mathrm{syn}} g_{\mathrm{syn}}(t)\mathcal{L}_{\mathrm{syn}} \\ &+ \lambda_{\mathrm{para,syn}} g_{\mathrm{para,syn}}(t)\mathcal{L}_{\mathrm{para,syn}} \\ &+ \lambda_{\mathrm{para,cont}} g_{\mathrm{para,cont}}(t)\mathcal{L}_{\mathrm{para,cont}} \\ &+ \lambda_{\mathrm{ent}} g_{\mathrm{ent}}(t)\mathcal{L}_{\mathrm{ent}}. \end{aligned} \qquad (3)$$

No content projection, learned router, extra corpus, or additional trainable module is enabled. The single-relation systems retain their original full-384 losses; Table 1 therefore summarizes implementation differences rather than a 2×2 factorial identity.

4 Experimental Setup

4.1 Data and Training

All configurations use the official BabyLM 2026 Strict-Small corpus released on Hugging Face[1] and the shared setup in Table 2 (Choshen et al., 2026). The corpus contains approximately 10M words. Models train for 10 epochs, with cumulative exposure no greater than 100M words. We evaluate seeds 0, 1, and 42 for every primary system. Seed 0 was designated as the submission seed before the final three-seed results were available, and it is also used in the matched checkpoint analyses. All systems have 34,677,952 parameters. Parameter-neutral means that the auxiliary objectives introduce no trainable parameters, although they do add target-construction and loss-computation cost.

4.2 Evaluation and Reporting

We evaluate models using the BabyLM 2026 evaluation pipeline[2] (Choshen et al., 2026). Conven-

[1] BabyLM 2026 Strict-Small

[2] Evaluation pipeline commit 3d57ddc8c40e.

Setting	Value
Backbone	DeBERTa-v2 (He et al., 2021), 12 layers, 12 heads
Hidden / FFN size	384 / 1280
Tokenizer	SentencePiece, 40,000 tokens
Parameters	34,677,952
Sequence length	64 (epochs 1–5); 256 (epochs 6–10)
MLM probability	Linear $0.40 \rightarrow 0.15$
Replacement	80% mask, 10% random, 10% unchanged
Optimizer	LAMB (You et al., 2020)
Peak learning rate	0.007
Schedule	1% warmup, cosine decay
Batch size	256

Table 2: Shared backbone and training configuration.

tional NLP Performance includes BLiMP, BLiMP Supplement, EWoK, Entity Tracking, COMPS, GlobalPIQA, and (Super)GLUE, while Humanlikeness and Development includes Reading and Age of Acquisition (AoA).

We follow the metrics implemented by this pipeline. Entity Tracking scores whether the correct continuation receives the highest probability. GlobalPIQA averages its parallel and non-parallel subsets. (Super)GLUE macro-averages BoolQ, MultiRC, RTE, WSC, MRPC, QQP, and MNLI; MRPC and QQP use binary F1, and the remaining tasks use accuracy. Reading is 100 times the mean normalized ΔR^2 for self-paced reading and eye tracking. AoA is 100 times the Pearson correlation between model-implied and child acquisition ages, with a score of zero when $p > 0.1$.

NLP Average is the arithmetic mean of the seven Conventional NLP metrics, Human Average is the arithmetic mean of Reading and AoA, and Overall is the unweighted arithmetic mean of all nine top-level metrics. Results are means with sample standard deviations over seeds 0, 1, and 42; comparisons with Standard MLM use same-seed paired differences. With three pairs, these differences are reported descriptively rather than as significance tests. Checkpoint analyses use matched runs with seed 0.

5 Results

5.1 Overall Performance and Across-Seed Variation

Tables 3 and 4 show that the two single-relation systems are more consistent across seeds than Relay Joint. Paradigmatic has the highest mean Overall and exceeds its matched Standard MLM result in all three seeds. Syntagmatic also has a higher mean Overall, although its matched difference is positive in two of the three seeds. Relay Joint varies more: its submitted checkpoint trained with seed 0 is the strongest individual checkpoint among the four primary systems, but the advantage does not persist across seeds. The results therefore support the two relational targets, especially Paradigmatic, but do not show that the staged Relay integration is generally better.

5.2 Transfer Profiles and Integration Trade-offs

The overall averages hide clear differences across tasks. Relay Joint performs best on several grammar- and knowledge-related evaluations, whereas Paradigmatic provides the most consistent Overall improvement across the three seeds. No system improves every task family. The relational objectives therefore change which tasks benefit rather than producing uniform gains.

Table 5 shows a similar pattern across the integration variants. Clean Union has the highest mean NLP Average. It directly combines the unchanged single-relation recipes in the full 384-dimensional state, including their original activation gates, whereas Simultaneous Joint uses a separately specified simultaneous schedule. Adding routing, entity anchoring, or staging does not consistently improve performance. These comparisons therefore do not identify any component as necessary and do not show that the Relay schedule is generally preferable.

Method	BLiMP	B-Supp	EWoK	Entity	COMPS	GlobalPIQA	(Super)GLUE	NLP Avg
Standard MLM	66.95±1.13	58.54±1.07	50.83±0.49	21.38±1.02	51.96±0.16	**37.44**±2.05	65.06±0.81	50.31±0.13
Syntagmatic	66.88±0.72	60.67±1.46	50.69±1.43	21.21±0.40	52.19±0.34	36.79±2.34	**65.78**±1.52	50.60±0.29
Paradigmatic	67.14±0.39	60.43±1.66	51.59±0.61	**21.58**±1.05	**52.22**±0.33	37.10±2.16	65.63±0.29	**50.81**±0.27
Relay Joint	**67.57**±1.18	**61.56**±3.18	**52.32**±0.75	20.51±1.33	52.14±0.08	34.34±2.82	65.60±0.77	50.58±1.07

Table 3: Conventional NLP Performance, mean ± sample SD over three seeds. Bold marks the highest mean in each column.

Method	Reading	AoA	Human Avg	Overall	Paired Δ vs. Std.	Positive seeds
Standard MLM	5.69±0.39	**0.00**±0.00	2.84±0.19	39.76±0.08	–	–
Syntagmatic	5.85±0.57	**0.00**±0.00	2.92±0.28	40.01±0.28	+0.25±0.35	2/3
Paradigmatic	**6.00**±0.16	**0.00**±0.00	**3.00**±0.08	**40.19**±0.21	+0.43±0.18	3/3
Relay Joint	5.94±0.12	**0.00**±0.00	2.97±0.06	40.00±0.82	+0.24±0.88	1/3

Table 4: Human-likeness and Overall, mean ± sample SD over three seeds. Paired differences are computed against the matched Standard MLM seed from unrounded Overall values. Overall is the unweighted arithmetic mean of all nine top-level metrics.

Configuration	Obj.	R	A	Ret.	Integration	NLP Avg
Standard MLM	None	No	No	No	–	50.31±0.13
Simultaneous Joint	Both	No	No	No	Simultaneous	50.10±0.75
Routing Only	Both	Yes	No	No	Simultaneous	50.60±0.43
Entity Anchor Only	Both	No	Yes	No	Simultaneous	50.70±0.92
Routing + Entity Anchor	Both	Yes	Yes	No	Simultaneous	50.65±0.12
Clean Union	Both	No	No	No	Direct union	**50.88**±0.37
Staged Joint	Both	Yes	Yes	No	Staged	50.55±0.39
Relay Joint	Both	Yes	Yes	10%	Staged relay	50.58±1.07

Table 5: Integration comparison over three seeds. Obj. indicates active relational objectives; R, A, and Ret. denote routing, entity anchoring, and Syntagmatic retention. NLP Average covers the seven Conventional NLP metrics and is not Overall. Staged Joint and Relay Joint both use relation-specific routing and entity anchoring; Relay Joint additionally retains the Syntagmatic objective at 10% strength after syntax-level Paradigmatic supervision begins.

5.3 Developmental Trajectories

Figure 2 separates the learning biases more clearly than it ranks the systems. The trajectories differ substantially across objectives, but no system dominates consistently across measures or exposure levels. Changes in relative position across checkpoints make the curves descriptive evidence of relation-specific learning dynamics, not of uniformly accelerated development under one integration strategy.

These downstream results motivate a narrower question: whether the relational objectives leave distinguishable signatures in the learned representations even when aggregate transfer varies across seeds.

6 Representation Analysis

Downstream performance does not identify the mechanism behind a score change. We therefore use neighborhood retrieval to test the Paradigmatic signal and context-removal and dependency diagnostics to test the Syntagmatic signal. These probes measure target-related properties rather than general model quality; complete definitions and results appear in Appendix C.

6.1 Paradigmatic Neighborhood Structure

The probe asks whether a masked hidden state retrieves the correct ground truth token neighborhood among 64 candidates, using 2,000 shared items grouped by 31 target-token IDs. Figure 3 shows that all three relationally trained models move closer to the appropriate neighborhood in the full representation. Relay Joint has the largest change

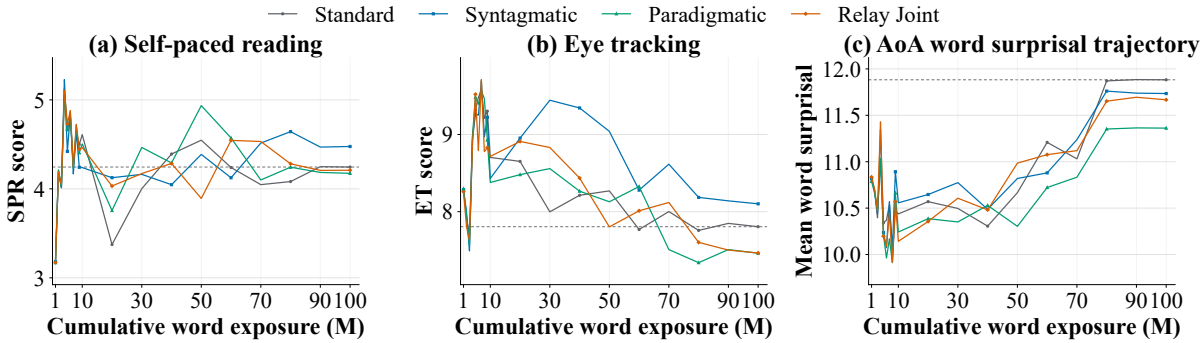


Figure 2: Matched Reading and AoA trajectories for seed 0. Higher is better for self-paced reading (SPR) and eye tracking (ET); lower is better for AoA target-word surprisal. Dashed lines mark the Standard MLM 100M reference. Curves are single runs and include no cross-run uncertainty bands. The AoA trajectory reports target-word surprisal and is distinct from the final AoA correlation score used in Table 4.

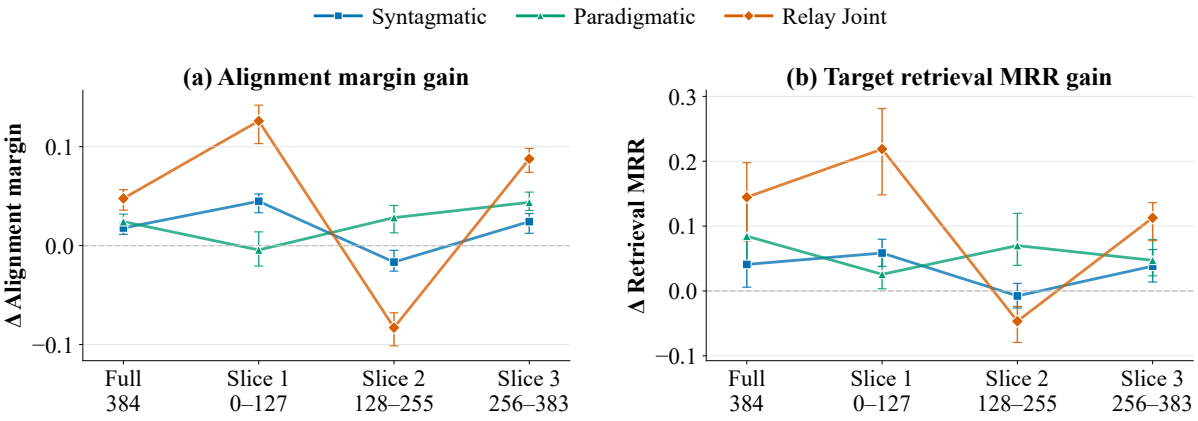


Figure 3: Paradigmatic neighborhood alignment for 2,000 shared items grouped into 31 target-token-ID clusters, reported as paired differences from Standard MLM. Points are item-macro paired means; bars are 95% intervals from 10,000 cluster-bootstrap resamples using target token ID as the resampling unit. Slices 1–2 cover dimensions 0–255 and Slice 3 covers dimensions 256–383. The intervals condition on matched checkpoints for seed 0 and do not measure variation across training seeds.

at the submitted checkpoint, improving alignment margin by $+0.0477$ and MRR by $+0.1445$ over Standard MLM. Both 95% cluster-bootstrap intervals remain above zero when target token ID is used as the resampling unit. These intervals describe the matched checkpoints for seed 0 only; they do not show that the same effect holds across training seeds.

6.2 Dimensionally Selective Reorganization

Relay Joint improves Slices 1 and 3 but reduces performance in Slice 2 relative to Standard MLM. Slices 1 and 2 both receive the Syntagmatic objective, yet they change in opposite directions. The hidden state therefore does not show a clean split between Syntagmatic and Paradigmatic information. Because these are fixed coordinate slices, the analysis also does not show that routing caused the observed pattern.

6.3 From Representation to Transfer

Removing attention-selected positions harms ground truth token prediction more than removing matched controls, indicating that selected context carries prediction-relevant information. The Syntagmatic model shows a larger average change, but with notable cross-seed fluctuation. Dependency recall shifts only slightly and precision is essentially unchanged. This is an independent Standard–Syntagmatic diagnostic, not mechanism evidence for Relay Joint.

The clearest representation and transfer results do not coincide. Paradigmatic gives the most consistent Overall gain across the three seeds, whereas Relay Joint has the strongest neighborhood signature at the submitted checkpoint but variable downstream performance across seeds. Strong representational signals are therefore not sufficient to guarantee stable downstream generalization. These probes identify targeted internal properties relevant to the relational objectives and should not be treated as overall model-quality scores.

7 Conclusion

We introduced two self-generated relational targets without additional data, annotation, or trainable parameters. Across three seeds, both single-relation systems yield higher mean Overall scores than Standard MLM, with Paradigmatic providing the highest and most consistent result. Relay Joint is less stable downstream but shows the strongest neighborhood alignment among the four primary systems at the submitted checkpoint. Its first and third slices improve while its second slice declines, so the results do not show a clean split between Syntagmatic and Paradigmatic information. Overall, relational supervision can help under limited exposure, but learning a relation and using it reliably for downstream tasks remain distinct challenges.

8 Limitations

Scope. We evaluate English BabyLM 2026 Strict-Small, one 34.7M-parameter DeBERTa-v2 architecture, and three seeds. Three seed-matched comparisons are insufficient for strong significance or broad generalization.

Relational approximations. Attention is not explicit syntax, embedding neighbors need not be genuine substitutes, and fixed English vocabulary categories are approximate. The fixed coordinate partition is basis-dependent and imposed rather than discovered.

Result boundaries. Directions vary across tasks and seeds; Relay Joint is positive against Standard for one of three seeds and trails Paradigmatic in mean Overall. The clustered bootstrap covers 31 syntax-token target IDs and conditions on one checkpoint per method. It controls repeated targets but not across-seed uncertainty or causality.

References

Roger Brown and Jean Berko. 1960. Word association and the acquisition of grammar. *Child Development*, 31(1):1–14.

Joan Bybee. 2010. *Language, Usage and Cognition*. Cambridge University Press, Cambridge.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Ting-Rui Chiang and Dani Yogatama. 2023. The distributional hypothesis does not fully explain the benefits of masked language model pretraining. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10305–10321, Singapore. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop. *Preprint*, arXiv:2602.20092.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Doris R. Entwisle, Daniel F. Forsyth, and Rolf Muuss. 1964. The syntactic-paradigmatic shift in children's word associations. *Journal of Verbal Learning and Verbal Behavior*, 3(1):19–29.

Susan M. Ervin. 1961. Changes with age in the verbal determinants of word-association. *The American Journal of Psychology*, 74(3):361–372.

John Rupert Firth. 1957. A synopsis of linguistic theory, 1930–1955. In *Studies in Linguistic Analysis*, pages 1–32. Blackwell, Oxford.

Adele E. Goldberg. 2006. *Constructions at Work: The Nature of Generalization in Language*. Oxford University Press, Oxford.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM challenge: Exploring the effect of variation sets on language model training efficiency. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning (CoNLL)*, pages 252–261, Miami, FL, USA. Association for Computational Linguistics.

M. A. K. Halliday. 1961. Categories of the theory of grammar. *Word*, 17(2):241–292.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations (ICLR)*.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Philip A. Huebner, Elior Sulem, Cynthia Fisher, and Dan Roth. 2021. BabyBERTa: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning (CoNLL)*, pages 624–646. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Mina Lee, Chris Donahue, Robin Jia, Alexander Iyabor, and Percy Liang. 2021. Swords: A benchmark for lexical substitution with improved data coverage and quality. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4362–4379. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

Timothee Mickus. 2024. Language models and the paradigmatic axis. In *Proceedings of the Society for Computation in Linguistics 2024*, pages 63–74, Irvine, CA. Association for Computational Linguistics.

Magnus Sahlgren. 2008. The distributional hypothesis. *Rivista di Linguistica*, 20(1):33–53.

Ferdinand de Saussure. 1916. *Cours de linguistique générale*. Payot, Lausanne and Paris.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2888–2913. Association for Computational Linguistics.

Fei Sun, Jiafeng Guo, Yanyan Lan, Jun Xu, and Xueqi Cheng. 2015. Learning word representations by jointly modeling syntagmatic and paradigmatic relations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 136–145, Beijing, China. Association for Computational Linguistics.

Michael Tomasello. 2003. *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press, Cambridge, MA.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 1–34, Singapore. Association for Computational Linguistics.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations (ICLR)*.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3368–3373, Florence, Italy. Association for Computational Linguistics.

A Training Configurations

Both targets use $K = 8$. Syntagmatic uses the last attention layer with weight 5×10^{-4}. Syntax Paradigmatic uses 5×10^{-5}; content Paradigmatic uses 10^{-6}. Relay Joint's full-state anchor has weight 2.5×10^{-4}. No relational component adds trainable parameters.

Vocabulary-derived token categories. No POS tagger or NER system is used. The tokenizer vocabulary is scanned once before training. One recognized leading SentencePiece, byte-level BPE, WordPiece, or space marker is removed; trailing punctuation is stripped; and fixed-list matching is case-insensitive. Byte tokens and entries with no alphabetic character are invalid. The lookup then assigns the first matching category in this order: syntax, reading-sensitive, entity, and content. Thus the categories are mutually exclusive even when lexical lists overlap. An entity is a remaining entry containing a digit, or an alphabetic entry of at least three characters that begins with an uppercase letter. Remaining alphabetic entries are content tokens.

Fixed lexical lists. **Auxiliaries:** is, are, was, were, be, been, being, am, do, does, did, have, has, had, can, could, will, would, should, may, might, must, shall, ought, need, dare, used. **Determiners:** a, an, the, this, that, these, those, some, any, all, each, every, many, few, no, much, more, most, several, both, either, neither, such, what. **Pronouns:** he, she, it, they, him, her, them, his, their, its, we, us, our, you, your, i, me, my, myself, yourself, himself, herself, itself, ourselves, themselves, one, ones. **Negation:** not, n't, never, no, nor, neither. **Prepositions:** in, on, at, by, with, from, to, of, for, into, onto, over, under, near, about, between, through, during, without, within, along, across, behind, beyond, toward, towards, upon, among, amongst, beside, besides, against, around, before, after, above, below, off, up, down, out. **Conjunctions:** and, or, but, because, although, if, when, while, before, after, since, until, unless, whereas, so, yet, than, as, though, whether, once, till, lest, except, provided, given, suppose, assuming, whenever, wherever. **Wh-words:** who, whom, whose, which, where, why, how, what, whatever, whichever, whoever, however, wherever. **Comparatives:** more, less, fewer, better, worse, bigger, smaller, higher, lower, longer, shorter, older, younger. **Temporal expressions:** now, then, ago, later, earlier, soon, already, still, yet, finally, eventually, previously, formerly, currently, recently, lately, immediately, suddenly, gradually.

Ground truth token and context handling. Target-position eligibility is looked up from the clean ground truth token ID, never the corrupted input. The same category therefore applies whether an MLM-selected token is replaced with `[MASK]`, a random token, or left unchanged. The Syntagmatic context set excludes padding, `[CLS]`, `[SEP]`, `[MASK]`, the target itself, and every other MLM-selected position. Context embeddings are then gathered from the original uncorrupted token IDs. Subword entries are classified independently; the code performs no word reconstruction.

Method	Syn onset	Syntax Para	Content Para	Representation	Anchor	Retention
Standard MLM	Off	Off	Off	–	Off	Off
Syntagmatic	∼10%	Off	Off	Full 384	Off	Off
Paradigmatic	Off	∼55%	final 1%	Full 384	Off	Off
Relay Joint	∼10%	∼20%	final 1%	Syn 256 / Para 128	∼10–60%	10% Syn

Table 6: Configuration specification. All systems have 34,677,952 parameters. Routing uses fixed coordinates, not learned projections.

Objective	Eligible MLM-selected positions
Syntagmatic	Syntax, reading-sensitive, or content category; entity and invalid tokens are excluded.
Paradigmatic, syntax	Fixed syntax-list match; active after about 55% alone and about 20% in Relay Joint.
Paradigmatic, content	Content category; active only in the final 1%. Entity and reading-sensitive categories have zero weight.
Entity anchor	Relay Joint entity category; active from about 10% through 60%.

Table 7: Ground truth token eligibility. Categories are mutually exclusive fixed token-ID lookups built before training.

B Per-seed and Component Results

Method	Seed	BLiMP	B-Supp	EWoK	Entity	COMPS	GlobalPIQA	(S)GLUE	Reading	AoA	Overall
Standard MLM	0	66.80	58.57	51.39	22.53	51.90	35.13	64.82	6.03	0.00	39.69
Standard MLM	1	68.15	57.46	50.60	21.01	52.14	39.06	64.39	5.76	0.00	39.84
Standard MLM	42	65.90	59.60	50.49	20.59	51.85	38.11	65.96	5.27	0.00	39.75
Syntagmatic	0	67.56	60.37	49.10	20.84	52.50	38.62	67.52	6.29	0.00	40.31
Syntagmatic	1	66.12	59.38	51.10	21.64	51.82	37.61	65.00	5.21	0.00	39.76
Syntagmatic	42	66.97	62.26	51.88	21.16	52.26	34.15	64.81	6.04	0.00	39.95
Paradigmatic	0	67.13	62.30	52.28	21.67	52.22	34.62	65.78	5.81	0.00	40.20
Paradigmatic	1	67.53	59.89	51.14	22.59	51.89	38.58	65.81	6.11	0.00	40.39
Paradigmatic	42	66.76	59.11	51.35	20.49	52.55	38.09	65.30	6.09	0.00	39.97
Relay Joint	0	67.84	65.21	53.00	21.47	52.24	37.59	65.29	5.83	0.00	40.94
Relay Joint	1	66.28	60.09	52.44	21.07	52.09	32.71	65.03	5.92	0.00	39.51
Relay Joint	42	68.59	59.37	51.51	18.99	52.10	32.71	66.47	6.07	0.00	39.53

Table 8: Primary-system results by seed. Overall is the unweighted arithmetic mean of all nine top-level metrics.

Configuration	Seven-metric NLP Average
Standard MLM	50.31 ± 0.13
Syntagmatic	50.60 ± 0.29
Paradigmatic	50.81 ± 0.27
Simultaneous Joint	50.10 ± 0.75
Routing Only	50.60 ± 0.43
Entity Anchor Only	50.70 ± 0.92
Routing + Entity Anchor	50.65 ± 0.12
Clean Union	50.88 ± 0.37
Staged Joint	50.55 ± 0.39
Relay Joint	50.58 ± 1.07

Table 9: All configurations, mean $\pm$ sample SD over three seeds. This comparable aggregate excludes Reading and AoA and is not Overall. Clean Union directly combines full-state objectives without routing or anchoring.

C Representation Analyses

Paradigmatic neighborhood probe. The probe uses 2,000 shared syntax-token instances and 31 target-token IDs, used as bootstrap clusters, retrieving the correct eight-neighbor centroid among 64 candidates. Table 10 gives all paired item means and target-token cluster intervals from 10,000 resamples. The largest cluster is the token a (448 items). Cluster intervals average 3.74 times the width of item intervals. Full-384 is the primary analysis. The three 128-dimensional diagnostic slices are coordinate-based analyses, not discovered modules.

Syntagmatic diagnostics. Attention is ranked over SentencePiece positions before word-level mapping. Selected subwords are mapped to Universal Dependencies words, with duplicate words collapsed. Recall@8 is the fraction of ground truth direct dependency-neighbor words recovered; Precision@8 is the fraction of unique mapped selected words that are direct dependency neighbors. For context removal, the same selected token positions are zeroed in the input to the final Transformer layer. A matched control removes the same number of random non-special, non-target positions. Tables 11 and 12 include only the current Standard MLM and Syntagmatic runs.

Comparison	Space	Metric	Mean	CI low	CI high
Syn.–Std.	Full	Margin	+0.0178	+0.0114	+0.0227
Syn.–Std.	Full	MRR	+0.0409	+0.0058	+0.0796
Syn.–Std.	S1	Margin	+0.0448	+0.0333	+0.0522
Syn.–Std.	S1	MRR	+0.0585	+0.0380	+0.0797
Syn.–Std.	S2	Margin	-0.0166	-0.0258	-0.0047
Syn.–Std.	S2	MRR	-0.0076	-0.0262	+0.0118
Syn.–Std.	S3	Margin	+0.0243	+0.0124	+0.0325
Syn.–Std.	S3	MRR	+0.0383	+0.0137	+0.0638
Para.–Std.	Full	Margin	+0.0242	+0.0180	+0.0318
Para.–Std.	Full	MRR	+0.0845	+0.0388	+0.1463
Para.–Std.	S1	Margin	-0.0044	-0.0207	+0.0139
Para.–Std.	S1	MRR	+0.0257	+0.0033	+0.0625
Para.–Std.	S2	Margin	+0.0283	+0.0129	+0.0406
Para.–Std.	S2	MRR	+0.0700	+0.0395	+0.1197
Para.–Std.	S3	Margin	+0.0437	+0.0355	+0.0541
Para.–Std.	S3	MRR	+0.0472	+0.0233	+0.0794
Relay–Std.	Full	Margin	+0.0477	+0.0358	+0.0566
Relay–Std.	Full	MRR	+0.1445	+0.0773	+0.1977
Relay–Std.	S1	Margin	+0.1260	+0.1031	+0.1420
Relay–Std.	S1	MRR	+0.2188	+0.1482	+0.2812
Relay–Std.	S2	Margin	-0.0829	-0.1013	-0.0679
Relay–Std.	S2	MRR	-0.0467	-0.0794	-0.0238
Relay–Std.	S3	Margin	+0.0877	+0.0741	+0.0982
Relay–Std.	S3	MRR	+0.1128	+0.0774	+0.1363
Relay–Syn.	Full	Margin	+0.0299	+0.0211	+0.0368
Relay–Syn.	Full	MRR	+0.1036	+0.0463	+0.1517
Relay–Syn.	S1	Margin	+0.0812	+0.0641	+0.0935
Relay–Syn.	S1	MRR	+0.1603	+0.0985	+0.2146
Relay–Syn.	S2	Margin	-0.0663	-0.0873	-0.0501
Relay–Syn.	S2	MRR	-0.0390	-0.0645	-0.0200
Relay–Syn.	S3	Margin	+0.0634	+0.0522	+0.0757
Relay–Syn.	S3	MRR	+0.0745	+0.0361	+0.1065
Relay–Para.	Full	Margin	+0.0235	+0.0115	+0.0335
Relay–Para.	Full	MRR	+0.0600	+0.0056	+0.1060
Relay–Para.	S1	Margin	+0.1304	+0.1073	+0.1483
Relay–Para.	S1	MRR	+0.1931	+0.1237	+0.2587
Relay–Para.	S2	Margin	-0.1112	-0.1206	-0.1022
Relay–Para.	S2	MRR	-0.1166	-0.1723	-0.0821
Relay–Para.	S3	Margin	+0.0440	+0.0299	+0.0531
Relay–Para.	S3	MRR	+0.0656	+0.0249	+0.0913

Table 10: All neighborhood-probe comparisons. Intervals are 95% target-token clustered-bootstrap intervals conditional on matched checkpoints for seed 0, not across-seed intervals.

Method	Seed / summary	Recall@8	Precision@8
Standard MLM	0 / 1 / 42	.8707 / .8806 / .8699	.2997 / .3003 / .2970
	Mean ± SD	.8737 ± .0059	.2990 ± .0018
Syntagmatic	0 / 1 / 42	.8750 / .8759 / .8772	.2999 / .2979 / .2987
	Mean ± SD	.8760 ± .0011	.2988 ± .0010

Table 11: Dependency alignment by seed and sample mean ± SD.

Method	Seed / summary	Attended	Matched	Difference
Standard MLM	0 / 1 / 42	1.6525 / 1.6137 / .9038	-.0447 / .1678 / -.0039	1.6972 / 1.4459 / .9077
	Mean ± SD	1.3900 ± .4215	.0397 ± .1128	1.3503 ± .4034
Syntagmatic	0 / 1 / 42	1.8058 / 2.1002 / .7960	-.0438 / .0159 / -.0082	1.8496 / 2.0842 / .8042
	Mean ± SD	1.5673 ± .6840	−.0120 ± .0300	1.5793 ± .6814

Table 12: Ground truth token NLL increase after context removal, by seed and sample mean ± SD. Difference is attended minus matched.

Domain-Based Routing in Mixture-of-Experts Models for BabyLMs

Jeremias Bohn and Sebastian Feiertag and Nicolas Raube and Emil Schwenger and Georg Groh
Technical University of Munich
School of Computation, Information and Technology
Germany
{firstname.lastname}@tum.de

Abstract

Recent language model improvements were mostly achieved by increasing model scale, compute, and in particular training data size. This differs significantly from human language acquisition, which relies on significantly fewer words. The BabyLM challenge tackles this issue by encouraging approaches inspired by cognitive insights to reduce data dependency, thus lowering the barrier for NLP researchers to contribute to their field. We propose a Mixture-of-Experts approach with guided domain-based routing to learn subnetworks specialized on human-interpretable topics, thus making experts focus on domain knowledge and style rather than specific token features. Additionally, we apply a CEFR-based curriculum learning to mimic human language learning. Our model underperforms compared to the 7 times larger baseline model, but scores best on the BabyLM strict-small leaderboard among models with less than 25M parameters. Our code is publicly available at https://github.com/SmallLM-DBR-MoE-Submission-2026/Baby-DBR, and our model can be found at https://huggingface.co/elschr/BabyLM2026-DBR.

1 Introduction

Recent progress in language modeling has, to a large degree, been driven by scale. Scaling laws show that model performance improves predictably as parameter count, dataset size, and compute budget grow (Kaplan et al., 2020; Hoffmann et al., 2022). As a result, state-of-the-art language models (LMs) are trained on trillions of tokens using compute budgets that reach tens of millions of dollars per training run (Grattafiori et al., 2024; Cottier et al., 2025). This dependence on massive resources stands in stark contrast to human language acquisition: children are exposed to roughly 100 million words by age 13, yet acquire robust linguistic competence. It also places meaningful LLM pretraining research out of reach for most academics.

The BabyLM Challenge (Choshen et al., 2026) was created to address both issues. By restricting the pretraining data to 10 million words (for the strict-small track), it turns language modeling into a data-efficiency problem in which architectural and training innovations, rather than raw scale, determine success. This makes the setting both cognitively motivated and practically accessible to low-resource research environments.

In this work, we explore whether explicit, human-interpretable structure can improve data efficiency in this setting. We propose a Mixture-of-Experts (MoE) architecture in which expert routing is not learned purely implicitly, as in standard MoE models (Shazeer et al., 2017), but is instead guided by external topic labels: we partition our training corpus into eleven manually curated topic domains and use these gold labels as a supervision signal for the router during a dedicated specialization phase.

2 Related Work

Scaling Laws Kaplan et al. (2020) established that language model loss follows power laws in model size, dataset size, and compute, suggesting that training larger models on more data results in predictable gains. Hoffmann et al. (2022) refined this picture with the Chinchilla scaling laws, showing that many models of its time were undertrained relative to their size. For compute optimal training they show model size and training tokens should be scaled in equal proportion, with roughly 20 tokens per parameter. Under the BabyLM constraint of 10M words, this relationship is inverted: Data, not compute, is the binding constraint, which motivates architectural and algorithmic approaches to data efficiency rather than parameter scaling.

Proceedings of the Second BabyLM Workshop, pages 843–858
October 29, 2026 ©2026 Association for Computational Linguistics

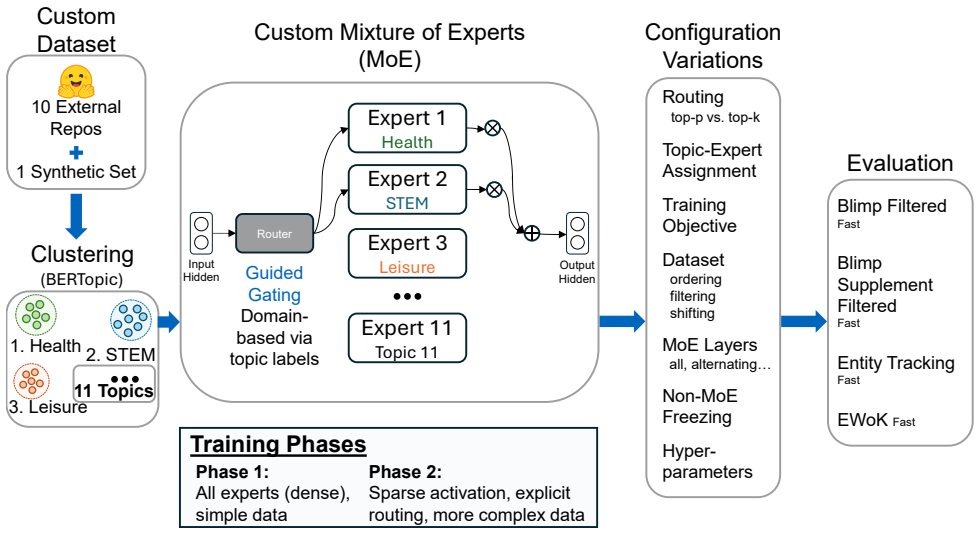


Figure 1: Overview of our core approach.

Mixture-of-Experts Mixture-of-Experts (MoEs) were introduced by Jacobs et al. (1991) as an alternative form of ensemble learning, allowing components of a full model to specialize on particular subtasks without deteriorating the overall model's performance. The approach gained significant traction through the work of Shazeer et al. (2017), who introduced sparse MoEs: a learned router dispatches each token to a small subset of expert feed-forward networks, decoupling model capabilities from inference compute. This design underlies many recent large-scale models (Meta AI, 2024; Team et al., 2026; Dai et al., 2024). The shared-expert design we build on, in which one expert is always active to capture common knowledge while routed experts specialize, was popularized by DeepSeekMoE (Dai et al., 2024). However, routing in these models is learned implicitly, and analyses have found that experts tend to specialize along shallow token-level or syntactic patterns rather than human-interpretable domains (Jiang et al., 2024). Our work differs in that as we impose an explicit, semantically meaningful routing signal derived from topic labels.

MoE in BabyLM 2025 Mehta et al. (2025) find that while MoE alone already increases performance, it does so even more in combination with Multi-head Latent Attention (MLA) and Rotary Position Embeddings (RoPE). Tapaninaho (2025) experiments with shrinking the hidden dimension, then applying MoE-style routing to route tokens to multiple transformer blocks at the same time, and growing the hidden dimension afterward. This approach outperformed all strict-small baseline models. Overall, architectural improvements are recommended over parameter scaling in low-resource language modeling (Mehta et al., 2025; Charpentier et al., 2025).

Domain-based Routing DEMix layers (Gururangan et al., 2021) condition expert selection in the feed-forward sublayer directly on document-level domain labels, training one expert per domain while sharing attention parameters across domains. Branch-Train-Merge (Li et al., 2022) instead trains separate expert LMs per domain and combines them at inference. Closest to our approach is DES-MoE (Li et al., 2025), which uses explicit domain metadata during the finetuning stage of an existing pretrained model, restricting updates to domain-specific experts via a multi-stage freezing schedule to counteract catastrophic forgetting. We differ by investigating explicit topic supervision during the pretraining phase.

3 Methods

In standard MoE models, routing is learned implicitly under load-balancing (Shazeer et al., 2017) constraints, and experts tend to specialize along shallow token-level or syntactic lines rather than human-interpretable topics. In contrast, we partition the training corpus into eleven semi-automatically defined topics prior to training and use these topic labels as an explicit routing signal, encouraging experts to develop semantically meaningful, topic-aligned specialization.

Inspired by natural learning processes, we further divide our training into two phases: During the

base phase, we use simple natural language and train our model densely, i.e., we activate all experts simultaneously, to accustom the model to the data. Afterward, in the *expert phase*, we use sparse routing to activate only specific experts corresponding to specific topics. A visualization of our approach can be found in Figure 1.

3.1 Custom Dataset and Clustering

Dataset Composition Since initial experiments showed that the challenge's own dataset was hard to separate by topic modeling, we composed our own instead to better match our research question, partially reusing subsets of the BabyLM strict-small data. Apart from that, the remaining data comes from public sources of fixed domains, such as mathematics and news, with a significant proportion taken from the UniversalCEFR dataset (Imperial et al., 2025), which allows for curriculum learning based on the language difficulty, determined by the CEFR level. Further, datasets comprising simpler language, such as Daily Dialog (Li et al., 2017) and the Everyday Language Corpus (MultivexAI, 2026), are added.

We then filter the collected datasets to exclude long sequences (> 2048 characters) to ensure diversity while staying within the challenge's word limit, which accumulates to a total of 9,653,393 words, which we partition into *base* (12.5%) and *expert* (87.5%) subsets. A comprehensive list and distribution of words can be found in Table 11.

Moreover, since the *base phase* requires simple speech and the corresponding number of dataset entries is relatively low, we synthesized additional data by sampling from our *base* subset and generating variations using DeepSeekV4 Flash. The exact prompt is given in Appendix B. In 70% of generations, the prompt was extended with a topic word to aim for unique examples, while a static prompt was used in the remaining 30%. In total, the generated data comprises 1,727,276 words, from which we only sample to fill up to a maximum of 10M words. As described in Section 4.6, additional instances from this subset are used for the *base phase* after additional filtering of the *expert phase* subset.

Topic Modeling To find topic groupings for each of the experts, we cluster dimensionality reduced embeddings of our examples using BERTopic (Grootendorst, 2022). This approach revealed 198 niche topics, with outliers being clustered into a single one, which we then combined into 11 more general topics. The clustering was conducted by first prompting an LLM (Gemini 3.1 Pro) based on 10 BERTopic keywords for each topic, and manually reviewing and adjusting afterward; as part of this manual reviewing process, 8 specific topics were relocated. The final general topic descriptions are as follows:

- Math, Physics, Earth Science, Chemistry, Space
- Technology, Business, and Economy
- Health, Medicine, and Anatomy
- Nature, Zoology, and Botany
- History, Geography, and Law
- Education, Language, and Arts
- Entertainment, Sports, and Gaming
- Everyday Life, Home, and Transport
- Interpersonal Communication & Society
- Fantasy, Adventure, & Historical Fiction
- Contemporary Fiction & Character Dialogue

3.2 Model Architecture & Training Procedure

We ground our model on a modern Transformer basis, building upon the Llama 3 (Grattafiori et al., 2024) architecture implemented as a HuggingFace transformers (Wolf et al., 2020) module. We keep the original architecture and focus on changes to the feed forward network (FFN): the FFN in each layer is replaced by an MoE module comprising 11 experts, with each expert designed as a single SwiGLU module. A linear router maps each token-level hidden state to a softmax distribution over the experts, from which tokens are dispatched using either top-k or top-p selection. Optionally, a shared expert is applied to every token.

As mentioned before, the training process is then divided into two phases, each running for 10 epochs on their corresponding subset: Phase 1 (*base phase*) establishes general language competence with all experts active. It is followed by phase 2 (*expert phase*) in which the topic-supervised routing signal is applied to encourage interpretable specialization. All models use a BPE tokenizer trained from scratch on our corpus.

The hyperparameters that were kept consistent across experiments are listed in Appendix D. Additional hyperparameters are described in the corresponding sections below.

4 Experiments and Results

While our main architecture and dataset are fixed, we conduct an extensive ablation study on different factors influencing the final model's performance,

including the routing mechanism, MoE layer locations, the gold label topic to model expert assignment strategy, dataset shifting, training data ordering and filtering, non-MoE freezing and changes to the training objective and usage of a (weighted) shared expert.

Throughout all experiments, we evaluate on the core tasks of the BabyLM challenge, i.e., BLiMP (Warstadt et al., 2020) and BLiMP Supplement (Warstadt et al., 2023), as well as Entity Tracking (Kim and Schuster, 2023) and EWoK (Ivanova et al., 2025).

Full experiment results for all ablation experiments are shown in Appendix A.

4.1 Catastrophic Forgetting

One of the most significant challenges we faced is catastrophic forgetting (McCloskey and Cohen, 1989) during the second phase, the expert specialization training. Our experiments in Appendix A show that for the *expert phase* models, the evaluation average tends to decline in comparison to their respective *base phase* evaluation results. This is primarily driven by decreases in BLiMP Supplement and Entity Tracking scores, while BLiMP scores generally rise. The basic model trained on simple speech tends to perform better on these tasks. This is also apparent in the score-by-epoch graph of our final submission model, as shown in Figure 3. The score for BLiMP Supplement suddenly decreases after the switch to the second training phase and fails to recover.

In an effort to address this problem, we trained multiple models that varied in terms of expert selection and MoE location. Furthermore, for the experiments in Table 5, at the start of the second (expert specialization) training phase, all non-MoE weights, except for the language model head, were frozen. This change aimed at improving entity tracking and BLiMP supplement scores by reducing the number of trainable parameters to expert internals. In addition, all models in this ablation had their phase one training dataset extended by a 30% sample from the expert dataset. We considered this necessary since we cannot expect the model to handle grammar and syntax of expert data it has never seen. The evaluation results are shown in Table 5.

In general, the changes did not help address the decline in average evaluation results. Instead, we further observed now that BLiMP scores also tend to decrease in the *expert phase*. Hence, the combination of shifting an expert dataset sample to the basic dataset and restricting training to experts only in the latter phase seemed to hurt understanding of grammar and syntax.

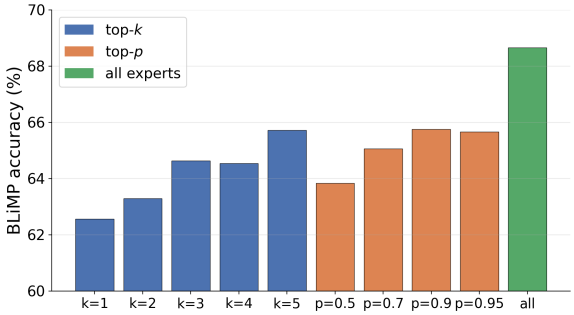


Figure 2: BLiMP fast zero-shot accuracy by routing configuration.

We did not observe any consistent improvements for this problem over the course of our experiments. We assume the problem is rooted in our training data, which may shift focus from basic understanding toward expert knowledge. This gain in knowledge is not properly reflected in the tasks we evaluated on, but may be beneficial in general autoregressive language modeling nonetheless.

4.2 MoE Adjustments

Given the specialization of our experts, the output of each of them may not be necessary for every task. Thus, we experiment with modifying the amount of active experts in a sparse MoE scenario, applying both Top-k (Shazeer et al., 2017) and Top-p (Holtzman et al., 2020) routing. Top-k routing sends each token to the k experts with the highest router scores, whereas Top-p routing sends each token to the smallest set of highest-scoring experts whose cumulative routing probability reaches the threshold p. This allows the number of experts to vary across tokens depending on router certainty.

As can be seen in Figure 2, we observe a positive correlation between the number of active experts (i.e., a higher k or p) and the accuracy on the BLiMP Benchmark. This correlation holds even up to the activation of all experts (all, so equivalent to $k = 11$ or $p = 1$), indicating that the sparse activation, while being computationally cheaper, does not boost accuracy in this separated ablation experiment. However, this may be a finding specific to the BLiMP dataset, since training a model with all experts active scored worse on the full evaluation submitted to the leaderboard (cf. Table 4). We therefore opted for $p = 0.7$ for our final sub-

mission, since this value showed a significant improvement over $p = 0.5$, while such an increment is not as prevalent after increasing p further.

Additionally to choosing experts dynamically, we further conducted experiments in which we added a shared expert (Dai et al., 2024) that is always active, allowing for a more consistent gradient flow and diverting basic knowledge away from our topic-primed experts. The intention is that a shared expert learns common basic language functions, such as grammar or basic words, that are needed for most inference tasks. Additionally, we experiment with introducing a dynamically calculated weight to scale its output. This weight is calculated based on the concatenated input vector and routing probabilities for the input. This vector is passed through a linear layer and a sigmoid function, which outputs one normalized score per token. This score is multiplied with the output of the shared expert.

Table 1 shows the results of these experiments. Both the regular shared expert and our weighted variant increase the average score to almost the same level compared to omitting the shared expert. The weighted shared expert achieves the highest result in BLiMP. The regular shared expert has the highest score for the base phase model. Additionally, using the top-k ($k = 1$) routing method instead of top-p ($p = 0.7$) leads to a better score in the base phase for the regular shared expert, but a worse score in the specialized phase. It is notable that Entity Tracking particularly improves in the *base phase* trained model. Given the mostly consistent improvements over all benchmarks after the expert phase, we decided to incorporate a weighted shared expert into our final submission model.

Furthermore, we considered replacing the Llama FFN with our custom MoE only in specific layers, e.g., by placing the MoE only in the top layers, alternating between MoE and regular SwiGLU layers, or replacing it in all layers. Table 6 shows the respective evaluation results. Not replacing all standard FFNs proved to be a solid choice for most tasks, with Entity Tracking once again profiting significantly after the *base phase*. In terms of average evaluation results, the alternating options seem to perform best across both evaluation points, and therefore being our choice for our final submission.

4.3 Router Teacher Forcing

While routing purely according to our gold labels in the training phase (to which we refer as *router teacher forcing*) guarantees no contamination of experts with different topics, it also imposes a significant constraint on the flexibility of the optimization process and may create a mismatch between training and inference, as the model then has to rely on its own routing scores instead. When applying router teacher forcing, the routing probabilities are only used to calculate the router loss, the experts are activated according to our gold labels. When deactivated, routing probabilities are used both to calculate the routing loss and to activate experts, allowing for a *soft topic guidance*, i.e., domain routing is only indirectly enforced by weight optimization. Related to routing is the auxiliary load balancing loss. This loss tries to activate experts more uniformly in the number of tokens that are routed to each expert and how uniformly distributed the activation probabilities are.

Our experiments show that the *expert phase* model performs better if trained without router teacher forcing (cf. Table 7), and with an auxiliary loss (`p_0.7_no_teacher_with_aux-expert`). No shared expert was used for this experiment, thus results are to be compared with `p_0.7_no_shared-expert` in Table 1.

While a clear separation between experts might be desirable for interpretability, the negative effects on all benchmarks tested served as a clear indicator to remove router teacher forcing from our final submission.

4.4 Expert Assignment

Since our dataset has eleven predefined topics and the *expert phase* training requires mapping from gold-label topics to model experts, we need a sophisticated assignment algorithm. The three evaluated options are loss-based assignment, gradient-based assignment, and fixed assignment.

To prepare data for this matching phase, we group all entries from the expert dataset by their highest assigned topic probability. Then we filter out all documents with a maximum likelihood below 75%, which results in 11 high-probability expert datasets.

With a similar intuition as presented in transferability literature (Nguyen et al., 2020), we implemented a *loss-based assignment*. For the loss-based assignment, we loop over every expert dataset and model expert combination. We perform a forward pass while routing tokens only to the selected model expert and observe the language modeling (cross-entropy) loss. This process produces a ma-

Name	BLiMP	Supplement	ET	EWoK	Avg.
p_0.7_no_shared-base	62.04	60.00	14.63	48.68	46.34
p_0.7_no_shared-expert	65.07	53.20	*15.70*	50.14	46.03
p_0.7_shared_exp-base	61.22	62.00	**39.96**	51.00	**53.55**
p_0.7_shared_exp-expert	65.68	*57.60*	13.08	50.82	46.80
p_0.7_weighted_shared_exp-base	61.48	**64.00**	33.35	51.09	52.48
p_0.7_weighted_shared_exp-expert	***66.26***	55.60	13.78	***51.82***	*46.87*
k_1_shared_exp-base	61.21	62.00	39.82	50.36	53.35
k_1_shared_exp-expert	64.68	56.00	13.63	50.00	46.08

Table 1: Baseline compared to a standard shared expert, the weighted shared expert and the standard shared expert with top-k ($k = 1$). All models with shared experts consistently outperform the base except after the *expert phase* for Entity Tracking and some instances of EWoK. Best scores per task in **bold**, best after expert phase in *italics*.

trix that assigns a loss to each tuple (gold-label topic, model expert). In the final matching, gold-label topics are assigned to model experts in a 1-to-1 manner to minimize the total loss, following the assumption that a lower loss indicates a better fit.

In a comparable fashion we also decided to experiment with a *gradient-based assignment*. For the gradient-based assignment, we proceed as follows for every expert dataset: First, a forward-and-backward pass is performed that routes tokens to every expert with uniform probability. Afterward, we calculate the L2 norm of the gradients for each of the 11 experts across all layers. For all 11 expert datasets, this results in a matrix that assigns a gradient to each tuple (gold-label topic, model expert). In the final matching, gold-label topics are assigned to model experts in a 1-to-1 manner to minimize their gradient sum, following the assumption that a lower gradient indicates a better fit.

Finally, the *fixed assignment* naïvely assigns the first gold-label topic to the first model expert, and the following topics to the subsequent model experts, respectively. In this process, no performance metric is involved.

Experiment results are shown in Table 8. Unexpectedly, the fixed assignment achieved the highest average evaluation result, with none of the more sophisticated methods outperforming the naïve approach. It is subject to further investigation whether this is due to noise or due to the gradient-based or loss-based assignment methods being insufficient proxies. These findings lead us to choose a fixed assignment for our final model.

4.5 Parameter Adjustments

Though we have chosen our model size according to established scaling laws, we further tested a larger model by doubling the FFN size, context window, and vocabulary size to accommodate for potential underfitting due to sparse expert activation. To this end, we incorporate all changes discussed so far and conduct the following experiments using these findings.

Table 9 shows the results of five experiments, including a baseline with no changes, and four variants that always have an FFN doubled in size and combinations of context window (256/512) and vocabulary size (5k/12k) changes. The results clearly show that the expert model neither significantly improves nor degrades with any configuration change. For the sake of simplicity, we decided not to incorporate any tested configuration changes in our final submission.

4.6 Curriculum Learning & Complexity Reduction

Curriculum learning was a re-occurring approach in previous BabyLM challenges (Charpentier et al., 2025). We try to implement curriculum learning by sorting our dataset by the lix score, a simple measure of complexity. Developed by Björnsson (1968) (according to Anderson (1983), original source not traceable), it is calculated using the average sentence length and the percentage of words that have more than six letters. We additionally run experiments that remove the most complex data (e.g., keeping only the bottom 80% of complexity) from our dataset and replace it with samples from our synthetic dataset. Results are shown in Table 10. The average score increases from keeping only the bottom 60% to the bottom 80%. Afterward, when keeping the bottom 90%, the average score decreases again, with BLiMP accuracy still increasing slightly. Given these insights, we decided to replace the top 20% complex instances from our corpus for the final training run.

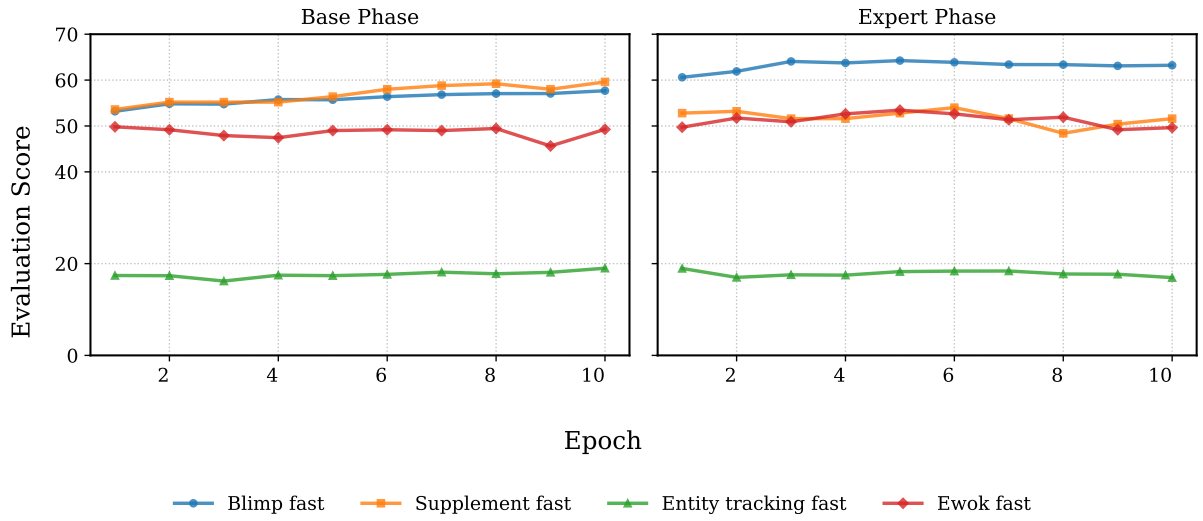


Figure 3: Submission model evaluation performance across different epochs

4.7 Final Setup

As our baselines, we use both the GPT-2-Strict-Small baseline, as reported by the BabyLM leaderboard, and a dense Llama 3 model trained on our custom dataset. The Llama baseline performs competitive with the reported GPT-2 Baseline. As can be seen in Table 2, our Baseline gets a slightly higher BLiMP score but a weaker Entity Tracking score. This is to be expected as our dataset contains fewer examples of simple children speech, which we consider particularly well-suited for tasks like entity tracking, but more examples of harder and higher quality text that improves grammar understanding as measured by BLiMP.

Name	BLiMP	Suppl.	ET	EWoK	Avg.
GPT-2	65.23	57.25	19.10	50.63	48.53
Llama 3	66.81	56.00	14.01	50.27	46.77
Base	57.68	59.6	19.0	49.27	46.39
Expert	63.23	51.6	16.95	49.64	45.36

Table 2: Zero-shot results: BabyLM-2026 GPT-2 Strict-Small baseline vs. our LLaMA dense baseline vs. our submission MoE models, evaluated after the *base phase* and the *expert phase* on BLiMP, BLiMP Supplement, Entity Tracking, and EWoK

The configuration of our submission model, shown in Table 3, was built by combining the findings of the comparative experiments mentioned in the above sections. Additionally, we submitted a second model, `BabyDBRFull`, that was trained using all experts, i.e., setting $p = 1.0$. The leaderboard results (best score of all epochs per task) for both models can be found in Table 4.

Configuration	Value
Top-p	0.7
Context Window	256
MoE Layers	2 MoE layers out of 4 placed alternatingly
Shared Expert	One weighted shared expert
Vocab Size	5,000
Training Objective	Load balancing (aux) loss + routing loss
Expert Assignment	Fixed expert assignment
Curriculum Learning	Order by text complexity (lix)
Data Complexity	The most complex 20% of training data was dropped

Table 3: Submission Model Configuration

Table 2 shows the final model evaluation results for GPT-2, LLaMA, and our submission model. Overall, the reported GPT-2 baseline, trained on the official BabyLM Strict-Small dataset, and the LLaMA baseline, trained on our base and expert dataset, outperform our expert submission model on average and on BLiMP in every setting. Some

Name	BLiMP	Suppl.	ET	EWoK	Avg.
Base	63.47	51.53	19.02	51.53	46.39
Full	65.26	49.78	18.29	50.76	46.02

Table 4: Leaderboard results for `BabyDBR` (Base) and `BabyDBRFull` (Full)

features to highlight include the resulting model from training phase one, which outperforms the GPT-2 and LLaMA baselines in BLiMP Supplement and the Llama baseline in Entity Tracking. In particular, BLiMP Supplement experiences a decline of 8 points between the *base phase* and

the *expert*, as noted in Section 4.1. While entity tracking slightly decreases and EWoK stays similar, BLiMP increases by almost 6 points. In addition, Figure 3 shows evaluation results for our submission model from epochs 1 to 10 during training on the base and expert datasets. Entity Tracking performance during training stays approximately constant.

5 Discussion

While our model does not outperform the full-sized baselines, it is worth noting that our model is significantly smaller in size, amounting to a total of 13.8M parameters versus 98.4M of the GPT-2 baseline. In particular, at submission time of this paper, our model is the best-performing on the strict-small leaderboard with less than 25M parameters (of which there are 9 verifiably below this limit).

Our experiments also show apparent dissonances between human-intent design and performance gains in language models, as seen in our router teacher forcing and expert assignment experiments: Clear separation and interpretability can hurt performance, and algorithm designs intended to exploit inert preference in random initializations may still perform worse than arbitrary choices.

As implied by our section on catastrophic forgetting and complexity reduction, the main question arises whether the data collected for our experiments is insufficient for good language modeling, or whether the given evaluation benchmarks are not a suitable proxy for measuring it. Since language models are often used for knowledge tasks, we propose a stronger focus on such evaluations in future editions of the workshop, but providing a fixed knowledge database for competition participants to keep the focus on language modeling.

6 Conclusion

We presented a structured MoE approach for the BabyLM Strict-Small track in which expert routing is supervised by externally derived, human-interpretable topic labels rather than learned purely implicitly. To this end, we constructed a custom 10M-word corpus, derived eleven topic domains via BERTopic clustering and manual curation, and conducted systematic ablations over routing mechanisms, MoE placement, shared experts, expert assignment strategies, curriculum ordering, and model size. Our best configuration, alternating MoE layers, a weighted shared expert, no router teacher forcing, and fixed expert assignment, outperforms both the reported GPT-2 baseline and our dense LLaMA baseline on BLiMP when all experts are active, while the sparse variant roughly matches the dense baseline at lower compute cost. At the same time, our ablations paint a consistent and instructive picture: the domain-based routing model performs well given its size, and benchmark scores improve by increasing flexibility during the training process, whether through more active experts, load-balancing auxiliary losses, or router self-prediction instead of teacher forcing. The catastrophic forgetting observed in the specialization phase, however, suggests a mismatch between the training data and the evaluation target.

7 Outlook

To clarify whether the limitations we observe are fundamental or specific to our setup, an interpretability analysis of what the experts actually learn under domain supervision would show whether the intended topic specialization emerges at all, independent of benchmark scores.

While our router teacher forcing approach showed slightly worse results, it may be interesting to investigate whether the pure load balancing loss distributes general knowledge more evenly across the experts, thus improving performance similar to introducing a shared expert. Further experiments would be required to measure this effect, which we leave to future work.

Limitations

While we conducted our experiments with rigor, some caveats remain: For all our experiments, we only train our models and evaluate on the BabyLM fast subsets once per configuration using a single seed; more runs to better estimate the effect on model capabilities could improve our architecture further or suggest different findings. An extensive evaluation of the clustering result, which labeled each data point with one of 198 topic clusters was not possible due to time constraints; a manually curated dataset may improve the routing, but is very labor-intensive. Given our custom dataset, comparability with other BabyLM architectures may be limited; while our approach did not work with the provided dataset, other leaderboard entries may perform better using our own dataset.

Acknowledgments

We used Claude Opus 4.6, Claude Sonnet 4.6, Gemini 3.1 Pro, Gemini 3.5 Flash, Gemini 3.1 Flash, Deepseek V4 Flash, Deepseek V4 Pro, GPT-5 Mini to gather ideas for details of our experiments, write out syntax in the implementation, and review. The core approach and general ideas for ablation studies are our own. Deepseek V4 Flash was used to generate synthetic data.

References

Jonathan Anderson. 1983. Lix and rix: Variations on a little-known readability index. *Journal of Reading*, 26(6):490–496.

Yuki Arase, Satoru Uchida, and Tomoyuki Kajiwara. 2022. CEFR-based sentence difficulty annotation and assessment. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6206–6219, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Carl-Hugo Björnsson. 1968. *Läsbarhet: hur skall man som författare nå fram till läsarna?* Bokförlaget Liber.

Mark Breuker. 2023. Cefr labelling and assessment services. In Georg Rehm, editor, *European Language Grid: A Language Technology Platform for Multilingual Europe*, pages 277–282. Springer International Publishing, Cham.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data. In *Proceedings of the First BabyLM Workshop*, pages 399–420, Suzhou, China. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ben Cottier, Robi Rahman, Loredana Fattorini, Nestor Maslej, Tamay Besiroglu, and David Owen. 2025. The rising costs of training frontier ai models. *Preprint*, arXiv:2405.21015.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *Preprint*, arXiv:2401.06066.

Lennart Finke, Chandan Sreedhara, Thomas Dooms, Mat Allen, Juan Diego Rodriguez, Noa Nabeshima, Thomas Marshall, and Dan Braun. 2025. Parameterized synthetic text generation with simplestories. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

S. Granger, M. Dupont, F. Meunier, H. Naets, and M. Paquot. 2020. The international corpus of learner english (version 3). Louvain-la-Neuve: Presses universitaires de Louvain.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Maarten Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. 2021. Demix layers: Disentangling domains for modular language modeling. *Preprint*, arXiv:2108.05036.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pages 30016–30030.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Joseph Marvin Imperial, Abdullah Barayan, Regina Stodden, Rodrigo Wilkens, Ricardo Muñoz Sánchez, Lingyun Gao, Melissa Torgbi, Dawn Knight, Gail Forey, Reka R. Jablonkai, Ekaterina Kochmar, Robert Joshua Reynolds, Eugénio Ribeiro, Horacio Saggion, Elena Volodina, Sowmya Vajjala, Thomas François, Fernando Alva-Manchego, and Harish Tayyar Madabushi. 2025. UniversalCEFR: Enabling open multilingual research on language proficiency assessment. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9703–9755, Suzhou, China. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *Preprint*, arXiv:1810.09305.

Junzhuo Li, Bo Wang, Xiuze Zhou, and Xuming Hu. 2025. Dynamic expert specialization: Towards catastrophic forgetting-free multi-domain MoE adaptation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 18478–18493, Suzhou, China. Association for Computational Linguistics.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *Preprint*, arXiv:2208.03306.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. 2024. Fineweb-edu: the finest collection of educational content.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Sushant Mehta, Raj Dandekar, Rajat Dandekar, and Sreedath Panat. 2025. Unifying mixture of experts and multi-head latent attention for efficient language models. In *Proceedings of the First BabyLM Workshop*, pages 42–51, Suzhou, China. Association for Computational Linguistics.

Meta AI. 2024. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation.

MultivexAI. 2026. Everyday language corpus. `https://huggingface.co/datasets/MultivexAI/Everyday-Language-Corpus`.

Tarek Naous, Michael J Ryan, Anton Lavrouk, Mohit Chandra, and Wei Xu. 2024. ReadMe++: Benchmarking multilingual language models for multi-domain readability assessment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12230–12266, Miami, Florida, USA. Association for Computational Linguistics.

Cuong V. Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. 2020. Leep: a new measure to evaluate transferability of learned representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Asankhaya Sharma. 2025. The 1 billion token challenge: Finding the perfect pre-training mix.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Anaïs Tack, Thomas François, Sophie Roekhaut, and Cédrick Fairon. 2017. Human and automated CEFR-based grading of short answers. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 169–179, Copenhagen, Denmark. Association for Computational Linguistics.

Joonas Tapaninaho. 2025. MoEP: Modular expert paths for sample-efficient language modeling. In *Proceedings of the First BabyLM Workshop*, pages 540–547, Suzhou, China. Association for Computational Linguistics.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, and 282 others. 2026. Gemma 4 technical report. *Preprint*, arXiv:2607.02770.

Peter Thwaites, Nathan Vandeweerd, and Magali Paquot. 2025. Crowdsourced comparative judgement for evaluating learner texts: How reliable are judges recruited from an online crowdsourcing platform? *Applied Linguistics*, 46(4):611–628.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Menglin Xia, Ekaterina Kochmar, and Ted Briscoe. 2016. Text readability assessment for second language learners. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 12–22, San Diego, CA. Association for Computational Linguistics.

A Experiment Results

Name	BLiMP	Supplement	Entity Tracking	EWoK	Avg.
k_1_moe_full-base	66.14	55.60	20.95	51.27	48.49
k_1_moe_full-expert	61.49	51.60	18.96	48.59	45.16
k_1_moe_two_alternating-base	**66.63**	56.40	15.03	50.09	47.04
k_1_moe_two_alternating-expert	63.69	51.20	14.27	51.23	45.10
k_2_moe_full-base	66.25	56.00	20.57	51.41	48.56
k_2_moe_full-expert	62.67	50.80	20.07	51.41	46.24
k_2_moe_two_alternating-base	65.98	57.60	**26.14**	50.09	**49.95**
k_2_moe_two_alternating-expert	64.20	57.20	14.73	51.05	46.80
p_0.4_moe_full-base	66.18	55.20	20.93	51.32	48.41
p_0.4_moe_full-expert	62.40	52.40	20.22	48.86	45.97
p_0.4_moe_two_alternating-base	65.86	**60.00**	17.83	50.09	48.45
p_0.4_moe_two_alternating-expert	64.01	50.40	20.89	**51.50**	46.70

Table 5: Freezing Non-MoE (Phase 2) + Shift 30% to Basic Dataset + 12k Vocabulary. Best scores per task in **bold**.

Name	BLiMP	Supplement	ET	EWoK	Avg.
p_0.7_all_moe-base	62.04	60.00	14.63	48.68	46.34
p_0.7_all_moe-expert	65.07	53.20	15.70	50.14	46.03
p_0.7_two_moe_alternating-base	61.43	**62.40**	**38.37**	48.82	**52.76**
p_0.7_two_moe_alternating-expert	**65.76**	56.00	17.93	50.09	47.45
p_0.7_two_moe_top-base	61.78	**62.40**	24.69	**50.95**	49.96
p_0.7_two_moe_top-expert	65.00	54.40	13.67	50.23	45.83

Table 6: MoE All vs. Alternating vs. Top. Best scores per task in **bold**.

Name	BLiMP	Supplement	ET	EWoK	Avg.
p_0.7_no_teacher_with_aux-base	61.76	**60.80**	13.89	48.45	46.23
p_0.7_no_teacher_with_aux-expert	**67.30**	55.60	16.60	**51.23**	47.68
p_0.7_no_teacher_no_aux-base	61.02	**60.80**	15.98	50.55	47.09
p_0.7_no_teacher_no_aux-expert	66.07	52.80	14.35	50.73	45.99
k1_no_teacher_no_aux-base	62.23	60.40	**24.66**	49.64	**49.23**
k1_no_teacher_no_aux-expert	63.26	54.40	13.41	50.64	45.43

Table 7: Deactivated Router Teacher Forcing with Auxiliary Loss vs. without Auxiliary Loss vs. top-k without auxiliary loss. Best scores per task in **bold**, best after expert phase in *italics*.

Name	BLiMP	Supplement	ET	EWoK	Avg.
p_0.7_fixed_all_moe-base	62.15	61.60	13.48	50.32	46.89
p_0.7_fixed_all_moe-expert	**66.03**	51.60	**25.23**	51.23	**48.52**
p_0.7_grad_all_moe-base	62.04	60.00	14.63	48.68	46.34
p_0.7_grad_all_moe-expert	65.07	53.20	15.70	50.14	46.03
p_0.7_loss_all_moe-base	62.48	**62.40**	14.96	50.05	47.47
p_0.7_loss_all_moe-expert	65.63	55.60	19.09	**51.77**	48.02

Table 8: Expert Assignment Fixed vs. Gradient based vs. Loss based. Best scores per task in **bold**.

Name	BLiMP	Supplement	ET	EWoK	Avg.
default_size_default_context_5k_vocab-base	61.43	59.20	34.02	50.00	51.16
default_size_default_context_5k_vocab-expert	**66.51**	55.20	13.99	49.91	46.40
double_size_256_context_12kvocab-base	59.26	60.00	35.26	50.36	**51.22**
double_size_256_context_12kvocab-expert	66.33	53.20	13.62	**51.18**	46.08
double_size_256_context_5kvocab-base	60.37	**61.20**	21.40	49.45	48.11
double_size_256_context_5kvocab-expert	65.89	57.20	14.45	49.36	46.73
double_size_512_context_12kvocab-base	58.21	56.00	**37.82**	47.18	49.80
double_size_512_context_12kvocab-expert	66.38	55.20	14.85	**51.18**	46.90
double_size_512_context_5kvocab-base	58.36	58.80	29.12	48.18	48.62
double_size_512_context_5kvocab-expert	64.97	50.80	15.13	49.64	45.14

Table 9: Double Expert Context Window, Double MoE FFN Size, 5k/12k Vocabulary. Best scores per task in **bold**.

Name	BLiMP	Supplement	ET	EWoK	Avg.
p_0.7_curric-base	56.29	58.80	**42.61**	49.09	**51.70**
p_0.7_curric-expert	65.43	52.00	14.44	49.91	45.45
p_0.7_curric_lix_keep_0_6-base	58.63	**62.80**	28.96	51.00	50.35
p_0.7_curric_lix_keep_0_6-expert	63.78	55.20	27.02	50.27	49.07
p_0.7_curric_lix_keep_0_8-base	58.87	60.40	38.11	48.82	51.55
p_0.7_curric_lix_keep_0_8-expert	64.87	55.60	29.27	52.09	50.46
p_0.7_curric_lix_keep_0_9-base	58.39	61.20	13.60	49.55	45.69
p_0.7_curric_lix_keep_0_9-expert	**65.35**	51.60	13.78	**52.27**	45.75

Table 10: Curriculum Learning via Lix Score Keeping All Expert Phase Data vs. Retaining Bottom 60/80/90% of Lix Score. Best scores per task in **bold**.

B Fill Dataset Prompt

B.1 Prompt Base

```
Your task is to help expand a dataset for language model training by generating
synthetic variations of input texts.
Given an English input example, generate a completely new example in English that
mimics the structural complexity, length, and format (e.g., sentence, paragraph,
or dialogue) of the input.

{topic_instruction}

Focus on creating a meaningful variation. Avoid simply rephrasing the input.
The language level and vocabulary MUST be at most CEFR A2. Use natural, simple,
everyday language suitable for a 5-year-old child.

The new example should have at least 4 words and at most 100 words.
Avoid referring to colors.

DO NOT use titles and keep the text free of any markdown or similar formatting.
Output ONLY the new example text, without any additional introductory or concluding
remarks, commentary or explanations.
```

B.2 Prompt - Topic Instruction

B.2.1 Static - 30%

```
Change the topic, entities, and subject matter to create a distinct example.
Entities in the new example can also remain the same as in the input if they
are common and generic words a 5-year-old would know (e.g., "dog", "cat",
"car"), but try to use different ones when possible.
```

B.2.2 Dynamic - 70%

```
Change the topic, entities, and subject matter to create a distinct example.
If possible, make the new example involve the word "{topic_word}" to encourage
topic diversity. However, if "{topic_word}" does not fit naturally, you can use it
as a general theme, or choose a different common word that a 5-year-old would know.
```

C Dataset Composition

Parent	Name & Source	Content	Target Dataset	Basic / Expert Words	License
	FinWeb-Edu 100M (Sharma, 2025) a sample of FineWeb-Edu (Lozhkov et al., 2024)	Educational	Expert	0 / 2925611	odc-by
	Daily Dialog (Li et al., 2017)	Daily Dialogs	Base	920394 / 0	cc-by-nc-sa-4.0
	OpenAI GSM8K (Cobbe et al., 2021)	Mathematics	Expert	0 / 475550	mit
	Facebook Empathetic Dialogues (Rashkin et al., 2019); Empathetic Dialogues GitHub	Emotions / Empathy	Expert	0 / 576485	cc-by-nc-4.0
	SimpleStories (Finke et al., 2025)	Synthetic / Story	Expert	0 / 1100650	mit
	BabyLM Strict-Small Simple Wiki (Choshen et al., 2026)	Wiki	Expert	0 / 1020687	mit
	BabyLM Strict-Small Gutenberg (Choshen et al., 2026)	Multi-Domain	Expert	0 / 1622367	mit
	WikiHow Cleaned (Koupaee and Wang, 2018)	How-to descriptions	Expert	0 / 540500	cc-by-nc-sa-3.0
	Everyday Language Corpus (MultivexAI, 2026)	Everyday Language	Base	79200 / 0	cc-by-4.0
UniversalCEFR (Imperial et al., 2025)	CEFR_SP_EN (Arase et al., 2022)	News / Wiki / Educational	Base(A-B1+) / Expert(>B1+)	55379 / 92506	cc-by-nc-sa-4.0
	README_EN (Naous et al., 2024)	Multi-Domain	Base(A-B1+) / Expert(>B1+)	20105 / 35954	cc-by-nc-sa-4.0
	CEFR_ASAG_EN (Tack et al., 2017)	Question-answers	Base(A-B1+) / Expert(>B1+)	12596 / 7943	cc-by-nc-sa-4.0
	ICLE500_EN (Thwaites et al., 2025; Granger et al., 2020)	Learner-essay	Base(A-B1+) / Expert(>B1+)	4791 / 7927	cc-by-nc-4.0
	ELG_CEFR_EN (Breuker, 2023)	News / Magazines / Educational / Government	Base(A-B1+) / Expert(>B1+)	90113 / 38023	cc-by-nc-4.0
	CAMBRIDGE_EXAMS_EN (Xia et al., 2016)	Cambridge English Exams	Base(A-B1+) / Expert(>B1+)	24105 / 2507	cc-by-nc-sa-4.0

Table 11: Dataset Sources

D Hyperparameters

Hyperparameter	Value
Model Architecture	
Block Size	256
Max position embeddings	256
Expert Intermediate Size	512
Attention Heads	4
Key-Value Heads	4
Number of Hidden Layers	4
Hidden Activation	SiLU
MLP Bias	False
RoPE Theta	500,000.0
Attention Dropout	0.1
Mixture-of-Experts (MoE) & Router	
Number of Experts	11
Shared Expert Intermediate Size	512
Shared Expert Gate Init Std	0.02
Router Initialization Std	0.02
Router Aux Loss Coefficient	0.01
Router Loss Type	KL Divergence
Training & Optimization	
Learning Rate (Phase 1 & Phase 2)	12×10^{-4}
LR Scheduler Type (Phase 1 & Phase 2)	Cosine
Optimizer	PyTorch's AdamW
Warmup Steps (Phase 1 / Phase 2)	25 / 100
Per-Device Train / Eval Batch Size	64 / 32
Gradient Accumulation Steps	1
Number of Training Epochs	10
Weight Decay	0.1
Max Gradient Norm	1.0
FP16 / BF16	False / False
Tokenizer Algorithm	BPE
Tokenizer Min Frequency	2
Random Seed	42

Table 12: Hyperparameters kept consistent across all experiments.

Dense Supervision, Gentle Decay: Curriculum-Guided Rebalancing of ELECTRA-Style RTD and MLM for Small-Data Pretraining

Aftab Hossain Mikat and Tamjid Hasan Fahim
Department of Computer Science and Engineering
Rajshahi University of Engineering & Technology, Bangladesh
ahmikat@gmail.com, tamjidhfahim@gmail.com

Abstract

When training data is very limited, the way different training signals are weighted over time can strongly affect what a model learns. We study hybrid replaced-token detection (RTD) and masked language modeling (MLM) pretraining under the BabyLM 2026 Strict-Small budget. We train five single-seed DeBERTa-v2 models along three curriculum axes: objective scheduling, masking policy, and data order. The primary system, bgb-decay, retains generator MLM, discriminator RTD, and auxiliary discriminator MLM throughout training while reducing the RTD coefficient in stages. We compare it with a matched fixed-RTD reference, a variant that adds easy-to-hard data ordering and adaptive auxiliary masking, and two stronger RTD-to-MLM handoff schedules with whole-word and adaptive masking. Among the five runs, bgb-decay achieves the best overall balance, while the more aggressive variants gain more on some tasks but drop more on others. Training diagnostics reveal a trade-off between replacement discrimination and auxiliary vocabulary prediction: under fixed high RTD weighting, RTD gradients are larger and nearly orthogonal to auxiliary-MLM gradients, while staged decay preserves RTD but shifts late optimization toward MLM. These results suggest that gradual objective rebalancing is a promising direction for small-data pretraining.

1 Introduction

The BabyLM Challenge studies language-model pretraining under severe data constraints, with the goal of understanding which learning strategies remain effective when models cannot rely on web-scale corpora (Warstadt et al., 2023). In the Strict-Small track, training is limited to a 10-million-word corpus and at most ten passes over that corpus (BabyLM Organizers, 2026). Under this constraint, training design becomes especially important: the same text can provide very different learning signals depending on which objective dominates, which tokens are selected for prediction, and which examples are presented first. We study this question through a hybrid pretraining setup that combines masked language modeling (MLM) with replaced-token detection (RTD) rather than treating them as competing objectives; Section 2 situates this choice relative to prior BabyLM systems and pretraining objectives.

Most curriculum-learning work in this space varies a single axis, such as example order or masking difficulty, while holding the underlying objective fixed (Schoenegger et al., 2025; Lucas et al., 2024). We instead organize our experiments along three curriculum axes that can be varied independently or in combination. An **objective curriculum** changes which losses are active or how strongly they are weighted. A **corruption curriculum** changes what the model must predict, for example by moving from uniform token masking to whole-word or difficulty-adaptive masking policy. A **data curriculum** changes when training examples are presented, such as ordering examples from easier to harder before returning to random shuffling. This distinction provides a common structure for the five runs in our study and makes it possible to compare otherwise heterogeneous interventions on equal footing.

Our main model, **bgb-decay**, varies only the objective axis: it keeps RTD and MLM jointly active for the full training budget but stages the RTD coefficient downward over time. We compare it against a matched fixed-RTD reference, a curriculum extension of that reference, and two more aggressive schedules that fully hand off from RTD to MLM partway through training, three of which were additionally motivated by improving performance on Human-like evaluation metrics such as Age of Acquisition (AoA) (Fenson et al., 2007; Chang and Bergen, 2022). Across the five single-seed runs, bgb-decay achieves the strongest aggregate profile,

Proceedings of the Second BabyLM Workshop, pages 859–874
October 29, 2026 ©2026 Association for Computational Linguistics

and its advantage is broad rather than concentrated in one task family: the more aggressive variants post the best scores on individual categories but also the largest losses elsewhere. This pattern suggests the central question is not simply whether to use RTD, MLM, or curriculum learning, but how to allocate a limited optimization budget among these signals over the course of training.

2 Related Work

Data-constrained pretraining and shared artifacts. The BabyLM Challenge asks which pretraining strategies remain effective when a model is limited to a developmentally plausible amount of text rather than a web-scale corpus (Warstadt et al., 2023; Hu et al., 2024; Choshen et al., 2026). We use the organizers' 2026 Strict-Small corpus release (BabyLM Organizers, 2026), evaluation implementation, and leaderboard. Prior BabyLM submissions vary architecture, data, and optimization as well as curriculum. Our five runs instead share the corpus, tokenizer family, backbone dimensions, and budget. This improves comparability, although only bgb-decay versus Baby Got BERT is a one-factor comparison; the other runs remain system-level comparisons.

Replaced-token detection versus masked language modeling. ELECTRA introduced replaced-token detection (RTD) as an alternative to MLM: a small generator proposes replacement tokens, and a discriminator is trained to detect which tokens in the resulting sequence were replaced (Clark et al., 2020). Because RTD supervises every non-special token rather than only a masked subset, it has been argued to be a more sample-efficient objective than MLM, a property that is directly appealing under the Strict-Small budget. We adopt a compact DeBERTa-v2 encoder as our discriminator backbone (He et al., 2021b,a). Rather than treating RTD and MLM as competing choices of pretraining objective, as is typical when comparing ELECTRA-style and BERT-style models, we combine them in a single discriminator and study how the *balance* between them should change over training — a question that sits one level above the usual RTD-versus-MLM comparison.

Data curricula. Curriculum learning conventionally orders or selects examples from easier to harder (Bengio et al., 2009); competence-based variants gradually expand the portion of data available to the learner (Platanios et al., 2019). BabyLM work has instantiated difficulty using sentence complexity (Oba et al., 2023), model uncertainty (Hong et al., 2023), acquisition-theoretic stages (Salhan et al., 2024), and training-example influence (Schoenegger et al., 2025). These studies report mixed, often capability-specific outcomes, so "curriculum" is not a single intervention with a single expected effect. Our *data curriculum* label is reserved for this example-ordering family; LAD is our only run that changes this axis.

Corruption and objective curricula. Curricula can also change the prediction problem without reordering examples. Whole-word masking (WWM) groups all subword pieces of a lexical item into one masking unit (Cui et al., 2021), while adaptive masking emphasizes tokens the current model predicts poorly (Yang et al., 2023; Edman and Fraser, 2025). Developmentally staged masking has also been explored in BabyLM by expanding the lexical or part-of-speech classes eligible for masking (Lucas et al., 2024). We call these *corruption curricula* because they change which targets are predicted. In contrast, our *objective curriculum* changes the relative weight or availability of RTD and MLM. This three-axis terminology is the paper's main conceptual distinction: prior example-order and masking curricula motivate LAD and Lilliput, while the matched bgb-decay comparison tests whether the balance between dense discrimination and lexical prediction should itself be scheduled.

3 Training Framework and Curriculum Axes

3.1 Hybrid Dense and Lexical Supervision

With a standard 15% mask rate, MLM applies its vocabulary-prediction loss only to the masked positions; unmasked tokens still provide context but are not themselves prediction targets. RTD instead turns most of the sequence into supervised signal: a generator fills selected positions with sampled replacements, and the discriminator predicts whether each non-special token in the resulting sequence is original or replaced. The two objectives are therefore related but not equivalent. RTD asks whether an observed token fits its context, while MLM asks which vocabulary item belongs in a missing position. So our systems train the discriminator with both signals. For a clean sequence x, the generator predicts masked tokens and produces a corrupted

sequence $\tilde{x}$; the discriminator then performs RTD on $\tilde{x}$ and vocabulary prediction on a separately masked version of the clean sequence. The joint loss at step t is

$$\mathcal{L}_t = \mathcal{L}_{\text{G-MLM}} + \lambda_t \mathcal{L}_{\text{RTD}} + \mathcal{L}_{\text{D-MLM}}, \quad (1)$$

where the generator MLM and discriminator MLM coefficients are fixed at one and λ_t controls the relative strength of RTD.

This formulation lets us study an important small-data question: *should dense RTD remain the dominant signal for the entire budget, or should its influence decrease once the model has already learned a useful replacement-discrimination representation?* The objective-curriculum experiments vary λ_t, or remove RTD at different stages, to test this trade-off.

3.2 Three Forms of Curriculum

Curriculum learning can intervene at different points in the training pipeline, so we distinguish three forms.

Objective curriculum. An objective curriculum changes the active losses or their relative weights: our schedules range from fixed high RTD, to staged RTD decay, to a complete transition from joint RTD+MLM training into discriminator-only MLM. This axis directly controls how strongly dense discrimination competes with vocabulary prediction in the shared encoder; complete checkpoint-level monitor-loss trajectories appear in Appendix D (Figure 4).

Corruption curriculum. A corruption curriculum changes the prediction problem itself. *Uniform token masking* samples eligible positions without considering model difficulty; *whole-word masking* (WWM) instead treats a lexical word as one masking unit, so all its subword pieces are masked together (Cui et al., 2021); and *adaptive WWM* assigns greater masking probability to words or word classes the model currently predicts poorly, directing more updates toward difficult lexical targets (Yang et al., 2023; Edman and Fraser, 2025). Our adaptive variants retain a uniform exploration component so that easier or less frequently selected words remain represented in training. The exact weighting rule is given in Appendix B.

Lilliput Curriculum additionally uses a *short-context bias* during its first stage, giving greater masking priority to utterances of at most five whitespace-delimited words while a uniform component keeps longer contexts represented, a coarse developmental prior testing whether locally constrained contexts before harder lexical targets could improve acquisition-oriented behavior, especially AoA.

Data curriculum. A data curriculum changes the order in which examples are seen. LAD Got BERT assigns each packed sequence a corpus-derived difficulty score based on lexical rarity, subword fragmentation, sentence length, and insufficient context; training begins with a stronger easy-to-hard order, weakens that preference over the next two epochs, and becomes uniformly shuffled from epoch 4 onward. Every example is still seen once per epoch, so only its presentation order changes. The full scoring function appears in Appendix C.

These axes affect different parts of learning: data curriculum controls *when* an example is seen, corruption curriculum controls *what must be predicted* within it, and objective curriculum controls *which learning signal has the most influence* on the shared representation. This separation is central to our analysis, a sophisticated masking or data policy still operates inside whatever objective balance the training procedure imposes.

4 Experimental Setup

4.1 Shared Data, Tokenization, and Backbone

All runs use the official BabyLM 2026 Strict-Small corpus (BabyLM Organizers, 2026), containing 10 million whitespace-counted words in the pinned training revision. No external teacher model, synthetic data, evaluation examples, AoA labels, or other evaluation labels are used for training, curriculum ordering, or adaptive mask selection. The maximum budget is ten corpus passes, represented as 100M cumulative word presentations. Nineteen milestone artifacts are exported: 1M through 9M, followed by 10M through 100M in 10M increments.

All five runs use a 16K SentencePiece unigram tokenizer (Kudo and Richardson, 2018) trained only on the permitted corpus and pack text into sequences of up to 128 tokens. The discriminator is a 12-layer DeBERTa-v2 encoder with hidden size 384, feed-forward size 1536, and six attention heads. A four-layer generator shares token embeddings with the discriminator. Joint-training phases use micro-batches of 8 with 16-step gradient accumulation, a peak learning rate of 5×10^{-4}, weight

decay 0.01, and seed 42. MLM-only phases use a learning rate of 2×10^{-4}. Full architecture and optimization details are provided in Appendix A.

4.2 Experimental Variants

Table 1 summarizes the five executed schedules. We describe each run by the intervention it makes and the hypothesis that motivated it.

bgb-decay: retain RTD, reduce its late dominance. The main model keeps generator MLM, discriminator RTD, and discriminator MLM active throughout all 100M exposures. Data order remains uniformly shuffled and corruption remains uniform 15% token masking. Only the RTD coefficient changes: 50 from 0–40M exposures, 25 from 40–70M, and 10 from 70–100M. The motivation is to exploit dense discrimination strongly while the encoder is forming general representations, then reduce its relative influence so that vocabulary prediction can contribute more strongly during later refinement. This is a gradual rebalancing, not an objective handoff: RTD is never removed.

Baby Got BERT: fixed high RTD reference. Baby Got BERT uses the same three-objective training structure, uniform 15% masking, and uniform data order as bgb-decay, but keeps the RTD coefficient fixed at 50 for the full budget. This provides the cleanest comparison for the objective-curriculum hypothesis. The question is whether the strong RTD pressure that is useful early in training should remain equally strong after the discriminator has already learned the replacement task.

LAD Got BERT: Baby Got BERT plus curriculum. LAD is deliberately close to Baby Got BERT on the objective axis: generator MLM, RTD, and discriminator MLM remain active throughout training with fixed RTD weight 50. It adds curriculum around that same objective mixture. The data schedule starts with an easy-to-hard ordering in epoch 1, uses weaker difficulty bias in epochs 2 and 3, and returns to uniform shuffling from epoch 4. This tests whether changing the order of evidence can improve learning without changing objective weights.

LAD also adds a later corruption curriculum only to the auxiliary discriminator MLM stream. Epoch 5 is used to collect token-level auxiliary-MLM accuracy statistics. Beginning in epoch 6, these statistics determine which MLM targets receive more emphasis: words that were predicted less accurately receive higher masking probability, while uniform exploration preserves coverage. Generator masking and RTD corruption remain uniform at 15%, so the adaptive policy does not change the replacement task itself. The original motivation was partly acquisition-oriented: first present easier examples, then use the model's own learned difficulty estimates to focus lexical practice on words that remain hard. LAD therefore asks whether data and target curricula can improve the fixed-RTD reference without changing its objective balance.

Lilliput Lexicon: early RTD scaffold followed by lexical curriculum. Lilliput Lexicon makes a stronger change to both the objective and corruption axes. The first 10M exposures use joint generator MLM, RTD, and discriminator MLM with RTD weight 10. The generator and RTD objective are then removed. From 10–20M, the discriminator trains with uniform WWM at a 20% mask target. From 20–100M, training uses adaptive WWM, with the global mask target gradually reduced from 20% to 15% and 30% of mask selection kept uniform for exploration.

The hypothesis was that RTD might be most useful as an early representation-building scaffold. After that scaffold is established, the remaining budget is devoted to complete lexical units and increasingly difficult words. This broad-to-hard-word progression was also intended to test whether a more acquisition-like lexical curriculum could improve the Human-like evaluation, especially AoA.

Lilliput Curriculum: smoother RTD introduction and withdrawal. Lilliput Curriculum combines the developmental motivation with a more gradual RTD transition. During the first 10M exposures, joint training uses RTD weight 10 together with short-context-biased masking. From 10–20M, masking becomes uniform and RTD weight increases to 20. From 20–40M, joint training continues while RTD is annealed from 20 to zero. The final 40–100M exposures use discriminator-only adaptive WWM, with the mask target reduced from 18% to 15% and 30% uniform exploration.

The $10 \rightarrow 20 \rightarrow 0$ schedule was chosen to make both the introduction and removal of RTD smoother. RTD begins at a moderate weight while training is biased toward short contexts, becomes stronger after the model moves to the full context distribution, and is then withdrawn gradually before the long adaptive-MLM phase. The short-context stage was intended as a coarse child-

learning analogy: more constrained contexts first, followed by broader contexts and harder lexical targets. The experiment therefore combines a staged objective curriculum with developmental corruption and a late hard-word MLM phase.

4.3 Evaluation Suite

We use the official BabyLM 2026 Strict-Small evaluation implementation and report both category-level scores and challenge aggregates. Because the suite measures several distinct capabilities, no single category should be treated as a complete measure of language learning.

BLiMP measures zero-shot grammatical preferences using minimal contrasts (Warstadt et al., 2020); BLiMP Supplement is the additional held-out set introduced with the first BabyLM evaluation (Warstadt et al., 2023). EWoK evaluates basic world knowledge (Ivanova et al., 2025), COMPS probes conceptual property knowledge and inheritance (Misra et al., 2023), GlobalPIQA measures physical and culturally situated commonsense reasoning (Chang et al., 2025), and Entity Tracking tests whether a model can maintain entities and their changing attributes across a short discourse (Kim and Schuster, 2023).

The challenge's (Super)GLUE aggregate draws tasks from GLUE and SuperGLUE (Wang et al., 2018, 2019). Its reported score is the unweighted mean of BoolQ accuracy (Clark et al., 2019), MNLI accuracy (Williams et al., 2018), MRPC F1 (Dolan and Brockett, 2005), MultiRC accuracy (Khashabi et al., 2018), QQP F1 as distributed with GLUE, RTE accuracy (Dagan et al., 2006), and WSC accuracy (Levesque et al., 2012). GlobalPIQA is the macro-average of its parallel and nonparallel English subsets, evaluated using length-normalized conditional likelihoods. Reading measures the additional variance in human reading behavior explained by model surprisal after controlling for lexical and contextual predictors (Frank et al., 2013; de Varda et al., 2024). Its category score is the mean of the normalized change in R^2 for self-paced reading, including one-word spillover, and the corresponding score averaged across four eye-tracking measures. AoA fits acquisition curves to word surprisals across training checkpoints and compares the resulting model ages of acquisition with child estimates derived from MacArthur–Bates CDI norms (Fenson et al., 2007; Frank et al., 2017; Chang and Bergen, 2022). Under the current evaluator, AoA is the model–child Pearson correlation, reported only when it passes the scorer's significance criterion and otherwise set to zero.

The NLP Average is the unweighted mean of BLiMP, BLiMP Supplement, EWoK, Entity Tracking, COMPS, GlobalPIQA, and (Super)GLUE. The Human-like Average is the mean of Reading and AoA, while Overall is the mean of all nine category scores. Because these metrics differ in scale, difficulty, and variance, we use the aggregates only as summaries and interpret the full task profile when comparing training strategies.

Checkpoint-level fast-subset trajectories are reported only as diagnostics in Appendix E; all model comparisons use the final full-suite scores.

5 Results and Analysis

Table 3 reports the complete official result for all five runs. We treat single-seed differences as suggestive rather than statistically established.

5.1 Answers at a Glance and Comparison Logic

The short answer to *"does curriculum help?"* is: *it depends on which curriculum and which outcome*. Table 2 states the valid comparison for each question before we inspect individual tasks. Only the first row changes one factor. The remaining rows change two or more factors and therefore characterize complete training regimes rather than isolating a mechanism.

bgb-decay has the best Overall (40.16) and NLP Average (51.04), but it does not win every task. It leads BLiMP, Entity Tracking, and COMPS; Lilliput Lexicon leads EWoK and GlobalPIQA; Lilliput Curriculum leads (Super)GLUE and Reading; LAD leads BLiMP Supplement. So bgb-decay's advantage is about breadth, not dominance: it avoids the large losses that the more specialized schedules take on elsewhere.

Table 3 also lists three baselines from the official 2026 leaderboard. bgb-decay beats the GPT-2 and Interaction baselines on NLP Average, but those systems differ in architecture and training, so the comparison cannot isolate the effect of RTD decay. We leave Baseline-gpt-bert-masked-focus (mntp) out of the aggregate comparisons, since its zero scores on Entity Tracking, COMPS, and GlobalPIQA look like missing evaluations rather than true zeros (Appendix F).

AoA is 0 for all five of our runs, because none of their model–child correlations pass the scorer's sig-

System	Training schedule	Curriculum intervention and intended test
bgb-decay	0–40M: joint, $\lambda = 50$; 40–70M: joint, $\lambda = 25$; 70–100M: joint, $\lambda = 10$. Uniform 15% token masking and uniform shuffle throughout.	Objective curriculum only. Test whether strong early RTD can be retained but gradually downweighted so that MLM gains relative influence later.
Baby Got BERT	0–100M: joint, fixed $\lambda = 50$. Uniform 15% token masking and uniform shuffle throughout.	Matched fixed-objective reference for bgb-decay. Test sustained high RTD pressure with no data or corruption curriculum.
LAD Got BERT	0–100M: joint, fixed $\lambda = 50$. Easy-to-hard ordering in epoch 1, weaker bias in epochs 2–3, uniform from epoch 4. Epoch 5 collects auxiliary-MLM accuracy; epochs 6–10 use adaptive masks only for auxiliary MLM.	Baby Got BERT plus data and auxiliary corruption curricula. Test whether acquisition-oriented ordering and adaptive lexical practice help without changing objective balance.
Lilliput Lexicon	0–10M: joint, $\lambda = 10$; 10–20M: discriminator MLM with uniform WWM at 20%; 20–100M: adaptive WWM with 20%→15% mask target and 30% uniform exploration.	Early RTD scaffold followed by increasingly difficult lexical prediction. Test whether a broad-to-hard-word progression improves lexical and acquisition-oriented behavior.
Lilliput Curriculum	0–10M: joint $\lambda = 10$ with short-context bias; 10–20M: joint $\lambda = 20$ with uniform masks; 20–40M: joint $\lambda : 20 \rightarrow 0$; 40–100M: adaptive WWM at 18%→15% with 30% exploration.	Smooth RTD introduction and withdrawal, combined with short-context-first and later hard-word training. Test a more gradual developmental RTD-to-MLM curriculum.

Table 1: Executed training schedules. Exposure values are cumulative word presentations over the 10M-word corpus. "Joint" denotes generator MLM, discriminator RTD, and discriminator MLM trained together; λ is the RTD coefficient.

nificance threshold (Appendix F). So the Human-like Average here is really just half of Reading, and we cannot say whether the three acquisition-motivated curricula actually helped AoA.

5.2 Objective Curriculum: Decay Beats a Fixed Weight

bgb-decay and Baby Got BERT are matched on everything except the RTD weight: bgb-decay steps it down $50 \rightarrow 25 \rightarrow 10$, Baby Got BERT keeps it at 50. Decay wins: +0.34 Overall, +0.46 NLP Average, and gains on six of seven NLP categories (BLiMP +0.63, Supplement +0.59, EWoK +0.30, Entity Tracking +0.63, COMPS +0.75, GlobalPIQA +0.43). Fixed RTD only wins on (Super)GLUE (+0.12) and Reading (+0.15). The gaps are small, but the direction is consistent.

The training diagnostics explain why. Generator MLM accuracy barely moves (0.4249 vs. 0.4260), but bgb-decay's discriminator-MLM accuracy is higher (0.4416 vs. 0.4334) and its replaced-class RTD F1 is lower (0.2397 vs. 0.2757), while coarse RTD accuracy stays about the same (0.9001 vs. 0.9012). This is exactly what the schedule is meant to do: pull weight away from fine-grained replacement discrimination late in training and give more of it to lexical prediction, without ever removing RTD. The LAD gradient probes (Appendix D) back this up: under a fixed weight of 50, late RTD gradients run about twice the norm of the auxiliary-MLM gradients and point in a nearly orthogonal direction, which is the imbalance decay is correcting.

5.3 Data Curriculum: LAD Shifts Capability, Doesn't Rebalance

LAD keeps Baby Got BERT's fixed RTD schedule and adds two things on top: easy-to-hard ordering in epochs 1–3 (Appendix C) and adaptive masking for the auxiliary MLM from epoch 6 (Appendix B). So the right comparison is LAD minus Baby Got BERT, not LAD minus bgb-decay.

Against that reference, LAD loses 0.27 Overall and 0.33 NLP Average. Its one clear win is BLiMP Supplement (+2.85); everything else non-AoA drops, from -2.09 (EWoK) to -0.05. So the combined LAD curriculum does not help broadly. It buys one grammar-related gain, and not even on the AoA-related behavior it was designed for. Since data order and adaptive masking changed together, we cannot tell which one caused the gain or the losses.

Question	Comparison (first minus second)	ΔOverall	ΔNLP	Observed answer
Does objective rebalancing improve observed performance?	bgb-decay − Baby Got BERT	+0.34	+0.46	Yes in this single-seed comparison: decay improves six of seven NLP categories; the sole NLP exception is (Super)GLUE (-0.12), while Reading also decreases (-0.15).
How does LAD's combined curriculum compare with fixed RTD?	LAD Got BERT − Baby Got BERT	-0.27	-0.33	It does not improve the aggregate profile: BLiMP Supplement increases by +2.85, whereas every other reported non-AoA category is lower, with differences ranging from -2.09 to -0.05.
How does the early-handoff Lilliput Lexicon regime compare with bgb-decay?	Lilliput Lexicon − bgb-decay	-0.61	-1.00	The regime shows a task-level trade-off: EWoK (+0.89), GlobalPIQA (+2.02), and Reading (+1.48) improve, whereas BLiMP (-3.89) and (Super)GLUE (-3.70) decline substantially.
How does the multi-stage Lilliput Curriculum regime compare with bgb-decay?	Lilliput Curriculum − bgb-decay	-0.18	-0.59	The regime also shows a task-level trade-off: Reading improves by +2.49, with smaller gains on Supplement (+0.42), GlobalPIQA (+0.55), and (Super)GLUE (+0.59), while BLiMP (-2.09), EWoK (-1.61), and Entity Tracking (-1.94) decline.

Table 2: Comparison map for the five systems. Deltas are percentage-point differences computed from the displayed scores in Table 3.

Model	Overall	NLP	Human	BLiMP	Supp.	EWoK	Entity	COMPS	GPIQA	SGLUE	Reading	AoA
bgb-decay	**40.16**	**51.04**	2.08	**69.82**	60.06	51.90	**19.61**	**52.63**	34.62	68.65	4.17	0.00
Baby Got BERT	39.82	50.58	2.16	69.19	59.47	51.60	18.98	51.88	34.19	68.77	4.32	0.00
Lilliput Lexicon	39.55	50.04	2.83	65.93	59.69	**52.79**	18.27	52.06	**36.64**	64.95	5.65	0.00
Lilliput Curriculum	39.98	50.45	**3.33**	67.73	60.48	50.29	17.67	52.57	35.17	**69.24**	**6.66**	0.00
LAD Got BERT	39.55	50.25	2.09	67.86	**62.32**	49.51	17.86	51.55	34.14	68.49	4.17	0.00
BabyLM GPT-2 Strict-Small	37.38	48.99	-3.26	65.23	57.25	50.63	19.10	51.81	35.09	63.80	5.63	-12.15
BabyLM Strict-Small Interaction	38.71	49.04	2.55	63.07	58.13	51.45	19.62	51.29	36.14	63.62	5.10	0.00
Baseline-gpt-bert-masked-focus (mntp)†	28.49	35.72	3.19	70.36	63.71	49.95	0.00*	0.00*	0.00*	66.02	6.38	0.00

Table 3: Complete official evaluation results for our five runs and three additional models drawn from the official BabyLM 2026 Strict-Small leaderboard. Bold marks the best value within our five-run experiment. The Baseline-gpt-bert-masked-focus (mntp)† row's Entity Tracking, COMPS, and GlobalPIQA scores of 0.00* likely reflect an unevaluated entry rather than true zero performance (Appendix F), so its Overall/NLP Average are not comparable to the other rows. Because AoA is zero in every experimental row, the Human-like Average for our five runs is determined entirely by Reading.

5.4 Corruption Curriculum: RTD Handoffs Trade Breadth for Specialization

Lilliput Lexicon and Lilliput Curriculum both drop RTD entirely and spend most of training on whole-word and adaptive masking instead. Compared to bgb-decay, Lilliput Lexicon gains on EWoK (+0.89), GlobalPIQA (+2.02), and Reading (+1.48), but loses heavily on BLiMP (-3.89) and (Super)GLUE (-3.70); Overall and NLP Average drop by 0.61 and 1.00. That is a trade, not an improvement.

Lilliput Curriculum stays closer to bgb-decay on Overall (-0.18) but is still behind on NLP Average (-0.59). It gets the biggest Reading gain (+2.49), plus smaller ones on Supplement, GlobalPIQA, and (Super)GLUE, while losing on BLiMP, EWoK, and Entity Tracking. Its trade-off is milder than Lilliput Lexicon's, but the two runs differ in more than handoff smoothness alone — short-context bias, joint-training length, RTD weight, and mask schedule all change too — so we cannot credit the smoother withdrawal by itself.

One caveat on the auxiliary-MLM accuracies in Table 6: once masking deliberately targets harder words, a lower monitor accuracy can just mean a harder target set, not a weaker model. So we rely on the held-out task scores above rather than raw monitor accuracy here.

5.5 Synthesis

Figure 1 makes the pattern easy to see, and the three axes behave differently. The objective axis (§5.2) is our one matched, single-factor comparison, and it is also the only one that moves the

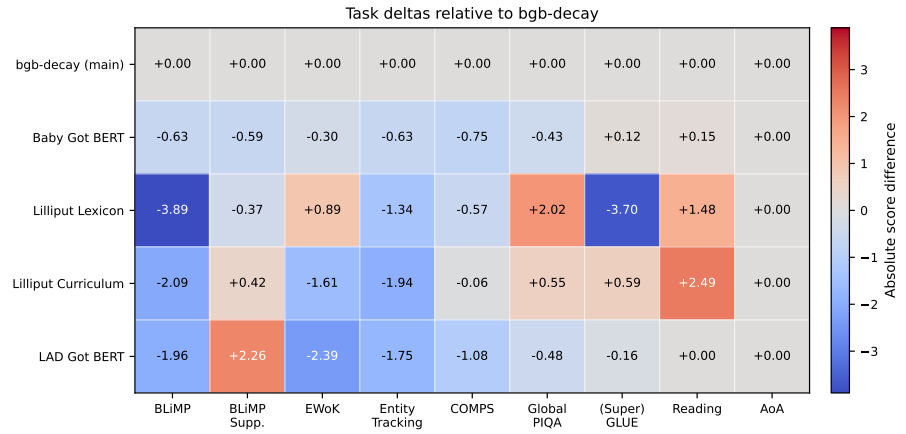


Figure 1: Signed task-score differences relative to bgb-decay. Positive cells indicate that a comparison run has a higher score. The figure emphasizes broad versus localized changes across task families rather than treating each cell as an independent finding.

aggregate broadly instead of locally. The corruption axis (§5.4) does the opposite: both Lilliput variants concentrate their gains and losses into the largest single-category swings in the study, with the smoother handoff softening that trade a little. The data axis (§5.3) sits in between: LAD redirects performance toward one grammatical category without touching the underlying RTD/MLM balance, so it moves things around without broadening them.

One more pattern, worth noting: four of the five runs achieve the highest score within our five-run comparison on at least one category, and each within-study lead aligns with the primary axis modified by that run—bgb-decay on BLiMP, Entity Tracking, and COMPS; LAD on BLiMP Supplement; and the two Lilliput models on the semantic, reasoning, and reading categories where their corruption handoffs concentrate their gains. Baby Got BERT, the only run without an additional curriculum mechanism, does not lead any category.

Our strongest claim stays narrow: between two otherwise-matched joint RTD+MLM runs, reducing RTD after early training gives the better observed balance. The other systems are useful pointers for where to look next, but since each changes several things at once, we cannot say which factor drove its task-specific gains. A natural next step is a factorial design that starts from the decayed schedule and adds data ordering and adaptive masking one at a time, across multiple seeds.

6 Conclusion

We studied five BabyLM Strict-Small pretraining runs through three curriculum axes: objective scheduling, corruption policy, and data order. This framing turns the runs into a coherent set of comparisons rather than a collection of unrelated submissions. bgb-decay changes only the objective balance, Baby Got BERT provides the matched fixed-RTD reference, LAD adds data and auxiliary masking curricula without changing that fixed objective schedule, and the two Lilliput models test stronger RTD-to-MLM transitions with whole-word and adaptive masking.

The clearest result is on the objective axis. In our matched pair, gradually decaying RTD improves Overall by 0.34 points and NLP Average by 0.46, with gains on six of seven NLP categories. The diagnostics sharpen this: the decayed model has better discriminator-MLM accuracy, while the fixed model keeps a stronger RTD F1. So a better task balance does not require maximizing the RTD metric itself — though multi-seed replication is still needed to call this gain stable.

The other variants do not support a general "curriculum helps" claim. LAD's combined data-order and masking curriculum improves BLiMP Supplement but hurts both aggregates relative to its fixed-RTD reference. The Lilliput models trade broad performance for gains on specific knowledge, reasoning, reading, or transfer tasks, and each one changes too many factors at once to say why. The natural next step is to keep the decayed objective schedule and add each curriculum axis on its own, under multiple seeds.

Limitations

The study has three main limitations. First, we trained each configuration once, and the five systems were built one after another rather than as a full factorial design. The fixed-vs-decay pair is closely matched, but Lilliput and LAD change several things at once, so their results describe training regimes rather than isolate single causes. We would need multi-seed replication before treating these small score gaps as stable effects.

Second, we settled on bgb-decay only after seeing all five result profiles, and every experiment uses one compact DeBERTa-v2 setup, one corpus, and one training budget. So these findings are hypotheses about objective allocation in this setting, not a universal recipe. The monitor metrics and gradient probes here only support our interpretation; they are not a substitute for held-out evaluation.

Third, the Baseline-gpt-bert-masked-focus (mntp) row in Table 3 comes from the 2026 leaderboard, and its Entity Tracking, COMPS, and GlobalPIQA scores were never re-evaluated under the updated task suite. We treat it as informal context rather than a controlled baseline, and note that it actually beats bgb-decay on BLiMP Supplement and Reading.

References

BabyLM Organizers. 2026. BabyLM 2026 Strict-Small training corpus. Hugging Face dataset. Accessed 2026-09-07.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48.

Tyler A. Chang, Catherine Arnett, Abdelrahman Eldesokey, Abdelrahman Sadallah, Abeer Kashar, Abolade Daud, Abosede Grace Olanihun, Adamu Labaran Mohammed, et al. 2025. Global PIQA: Evaluating physical commonsense reasoning across 100+ languages and cultures. *arXiv preprint arXiv:2510.24081*.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. BabyLM Turns 4 and Goes Multilingual: Call for Papers for the 2026 BabyLM Workshop. *Preprint*, arXiv:2602.20092.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2924–2936.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*, pages 177–190.

Andrea G. de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 english sentences, aligned with existing EEG and reading time data. *Behavior Research Methods*, 56(5):5190–5213.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*, pages 9–16.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*, pages 445–453, Suzhou, China. Association for Computational Linguistics.

Larry Fenson, Virginia A. Marchman, Donna J. Thal, Philip S. Dale, J. Steven Reznick, and Elizabeth Bates. 2007. *MacArthur–Bates Communicative Development Inventories: User's Guide and Technical Manual*, 2nd edition. Paul H. Brookes Publishing.

Michael C. Frank, Mika Braginsky, Daniel Yurovsky, and Virginia A. Marchman. 2017. Wordbank: An open repository for developmental vocabulary data. *Journal of Child Language*, 44(3):677–694.

Stefan L. Frank, Irene Fernandez Monsalve, Robin L. Thompson, and Gabriella Vigliocco. 2013. Reading time data for evaluating broad-coverage models of english sentence processing. *Behavior Research Methods*, 45(4):1182–1190.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

Xudong Hong, Sharid Loáiciga, and Asad Sayeed. 2023. A surprisal oracle for active curriculum language modeling. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 259–268.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, et al. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking

beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 252–262.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 3835–3855.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 552–561.

Evan Lucas, Dylan Gaines, Tagore Rao Kosireddy, Kevin Li, and Timothy C. Havens. 2024. Using curriculum masking based on child language development to train a large language model with limited training data. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 221–228.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949.

Miyu Oba, Akari Haga, Akiyo Fukatsu, and Yohei Oseki. 2023. BabyLM challenge: Curriculum learning based on sentence complexity approximating language acquisition. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 290–297.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1162–1172.

Suchir Salhan, Richard Diehl Martinez, Zébulon Goriely, and Paula Buttery. 2024. Less is more: Pre-training cross-lingual small-scale language models with cognitively-plausible curriculum learning strategies. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 174–188.

Loris Schoenegger, Lukas Thoma, Terra Blevins, and Benjamin Roth. 2025. Influence-driven curriculum learning for pre-training on limited data. In *Proceedings of the First BabyLM Workshop*, pages 356–379.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*, pages 353–355.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1112–1122.

Dongjie Yang, Zhuosheng Zhang, and Hai Zhao. 2023. Learning better masking for better language model pre-training. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7255–7267, Toronto, Canada. Association for Computational Linguistics.

A Architecture and Optimization Details

B Adaptive Whole-Word Masking

Lilliput Lexicon and Lilliput Curriculum maintain prediction counts for token types observed at masked positions. With c_v correct and i_v incorrect predictions, smoothed accuracy is

$$\hat{a}_v = \frac{c_v + 0.5}{c_v + i_v + 1}. \tag{2}$$

A hard-token proposal weight is

$$\tilde{w}_v = \max\left(w_{\min},\ p_t(1 - \hat{a}_v)\right), \tag{3}$$

where p_t is the current global mask target and $w_{\min} = 0.005$. The implementation updates

$$w_v \leftarrow 0.2w_v + 0.8\tilde{w}_v, \tag{4}$$

normalizes weights to the target average mask rate, caps per-position probability at 0.95, and mixes the adaptive distribution with 30% uniform selection. Whole words are recovered from tokenizer word-group metadata so all subpieces of a selected lexical item are masked together. The exploration component prevents the model from permanently ignoring already-easy words and reduces feedback loops caused by early prediction noise.

LAD Got BERT applies the same general difficulty principle only to the auxiliary discriminator MLM pass. Generator corruption and RTD retain uniform 15% masks. The executed run keeps 16% uniform exploration and updates adaptive weights every 100 optimizer steps. Thus LAD changes the target distribution of one objective without changing the data seen by RTD.

C Corpus-Derived Difficulty Curriculum

LAD computes an ordering score from the training corpus only; the score is removed before the neural forward pass. For normalized word v with frequency f_v and corpus maximum $f_{\max}$,

$$r(v) = 1 - \frac{\log(1 + f_v)}{\log(1 + f_{\max})}. \tag{5}$$

Sentence rarity is the mean of $r(v)$. With k SentencePiece tokens per whitespace word, fragmentation is

$$f = \mathrm{clip}\left(\frac{k - 1}{3}, 0, 1\right). \tag{6}$$

For a sentence of n words, the length term is

$$\ell = \min\left(\frac{\log(1 + n)}{\log 31}, 1\right). \tag{7}$$

The insufficient-context penalty $c(n)$ is 1.0 for zero or one word, 0.6 for two, 0.3 for three, and 0 for four or more. The final score is

$$d = 0.45\,\bar{r} + 0.20f + 0.20\ell + 0.15c(n). \tag{8}$$

Packed-sequence difficulty is the word-count-weighted average of constituent sentence scores. Epoch 1 traverses ten approximately equal-word buckets from easy to hard. Epoch 2 orders by $0.5d + 0.5u$, epoch 3 by $0.2d + 0.8u$, where $u \sim \mathrm{Uniform}(0, 1)$, and epochs 4–10 use a uniform shuffle. Every example appears once per epoch; difficulty controls presentation order, not sampling frequency or model input features.

D Checkpoint-Level Training Diagnostics

The handoff variants make the objective transition explicit: their RTD trajectories end when the generator/discriminator corruption stream is removed. Baby Got BERT, bgb-decay, and LAD retain RTD through 100M exposures, but only bgb-decay reduces its coefficient. bgb-decay maintains nearly the same coarse RTD accuracy as the fixed-RTD reference at the final checkpoint, but its replaced-class F1 is lower. Thus decay preserves the binary task at a broad level while reducing late emphasis on fine-grained replacement discrimination.

The fixed-versus-decay pair is directly comparable because both retain uniform masking. Their final generator MLM accuracies are almost identical, while bgb-decay has higher auxiliary discriminator-MLM accuracy and lower RTD F1. This is the monitor signature expected from reallocation: late capacity shifts toward vocabulary prediction without removing RTD. Absolute comparisons with the adaptive runs are inappropriate because their mask distributions become harder over time.

Checkpoint-level monitor losses. Figure 4 provides the corresponding objective-specific loss trajectories. These values are computed on the fixed training-corpus monitor sequences at the 19 saved milestones. The total monitor loss combines the objectives using the coefficients active in each system, so its absolute magnitude is not directly comparable across systems or across objective handoffs.

For LAD Got BERT, ten late-stage gradient records yield mean weighted final-layer norms of 2.764 for RTD and 1.353 for auxiliary MLM; mean cosine similarity is 0.009. Raw records are included in `data/lad_gradient_diagnostics.csv`.

System	Vocab	Max len.	D layers	G layers	Hidden	Heads	Microbatch	Accum.
bgb-decay	16K	128	12	4	384	6	8	16
Baby Got BERT	16K	128	12	4	384	6	8	16
Lilliput Lexicon	16K	128	12	4	384	6	8	16
Lilliput Curriculum	16K	128	12	4	384	6	8	16
LAD Got BERT	16K	128	12	4	384	6	8	16

Table 4: Architecture and batching for the executed runs. All tokenizers are 16K SentencePiece unigram models trained on the permitted corpus.

System	Joint LR	MLM-only LR	Weight decay	Base mask rate	Additional details
bgb-decay	5×10^{-4}	–	0.01	0.15	RTD 50 for epochs 1–4, 25 for 5–7, 10 for 8–10
Baby Got BERT	5×10^{-4}	–	0.01	0.15	RTD 50 throughout all 10 epochs
Lilliput Lexicon	5×10^{-4}	2×10^{-4}	0.01	0.15	Adaptive uniform mix 0.30; WWM target 0.20→0.15
Lilliput Curriculum	5×10^{-4}	2×10^{-4}	0.01	0.15	Adaptive mix 0.30; WWM target 0.18→0.15
LAD Got BERT	5×10^{-4}	–	0.01	0.15	Adaptive auxiliary masks from epoch 6; update every 100 steps; uniform exploration 0.16

Table 5: Optimization hyperparameters. All runs use seed 42. Joint phases use generator and auxiliary discriminator MLM coefficients of one. Complete phase schedules are merged into Table 1 in the main paper.

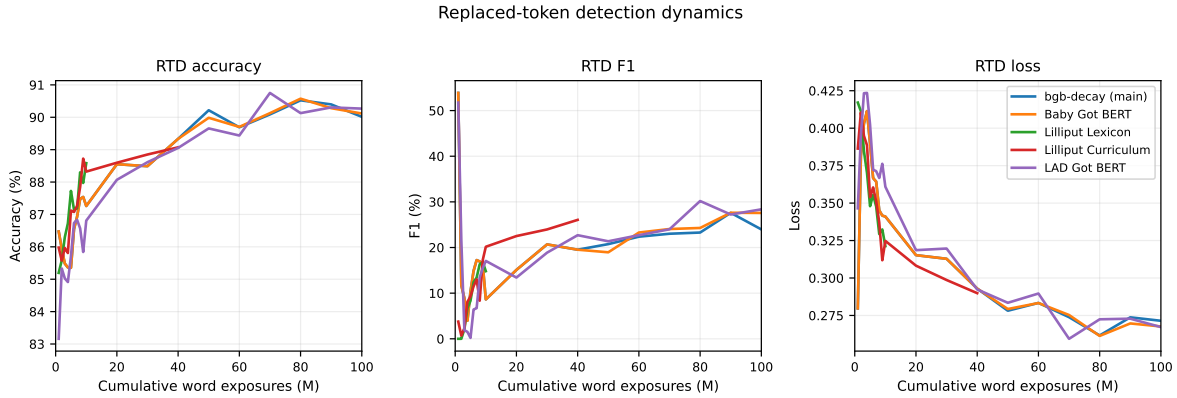


Figure 2: Replaced-token detection monitor trajectories. Lines end when RTD is disabled; an inactive objective is not plotted as zero performance. Accuracy is high partly because original tokens are the majority class, so F1 and loss are shown alongside it.

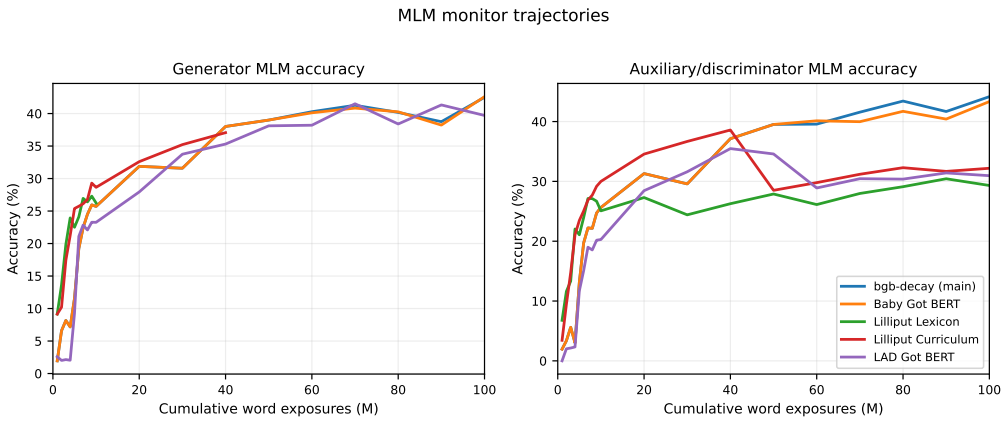


Figure 3: Generator and auxiliary/discriminator MLM accuracy on fixed training-corpus monitor sequences.

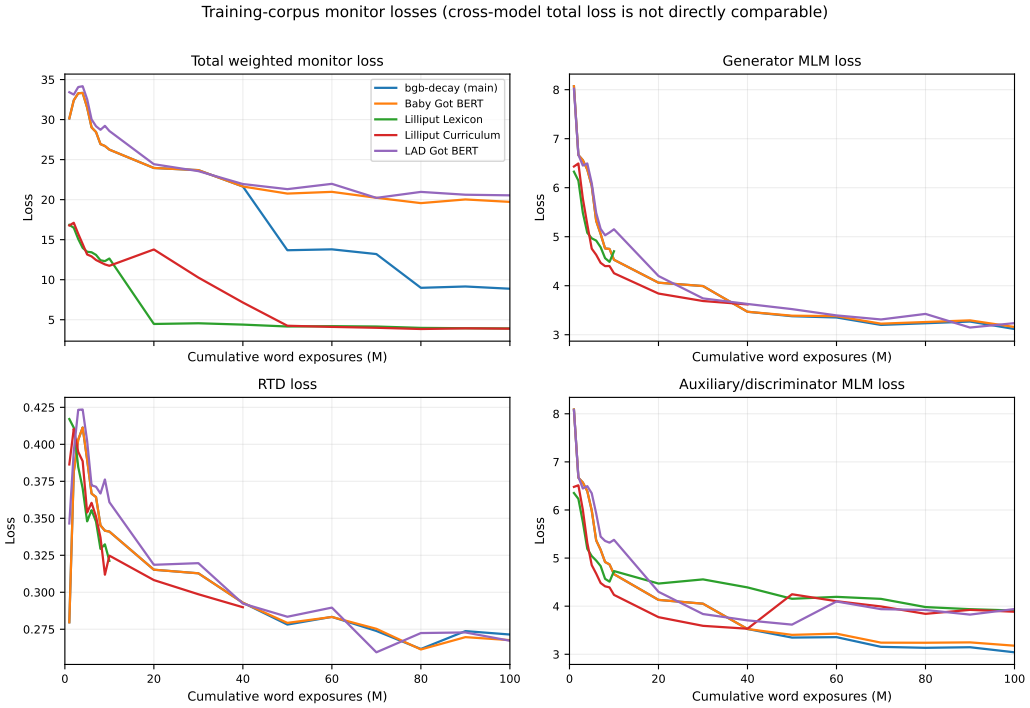


Figure 4: Checkpoint-level training-corpus monitor losses parsed from the 19 saved milestones for each model. Lines connect the recorded checkpoint values without smoothing. Zero-valued generator or RTD records are omitted after the corresponding objective is disabled. Because the systems differ in active objectives and loss coefficients, the total weighted monitor loss should not be compared directly across all models.

Final available checkpoint	Exposure	Aux MLM acc.	G MLM acc.	RTD acc.	RTD F1	RTD weight
bgb-decay	100M	0.4416	0.4249	0.9001	0.2397	10
Baby Got BERT	100M	0.4334	0.4260	0.9012	0.2757	50
Lilliput Lexicon	100M	0.2932	–	–	–	0
Lilliput Curriculum	100M	0.3219	–	–	–	0
LAD Got BERT	100M	0.3095	0.3971	0.9027	0.2836	50

Table 6: Final monitor diagnostics parsed from the supplied logs. Dashes denote objectives inactive at the final checkpoint. These are training-corpus diagnostics, not held-out scores.

E Checkpoint-Level Fast-Evaluation Diagnostics

The validated submission JSON files contain fast-subset results for all 19 milestones, from 1M through 100M cumulative word exposures. Figure 5 visualizes these checkpoint-level previews. They are useful for inspecting when particular capabilities emerge during training, but the fast subsets are smaller and noisier than the final evaluation suite. Accordingly, we do not use these trajectories for model ranking, effect-size claims, or checkpoint selection.

The plotted values are category means computed from the per-subtask fast scores stored in the validated submission JSON. Fast EWoK is omitted because the supplied local fast-collation branch does not implement the canonical context-plus-target comparison. All conclusions in the main paper therefore rely on the final full-suite results in Table 3.

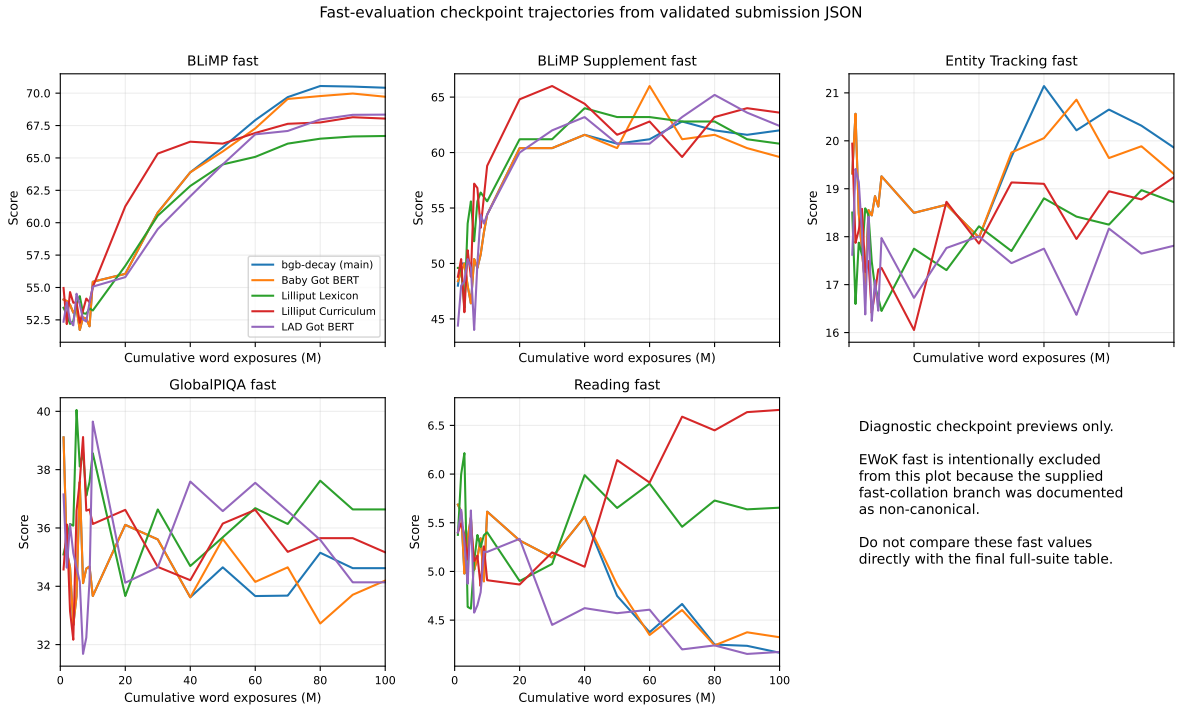


Figure 5: Diagnostic fast-evaluation trajectories across the 19 saved checkpoints. These curves show BLiMP, BLiMP Supplement, Entity Tracking, GlobalPIQA, and Reading fast-subset scores stored in the validated submission JSON. They are checkpoint previews rather than final full-suite results and should not be used for model ranking. Fast EWoK is omitted because the available local fast-collation branch is non-canonical.

F Evaluation Artifact Integrity

Using the official BabyLM 2026 evaluation implementation, the evaluation notebooks validate a native DeBERTa-v2 masked-LM artifact, check all 19 required checkpoint directories, rerun the updated Entity Tracking, GlobalPIQA, and AoA tasks, and collate the canonical result JSON. Model 5 was evaluated from scratch; Models 1–4 reused unchanged raw outputs when available and reran the updated tasks.

Several logs reach an optional local score-preview import that fails because the local `utils` module is not found. This occurs after successful preflight, canonical collation, and saved result JSON, so we treat it as a preview-path defect rather than a failed evaluation artifact.

The official AoA implementation changed during development, so earlier high AoA outputs are not retained. The validated submissions contain the current `aoa_surprisals` field. This distinction matters because the current leaderboard assigns AoA zero to legacy submissions that omit this field and therefore require re-evaluation. That compatibility rule does not explain the present results: all five runs were successfully collated with the new field. Instead, each run retained more than three valid fitted words but returned the failure-form diagnostic without a p-value, which identifies the evaluator branch used when the model–child AoA correlation has $p > 0.1$. Table 7 reports the resulting fitting coverage. These counts measure the number of words for which both model and child acquisition curves could be fitted and should not be interpreted as additional performance scores.

Model	Loaded	Fitted	AoA
bgb-decay	504	42	0.00
Baby Got BERT	504	25	0.00
Lilliput Lexicon	504	248	0.00
Lilliput Curriculum	504	241	0.00
LAD Got BERT	504	104	0.00

Table 7: AoA fitting coverage under the current evaluator. "Loaded"/"Fitted" are CDI word counts. All runs retain more than the minimum three valid words, but their model–child correlations do not pass the evaluator's $p \leq 0.1$ reporting threshold; counts are diagnostic coverage, not performance scores.

Tokens of Meaning: Semantically-Informed Morphological Tokenization in Low-Resource Language Modeling

Juan Diego Rodriguez
Department of Computer Science
The University of Texas at Austin
juand-r@utexas.edu

Abstract

Subword tokenizers for language models are typically trained to efficiently encode text into short sequences of reusable units. These often produce tokens which are not linguistically meaningful, such as `giraffe→g ira ffe`. We ask whether a tokenizer that instead splits words into semantically meaningful morphemes can help a language model learn under a developmentally plausible data budget, where every training token is scarce and sharing evidence across related word forms should matter most. We build such a tokenizer for English by hand-curating a word-to-morpheme-segmentation dictionary, built on top of MorphyNet, a database of derivational and inflectional morphology. Each word is only split into morphemes when the pieces are productive and the whole word's meaning is semantically related to them, thus splitting `unhappiness→␣un happy ness`, but keeping `business` whole. On the 10M-word BabyLM corpus this holds out-of-vocabulary tokens to 1.0%, with an inventory of ∼37k tokens. We compare the tokenizer intrinsically against both statistical (BPE, Unigram, WordPiece) and morphology-aware (Morfessor, MorphPiece) baselines, and measure its effect on a child-scale language model. In a controlled comparison against an uncurated baseline built from the same morphological source, curation yields +1.4 BLiMP. Our submitted model, which adds masked next-token prediction and vocabulary pruning, reaches 72.53 BLiMP and ranks second on COMPS and third on EWoK on the BabyLM 2026 strict-small leaderboard.[1]

1 Introduction

Remarkably successful though LLMs may be at producing fluent and felicitous language, they require orders of magnitude more training data than a human encounters in a lifetime. A child, on the other hand, acquires the ability to communicate in their native language from an exposure to only tens of millions of words (Warstadt et al., 2023). This gap in data efficiency is the motivation behind the BabyLM Challenge (Warstadt et al., 2023), a competition on language modeling with a developmentally-plausible budget (10M or 100M words). Recent work at this scale finds that tokenization is not a neutral preprocessing step, but can also yield gains on par with other modeling choices (Alqahtani et al., 2026; Altintas et al., 2025). Reduced or restructured vocabularies change what a small model can learn, and can yield improvements in downstream tasks (Muckatira et al., 2024; Finke et al., 2025).

We take inspiration from the importance of morphology in human language learning and use. Children readily recognize and extend morphological regularities. Shown a nonce word, they inflect it without hesitation (one wug, two wugs; Berko, 1958). Knowing *shelter*, a learner also has *sheltering*, *sheltered*, and *shelters*; and we grasp that slithy toves denotes more than one tove that is slithy, though we have seen neither word before. A language model could exploit the same structure. With a 10M-word budget, where exposure to any single word form is thin, **related words can draw strength from the morphemes they share**. Every occurrence of *walked* or *walking* also reinforces *walk*. We build this inductive bias directly into the tokenizer.[2]

Standard subword tokenizers do not do this. BPE (Sennrich et al., 2016), WordPiece (Wu et al., 2016), and Unigram (Kudo, 2018) are all trained to compress a corpus under a vocabulary size bud-

[1] Our code is available at https://github.com/juand-r/morpheus-tokenizer.

[2] We do not model the learning of morphological segmentation; we treat morphology as given and ask how much is gained when the learner starts with it. Bölücü and Can (2025) instead experiment with a curriculum that gradually introduces morphological segmentation during training, with mixed, task-dependent results.

Proceedings of the Second BabyLM Workshop, pages 875–891
October 29, 2026 ©2026 Association for Computational Linguistics

get, and the segments they induce often disregard morpheme boundaries. The result is segments with no semantic meaning and that fail to align related words: a purely statistical split may render `adder` as `ad der`, split *singleton* as `sing leton`, or tokenize *unhappiness* and *happiness* so that they have no tokens in common.

A growing body of work documents the downstream cost of this morphological blindness, for derivational generalization (Hofmann et al., 2021, 2022), for the reliability of model behavior (Rust et al., 2021), and for morphologically-rich languages (Schwartz et al., 2020). When a tokenizer is made morphology-aware the payoff is largest in the data-scarce and morphologically-rich regimes—from child-scale BabyLM models (Bölücü and Can, 2025; Fusco et al., 2025) to Kinyarwanda (Nzeyimana and Niyongabo Rubungo, 2022), Hebrew (Gueta et al., 2023), and the morphologically complex languages of Asgari et al. (2025).

How much does a small language model gain when its tokenizer performs morphological splits that are both **productive** and **semantically related**? It is not enough to split on morpheme boundaries: an aligned-but-meaningless split (`add er`) injects a false training signal, teaching the model that *adder* (Vipera berus, a snake) is a kind of adding, and polluting the embeddings of both pieces. A useful morphological split must satisfy two conditions: (1) the pieces must be genuinely reused across the lexicon (*productive*), and (2) the pieces must be semantically related to the word they build. We maximize how much evidence morphemes pool, but only over segmentations that could potentially help the model learn new meanings. A prior BabyLM submission pursued morphological tokenization but did not enforce this constraint (Bölücü and Can, 2025).

Beyond the modeling gain, there is an independent, computational-linguistic reason to want morphologically correct tokenization: it produces tokens that are morphemes, making the model's vocabulary interpretable and its behavior on morphological phenomena easier to analyze (Hofmann et al., 2021; Batsuren et al., 2022). A tokenizer whose units are real morphemes provides a useful instrument for studying morphological phenomena with language models.

Contributions As our entry to the BabyLM 2026 Challenge, this paper contributes: (1) an explicit, documented criterion for when and how to split a word into morphemes—productivity *and* semantic compositionality—stated formally enough to expose which parts could be automated (Section 3); (2) a publicly-available, hand-curated, lemma-based morphological tokenizer for English built on this criterion over MorphyNet (Batsuren et al., 2022), which holds corpus OOV to 1.0% with no subword fallback while collapsing related forms onto shared stems; (3) an intrinsic comparison of this tokenizer against five automatic and semi-automatic tokenizers (BPE, Unigram, WordPiece, Morfessor, MorphPiece) on identical data (Section 5.1); (4) a controlled downstream study (fixing the model and data, only varying the tokenizer) of the tokenizer's effect on a child-scale language model (Section 5.2); and (5) our BabyLM 2026 competition submission (Section 5.3).

Our submitted model (Morpheus)[3] ranked second on COMPS (54.67), third on EWoK (53.2), seventh on BLiMP (72.53), and seventh on GlobalPIQA (38.65) on the BabyLM 2026 strict-small leaderboard. We release the tokenizer and every comparison artifact.

2 Related Work

Standard subword tokenizers such as BPE (Sennrich et al., 2016), WordPiece (Wu et al., 2016) and Unigram (Kudo, 2018) optimize compression, not morphology. Unigram aligns with morphology better than BPE (Bostrom and Durrett, 2020; Vemula et al., 2025), yet it still fragments words across meaning (`singleton→single t on`) and never unifies an inflectional paradigm onto one unit. Morphology-aware tokenizers instead segment on morphemes. However, unsupervised (Morfessor; Smit et al., 2014), lexicon-plus-fallback (MorphPiece; Jabbar, 2023), decode-time (FLOTA; Hofmann et al., 2022), merge-constrained (MorphBPE; Asgari et al., 2025), or byte-level (MYTE; Limisiewicz et al., 2024) all apply morphology automatically. Closest to us, two BabyLM submissions (Bölücü and Can, 2025; Fusco et al., 2025) train child-scale models on automatically segmented vocabularies with no semantic constraint. We argue that *how* the split is chosen, not merely that morphology is used, should matter. Gains from morphology-aware to-

[3] https://huggingface.co/juand-r/morpheus-10M-v5

kenization recur across morphologically rich languages (Toraman et al., 2023; Nzeyimana and Niyongabo Rubungo, 2022; Gueta et al., 2023; Schwartz et al., 2020). We give the full discussion of related work in Appendix E.

3 Problem Formulation

Given a word, do we decompose it into morphemes or keep it atomic, and if we split, how? Here we state our procedure, both to document the method and to expose which parts could be automated. We applied this procedure semi-manually, assisted by Claude Code and always verifying the results.

3.1 The objective

We want our tokenizer to serve a model that learns one embedding per morpheme, such that a word's meaning and distribution should be **related to, and shared with, its pieces**. A split is good exactly when that sharing helps the model and bad when it misleads it. Two forces are in tension:

- **Economy / reuse** (push to split): decomposing onto shared morphemes shrinks the vocabulary and lets evidence for a morpheme pool across every word containing it (e.g., `seem` gets evidence from *seemed/seeming/seems*).
- **Semantic integrity** (push to keep whole): a split forces the model to explain the word from its pieces. If the pieces do not carry the word's meaning, the split injects a noisy training signal; e.g., it teaches the model that *business* is a kind of *busy-ness*, polluting `busy` with associations with commerce it should not carry.

For a word w with a candidate segmentation $s(w) = [m_1, \ldots, m_n]$ ($n \geq 2$), we accept the split if (P) every piece is productive and (S) the pieces are semantically related to the word; otherwise we keep w atomic. When more than one valid segmentation exists, we take the one that pushes as much of the word onto shared morphemes as the two conditions allow. We maximize morpheme reuse, but only over segmentations whose every piece is a real shared unit and whose pieces stay related to the word's meaning. Metrics that reward raw boundary alignment (MorphScore Arnett and Bergen, 2025) or subword productivity, such as the reuse measure ρ used by Parra (2024), optimize the sharing objective without the constraint.

Condition (P): productivity / reuse Every piece m must be a productive morpheme: genuinely shared, not a fragment invented for one word. This can happen in the following three ways:

(1) m is a closed-class **affix** in the recognized inventory (`un-`, `-ness`, `-ic`, `-ion`, `'s`, ...), or

(2) m is a **free word** (attested standalone), or

(3) m is a **bound root that recurs** — it appears as a piece in at least 2 other words' segmentations: *nutrit* recurs across *nutrition/nutritious/nutritive*; *mester* across *semester/trimester*. A piece used for a single word fails.

A piece used in exactly one word buys no sharing; it just fragments a word into *fragment* + *affix* and adds a dead vocabulary row.

Condition (S): semantic relatedness The pieces must be genuinely related to the word's meaning, so that the split does not create a false or misleading signal. There are four ways (S) can fail:

S1 — Unrelated meaning. Is w's dominant modern sense genuinely related to the pieces' senses? *crier* = one who cries (✓); *suburbia* = the realm of suburbs (✓); but *business* is not the state of being busy, so we keep it atomic, and likewise for *legitimate*, *livery*, *apartment*.

S2 — False etymology. The pieces may be real morphemes yet the connection is coincidental, not synchronically true: *adder* is a snake, not `add er`; *polish* is not `pole ish`; *glower* is not `glow er`.

S3 — Incorrect homonym sense. If the base is a homonym (one form, unrelated senses), split only when the sense the word uses is the one the shared token would usually carry: *limestone* is kept atomic because its *lime* is the mineral, not the far more common citrus fruit.

S4 — Pollution by collision. If a piece is spelled like a much more frequent, unrelated token, splitting pours w's semantics into it: *werewolf* is not `were wolf` (the morpheme *were-* collides with the copula *were*); *singleton* is not `single ton` (diminutive *-ton* collides with the weight unit). We keep these atomic.

$S1$–$S4$ are semantic judgments. These judgments are our contribution over automatic morphological

Trigger	Fails	Examples (kept atomic)
Lexicalized / unrelated meaning	S1	*business*, *legitimate*, *livery*
False etymology (real pieces, wrong sense)	S2	*adder*, *polish*, *tally*, *glower*, *carter*
Dead / deep etymology (only historical)	S2	*werewolf*, *remember*
Wrong-sense homonym	S3	*limestone*
Collision with frequent unrelated token	S4	*singleton* (*ton*), *werewolf* (*were*)
Phantom / non-recurring piece	P	*cranberry*
Proper name, common-noun use less frequent	entry	*hayward*, *hooper*
Genuine ambiguity	entry	*leaves*

Table 1: Keep-atomic triggers, indexed to the failing condition, with real cases decided during curation.

tokenizers. Possible automatic signals such as distributional relatedness in embedding space (*business* sits far from *busy*), definitional/gloss overlap, and frequency-collision checks may help reduce some of the annotation work in the future.

3.2 Decision procedure

Figure 1 shows the decision procedure. A candidate split must satisfy every condition in order to be accepted.

Two special cases keep a word atomic before these checks: **genuine ambiguity**, where a form has two valid analyses that discard information (*leaves* = *leaf*+s or *leave*+s); and **proper names** with no common-noun use in the corpus (*hayward*, *hooper*), whose split would attach name occurrences to unrelated common words.

3.3 How to split, once accepted

Once a split is accepted, we use a few conventions to normalize it. We emit canonical, lemma-level morphemes rather than surface substrings, treating orthographic realization as a separate layer in order to collapse spelling variants (e.g., `-ise`/`-ize`, British/American forms, elisions) onto one shared morpheme. The same convention absorbs *allomorphy* (the different surface shapes a single morpheme can take) so for example the plural allomorphs `-s`/`-es`/`-ies` map to one `-s`, and the negative prefix `in-`/`im-`/`il-`/`ir-` to one `in-`. A model that would otherwise spread its evidence across each variant instead treats these as a single unit. Beyond spelling, we prefer a free word or a recurring root to an invented bound stem (keeping *legitimate* whole rather than minting *legitim*, but allowing *nutrit* and *epidem* because they recur across a family); and we drop a meaningless linking element (interfix) rather than tokenize it (e.g., *cosmonaut*→`cosm naut`, dropping the *-o-*), except where dropping it would cause a collision.

4 Method

We fix one model architecture, one training corpus, one training recipe, one random seed, and one evaluation suite, and we swap only the tokenizer.[4] Every result difference we report is therefore attributable to how the corpus is tokenized.

4.1 Shared model and training configuration

We train a Llama-style decoder-only Transformer (Touvron et al., 2023)[5] from scratch under the standard autoregressive (causal) language-modeling objective. The configuration is held identical across conditions: hidden size 768, 12 layers, 12 attention heads, intermediate size 3072, a context length of 256 tokens, rotary position embeddings (Su et al., 2021), tied input and output embeddings, and attention dropout 0.1. We train for 10 epochs over the corpus. Optimization is done with Muon (Jordan et al., 2024) (learning rate 0.04) on the two-dimensional weight matrices, with an AdamW (Loshchilov and Hutter, 2019) auxiliary optimizer (learning rate 1×10^{-3}, weight decay 0.3) for the embeddings and normalization parameters. This autoregressive configuration is used for the controlled comparison in Section 5; the submitted model for the BabyLM competition (Section 5.3) adds masked-next-token-prediction (MNTP; BehnamGhader et al., 2024; Charpentier and Samuel, 2024) and vocabulary pruning (dropping morphemes seen fewer than three times in the corpus) on top.

The learning-rate schedule, batch size, epoch count, seed, and every other hyperparameter are the same in every run. The only quantity that changes across conditions is the vocabulary (and hence the number of rows in the embedding table) which is a property of the tokenizer.

4.2 Data and preprocessing

We use the 10M-word English corpus from the BabyLM 2026 `strict-small` track. We train on a cleaned version (9,732,593 whitespace-delimited words, produced by dropping CHILDES

[4] Unavoidably, this also changes the embedding matrix.

[5] Using HuggingFace's `LlamaForCausalLM`

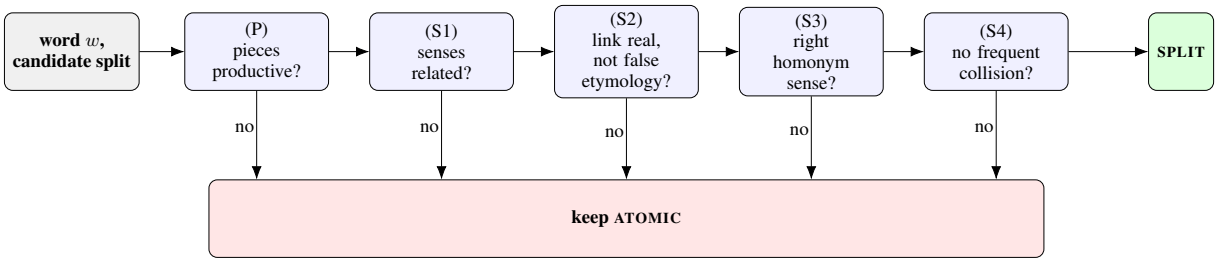


Figure 1: Decision procedure: a candidate split is accepted only if it satifies conditions (P) and (S1)–(S4). Failing any one keeps the word atomic. Two further cases keep a word atomic in addition to these: genuine ambiguity (e.g. *leaves*) and proper names with no common-noun use (e.g. *hayward*).

transcriber comments and fixing encoding garble and OCR artifacts in OpenSubtitles and Simple Wikipedia. The corpus is lowercased and all tokenizers use whitespace pre-tokenization.

4.3 Building the tokenizer

Our tokenizer is built on MorphyNet's morphological database (Batsuren et al., 2021), modifying and extending it as described below. The MorphyNet tables for derivational and inflectional morphology are used differently because their error rates differ sharply.

Inflectional morphology (grammatical endings: *plays, played, playing*). MorphyNet's inflectional table is clean enough to load directly; only ∼988 forms needed repair: splits that left a non-word stem (*seemed*→`seeme d`) or forms with competing analyses that needed disambiguation (*number* parsed as `numb er`). These were found with rule-based detectors and fixed manually.

Derivational morphology (word-formation: *nation*→*national*). MorphyNet's derivational table records historical etymology, not modern structure, so it is too noisy to load directly. For example, it proposes *music*=`muse ic`, *king*=`kin ing`, *remember*=`re member`. We use it only as a source of candidates to be validated, and supplement it with independently annotated words.

As both an independent check on MorphyNet and as a source of derivational morphological splits, we hand-segmented 22,137 high-frequency words drawn primarily from LDOCE (Mayor, 2009) headwords and their definitions. This hand verification rejected ∼53% of the MorphyNet derivational proposals these words overlapped with, rescuing ∼572k corpus tokens from etymological over-splitting.

For the MorphyNet derivational words the hand-segmented vocabulary didn't cover, the next most frequent 7,600 words in MorphyNet's derivational table were audited by parallel LLM agents: a 1,000-word sample found an ∼11% error rate; after fixing the flagged entries, a fresh 1,200-word sample found ∼5%.

Additional words Words the BabyLM corpus contains but MorphyNet does not cover, such as run-together words (*sportclub*→`sport club`), encoding garble, typos, elisions, and approved rare words, were segmented by hand with LLM assistance.

Curation and validation The curation workflow throughout consisted of LLM agents (Claude Code) reviewing each candidate adversarially, and the author making every final decision. As an independent check of quality, we compared our segmentations against MorphoLex (Sánchez-Gutiérrez et al., 2018), an expert-built derivational database of ∼70,000 English words (Appendix H): the two resources agree on ∼50% of words; we under-segment relative to MorphoLex on ∼38%, but these are almost entirely etymological decompositions into bound Latin/Greek roots that we deliberately keep whole; compared to MorphoLex we over-segment on only ∼1%.

Tokenizer statistics In total the tokenizer records an explicit decision for 28,571 words on top of MorphyNet: ∼10,000 substantively change it (1,114 splits rejected, 3,933 re-segmented, 4,961 added); the remainder confirm a MorphyNet split or keep the word whole.

4.4 Using the morphological tokenizer

To use our curated, morphological tokenizer built on MorphyNet (Batsuren et al., 2022), for pretraining, we first run it over the 10M corpus

and collect every distinct morpheme that is not an <unk> piece – 36,638 morphemes – and add four special tokens (<pad>, <unk>, <bos>, <eos>). Crucially, the tokenizer has *no* subword or BPE fallback: any piece it cannot resolve to a real morpheme is mapped to the single <unk> token rather than backed off to characters or byte pairs.[6] So that the model can recover where words begin, we mark word boundaries with a fused word-initial prefix on the first morpheme of each word (like BPE's ␣), e.g. `unhappiness→␣un happy ness`.

4.5 Baseline tokenizers

We compare against Unigram, Wordpiece, BPE, Morfessor 2.0, and MorphPiece. Unigram (Kudo, 2018), and WordPiece are each trained on the identical 10M corpus to a vocabulary of 32,000 – a standard size chosen to be close to our 36,638-morpheme inventory so that vocabulary size is not the axis of comparison. Morfessor 2.0, an unsupervised minimum description length (MDL) segmenter, is trained at its default settings and reported at whatever vocabulary it induces (68,874 here). Since MorphPiece is not open source, we reimplemented it, using the MorphyNet table with a BPE fallback and *no* curation (26,869 tokens). It shares our exact lexical source (MorphyNet), so the only difference between it and our tokenizer is our curation, stem normalization, and no-fallback design; it isolates the effect of the curation itself rather than of consulting a morphological resource at all.

4.6 Evaluation

We evaluate every trained model on the BabyLM 2026 zero-shot suite: BLiMP and the BLiMP Supplement (grammatical knowledge), EWoK (world knowledge), entity tracking, and COMPS (conceptual knowledge). All of these are scored by the standard minimal-pair method: for a pair of candidate continuations we compare the summed token log-probabilities the model assigns to each, count the pair as correct when the acceptable candidate is more probable, and macro-average accuracy per phenomenon. Because the tokenizers differ in fertility (tokens per word; Table 4), a fixed span of text becomes a different number of tokens under each, so we hold the context length constant in tokens rather than in words and report tokens-seen alongside the scores.

[6] However, for our BabyLM submission we do use character-level fallback in order to handle OOV items in the BabyLM evaluation data.

5 Results

5.1 Comparing the tokenizers on their own

We first compare the six tokenizers intrinsically on the same 10M corpus, with no model in the loop: **Ours**, BPE, Unigram, and WordPiece at a matched 32k vocabulary, Morfessor 2.0, and a reimplemented MorphPiece (raw MorphyNet plus a BPE fallback). Vocabulary size, fertility, and OOV are reported in Table 4 (Appendix A). Ours sits at a moderate fertility (1.18 tokens/word) with a 36,638-morpheme vocabulary comparable to the 32k baselines and holds OOV to 1% of tokens with no fallback, while Morfessor barely segments at its default (fertility 1.015) and raw MorphPiece covers only 19.3% of corpus tokens from its morpheme table.

Table 2 compares the tokenizers on a fixed set of words, chosen to expose where a purely statistical split diverges from a semantically-faithful one: derived words that should share a stem (*unhappiness*, *happiness*), words a statistical splitter mangles across a false boundary (*adder*, *singleton*), lexicalized wholes (*business*, *werewolf*), and transparent derivations (*nationalistic*, *disagreement*).

The contrast with the *uncurated* MorphPiece column shows that without the semantic constraint, the same MorphyNet base overdecomposes lexicalized wholes (*business*→`busy ness`, *werewolf*→`were wolf`) and produces incorrect splits (*running*→`run er ing`, *happiness*→`hap y ness`), while Ours keeps those items whole and splits only where meaning is preserved.

The curation, quantified. The gap between **Ours** and **MorphPiece (raw)** columns of Table 2 is measurable hand curation: 28,571 words carry an explicit decision (19,982 split, 7,344 kept whole, 1,245 normalized). Against the *raw* MorphyNet table MorphPiece consults, our decision differs for roughly ten thousand words — we reject 1,114 splits it would make (S1–S4), change 3,933 it mis-segments (`kingdom→kin ing dom` becomes `king dom`), and add 4,961 it misses.

How much sharing does this buy? In total, 15.8% of corpus tokens fall in words we decompose. Of the 36,638 morphemes Ours emits, 41.9% are reused across two or more word forms; the rest are whole words kept atomic for which

Word	Ours	BPE	Unigram	WordPiece	Morfessor	MorphPiece (raw)
business	business	business	business	business	business	busy ness
adder	adder	ad der	adder	add ##er	adder	add er
nutrition	nutrit ion	nutrition	nutrition	nutrition	nutrition	nutrition
nutritious	nutrit ous	nutritious	nutritio u s	nutri ##ti ##ous	nutritious	nutrition ous
unhappiness	un happy ness	unhappiness	unhappi ness	unhapp ##iness	unhappiness	un hap y ness
happiness	happy ness	happiness	happi ness	happiness	happiness	hap y ness
seeming	seeme ing	seeming	seeming	seeming	seeming	seeme ing
singleton	singleton	sing leton	single t on	single ##ton	singleton	single ton
crier	cry er	crier	crie r	crier	crier	cry er
suburbia	suburb ia	subur bia	suburb i a	suburb ##ia	suburbia	suburb ia
caliphate	caliph ate	caliph ate	caliph a t e	caliph ##ate	caliphate	caliph ate
running	run ing	running	running	running	running	run er ing
nationalistic	nation al ist ic	national istic	national istic	nationalist ##ic	national istic	nation al ist ic
werewolf	werewolf	werewolf	werewolf	werewolf	werewolf	were wolf
undoing	undo ing	undoing	undo ing	undo ##ing	undoing	un do ing
disagreement	dis agree ment	disagreement	disagreement	disagreement	disagreement	dis agree ment
prehistoric	pre history ic	prehistoric	pre historic	prehistoric	prehistoric	pre history ic
tokenization	tokenization	token ization	token ization	token ##ization	token ization	token ization

Table 2: Qualitative segmentation comparison. Each cell is the token sequence for the word. WordPiece's ## marks a word-internal continuation piece. The uncurated MorphPiece column (same MorphyNet base as Ours, no semantic constraint, and with BPE fallback) over-splits lexicalized wholes and yields noisy chains where Ours does not.

no productive split exists. Among the 41.9% of morphemes that are shared, the typical stem appears in about 3 word forms (e.g., help pools signal from helped, helping, helps), giving it roughly 36% more training exposure than its most frequent word form alone.

5.2 Comparing models trained on different tokenizers

The intrinsic comparison shows the tokenizations differ. But does this difference *matter* to a language model trained with it? We pretrain one language model per tokenizer, holding everything else identical (Section 4): the same 12-layer, 768-hidden architecture, optimizer and schedule, 10M corpus, and the same seed. The only thing that varies is the tokenizer — and, unavoidably, the size of the embedding table it induces. Table 3 reports downstream accuracy on BLiMP, the BLiMP supplement, and COMPS, alongside each condition's parameter count.[7]

Curation is what matters. The cleanest comparison in the table is Ours vs. the reimplemented MorphPiece: both draw their morphemes from the *same* MorphyNet source, and differ only in our curation and no-fallback lemma scheme. Ours reaches 66.35 BLiMP against MorphPiece's 64.96 — a $+1.4$**-point gain from curation**. It is not that a tokenizer consults morphology that helps, but *how faithfully* it does so.

[7]For this comparison, the Ours condition restricts MorphyNet to its 160,000 most frequent forms (by `wordfreq`), yielding 36,638 morphemes; the submitted model (Section 5.3) uses MorphyNet in full.

Tokenizer	Params	BLiMP	Supp.	COMPS
Ours (no fallback)	145.0M	66.35	57.60	**56.91**
Ours + char. backoff	145.2M	67.13	57.21	—
BPE	146.3M	**67.26**	57.46	56.21
Unigram	145.3M	65.28	**58.54**	56.43
WordPiece	141.2M	66.20	55.45	56.09
Morfessor 2.0	175.6M	67.18	57.35	56.74
MorphPiece (raw MorphyNet)	138.9M	64.96	58.11	**57.10**

Table 3: Downstream accuracy under an identical model and training recipe, varying only the tokenizer. *Params* is the total parameter count (non-embedding parameters are identical across conditions; the spread is entirely the embedding table). *Supp.* is the BLiMP Supplement. All conditions use a plain autoregressive objective to isolate the tokenizer; the submitted model (Section 5.3, 72.53 BLiMP) adds MNTP and vocabulary pruning.

On COMPS (conceptual knowledge) the three morphology-based tokenizers occupy the top three positions (MorphPiece 57.10, Ours 56.91, Morfessor 56.74), with the purely statistical tokenizers trailing.

On BLiMP, BPE outperforms Ours (67.26 vs. 66.35). Ours wins slightly more BLiMP phenomena than it loses (35 of 67), and wins them where structure spans the sentence — NPI licensing (+10 to +13), Principle-A binding (+14.8), islands (+11.3), ellipsis (+9.9), irregular subject–verb agreement (+12.2). But it loses a handful of phenomena by large margins: determiner–noun agreement (-27) and superlative quantifiers (-39). Those categories, where splitting a determiner or inflection fragments a local agreement cue, drag the average below BPE.

MNTP reverses the BPE advantage. Table 3 trains under a plain autoregressive objective (with the same Muon optimizer used throughout). When we swap in the masked-next-token-prediction (MNTP) objective (BehnamGhader et al., 2024; Charpentier and Samuel, 2024) of our BabyLM submission recipe — holding the optimizer, tokenizer, and everything else fixed — and re-run the Ours-vs-BPE comparison, the ranking reverses: Ours reaches 67.71 BLiMP against BPE's 67.11.

Finally, character backoff is worth $+0.8$ BLiMP over the no-fallback tokenizer (67.13 vs. 66.35) (Section 5.3); this ablation uses the 160k-restricted tokenizer, not the submission's full-MorphyNet analyzer.

Further results The full BabyLM submission recipe tokenizer comparison across all six tokenizers is in Appendix C (Table 6). A model-size scaling curve can be found in Appendix D (Table 7).

5.3 BabyLM Submission

For the BabyLM 2026 `strict-small` (10M) track we submit a single model that pairs our tokenizer with the strongest training recipe we found. The tokenizer is the curated morphological analyzer of Section 4 with two additions: (i) a *pruned* vocabulary in which morphemes attested fewer than three times in the training corpus are dropped and routed through character backoff rather than kept as under-trained embedding rows, giving a 28,366-token vocabulary; and (ii) *character backoff* for out-of-vocabulary pieces (an unknown surface is emitted as its characters, not a single `<unk>`), so the model is never blind to a rare word — the ablation in Section 5.2 shows this backoff is worth $+0.8$ BLiMP over the no-fallback tokenizer. The model is a 12-layer, 768-hidden `LlamaForCausalLM` ($\sim$135M parameters) trained for 10 epochs with the Muon optimizer and a masked-next-token-prediction (MNTP; BehnamGhader et al., 2024) objective applied to 20% of microbatches (GPT-BERT-style hybrid; Charpentier and Samuel, 2024), with the per-token mask rate inside those microbatches decaying linearly from 0.30 to 0.15 over training and 80/10/10 corruption of the masked positions.

On the BabyLM 2026 `strict-small` leaderboard this model achieves **72.53** on BLiMP, 58.3 on the BLiMP supplement, and ranks 2nd on COMPS (54.67), 3rd on EWoK (53.2), and 7th on GlobalPIQA (38.65).

It is released as a self-contained HuggingFace repository: the analyzer, its resource tables, and the vocabulary ship with the model, and the tokenizer requires no external dependencies at load time. All tokenizer artifacts, the pretraining configuration, and the build scripts are publicly available. The character-backoff and vocabulary-pruning choices are the only submission-specific deviations from the controlled comparison of Section 5.2, which deliberately uses the plain, unpruned tokenizer with no fallback to isolate the effect of morphological segmentation itself.

Finally, training an autoregressive model with the submission tokenizer but without MNTP gives 71.29 BLiMP: **most of the $\sim$6.2-point jump from Table 3's 66.35 comes from the tokenizer configuration itself** ($\sim$4.9 points), and MNTP adds only $\sim$1.2 on top.

6 Conclusion

In the low-resource regime, where every word form is thinly attested, a tokenizer that lets related forms share a morpheme is spending the data budget more efficiently, but only if the sharing is semantically related. We articulate this criterion precisely and provide a hand-curated English tokenizer that implements it. Intrinsically our tokenizer segments words into morphologically meaningful tokens, whereas statistical tokenizers cross morpheme boundaries. A comparison with an uncurated MorphPiece baseline shows that the same morphological resource, *without* the semantic constraint, over-splits lexicalized wholes, and this has downstream effects when pretraining. Our results suggest that morphological tokenization helps most when segmentation decisions are constrained by meaning, not just by boundary alignment.

Limitations

English is morphologically poor. English inflection is sparse and its derivation is heavily lexicalized, so the amount of sharing a morphological tokenizer can recover is modest. Our gains should be read as a *conservative lower bound* on what the same approach would yield for a morphologically rich language, where a single word can pack many productive morphemes and statistical tokenizers fare worst. Across Turkish (Toraman et al., 2023), Finnish and Russian (Hou et al., 2023), Hebrew (Gueta et al., 2023), Latin

(Hudspeth et al., 2026), Slovak (Drzík and Kapusta, 2026), Kinyarwanda (Nzeyimana and Niyongabo Rubungo, 2022), and polysynthetic languages (Schwartz et al., 2020), morphology-aware tokenization has shown its clearest gains — and MorphyNet, our lexical backbone, already spans 15 languages, so the curated-lemma method could be extended to these languages in principle. Testing the criterion in those settings is a natural next step.

The criterion is hand-applied. The semantic half (condition S) is a human judgment; we make it explicit and identify automatable signals, but the tokenizer as delivered reflects manual curation and is not an automated pipeline. This bounds how quickly it transfers to new languages.

Maintenance. Because the tokenizer's lookup tables are static, newly coined words, emerging senses, and shifting compositional intuitions (e.g. *ghosting* acquiring a social sense unrelated to *ghost*) would require periodic re-curation.

The gains do not extend to GLUE. They concentrate in the minimal pair evaluations and do not carry to finetuned (Super)GLUE, where our models plateau below the best statistical-tokenizer systems, and neither adaptive masking nor task-targeted synthetic data closed the gap (Appendix F).

References

Sawsan Alqahtani, Mir Tafseer Nayeem, Md Tahmid Rahman Laskar, Tasnim Mohiuddin, and M Saiful Bari. 2026. Stop taking tokenizers for granted: They are core design decisions in large language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8410–8432, Rabat, Morocco. Association for Computational Linguistics.

Gül Sena Altintas, Malikeh Ehghaghi, Brian Lester, Fengyuan Liu, Wanru Zhao, Marco Ciccone, and Colin Raffel. 2025. Toksuite: Measuring the impact of tokenizer choice on language model behavior. *CoRR*, abs/2512.20757.

Catherine Arnett and Benjamin Bergen. 2025. Why do language models perform worse for morphologically complex languages? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6607–6623, Abu Dhabi, UAE. Association for Computational Linguistics.

Catherine Arnett, Marisa Hudspeth, and Brendan T. O'Connor. 2025. Evaluating morphological alignment of tokenizers in 70 languages. *CoRR*, abs/2507.06378.

Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. 2025. Morphbpe: A morpho-aware tokenizer bridging linguistic complexity for efficient LLM training across morphologies. *CoRR*, abs/2502.00894.

Khuyagbaatar Batsuren, Gábor Bella, Aryaman Arora, Viktor Martinovic, Kyle Gorman, Zdeněk Žabokrtský, Amarsanaa Ganbold, Šárka Dohnalová, Magda Ševčíková, Kateřina Pelegrinová, Fausto Giunchiglia, Ryan Cotterell, and Ekaterina Vylomova. 2022. The SIGMORPHON 2022 shared task on morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 103–116, Seattle, Washington. Association for Computational Linguistics.

Khuyagbaatar Batsuren, Gábor Bella, and Fausto Giunchiglia. 2021. MorphyNet: a large multilingual database of derivational and inflectional morphology. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 39–48, Online. Association for Computational Linguistics.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *CoRR*, abs/2404.05961.

Jean Berko. 1958. The child's learning of english morphology. *Word*, 14(2-3):150–177.

Necva Bölücü and Burcu Can. 2025. A morpheme-aware child-inspired language model. In *Proceedings of the First BabyLM Workshop*, pages 279–287, Suzhou, China. Association for Computational Linguistics.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Jan A. Botha and Phil Blunsom. 2014. Compositional morphology for word representations and language modelling. In *International Conference on Machine Learning*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

David Drzík and Jozef Kapusta. 2026. The importance of morphology-aware subword tokenization for NLP tasks in slovak language modeling. *Expert Syst. Appl.*, 312:131492.

Lukas Edman and Alexander Fraser. 2025. Mask and you shall receive: Optimizing masked language modeling for pretraining BabyLMs. In *Proceedings of the First BabyLM Workshop*.

Lennart Finke, Chandan Sreedhara, Thomas Dooms, Mat Allen, Emerald Zhang, Juan Diego Rodriguez, Noa Nabeshima, Thomas Marshall, and Dan Braun. 2025. Parameterized synthetic text generation with simplestories. *CoRR*, abs/2504.09184.

Achille Fusco, Maria Letizia Piccini Bianchessi, Tommaso Sgrizzi, Asya Zanollo, and Cristiano Chesi. 2025. Linguistic units as tokens: Intrinsic and extrinsic evaluation with BabyLM. In *Proceedings of the First BabyLM Workshop*, pages 496–507, Suzhou, China. Association for Computational Linguistics.

Zébulon Goriely, Richard Diehl Martinez, Andrew Caines, Paula Buttery, and Lisa Beinborn. 2024. From babble to words: Pre-training language models on continuous streams of phonemes. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 37–53, Miami, FL, USA. Association for Computational Linguistics.

Eylon Gueta, Omer Goldman, and Reut Tsarfaty. 2023. Explicit morphological knowledge improves pretraining of language models for hebrew. *CoRR*, abs/2311.00658.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Valentin Hofmann, Hinrich Schuetze, and Janet Pierrehumbert. 2022. An embarrassingly simple method to mitigate undesirable properties of pretrained language model tokenizers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–393, Dublin, Ireland. Association for Computational Linguistics.

Jue Hou, Anisia Katinskaia, Anh-Duc Vu, and Roman Yangarber. 2023. Effects of sub-word segmentation on performance of transformer language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7413–7425, Singapore. Association for Computational Linguistics.

Marisa Hudspeth, Patrick J. Burns, and Brendan O'Connor. 2026. Contextual morphologically-guided tokenization for Latin encoder models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5761–5775, Rabat, Morocco. Association for Computational Linguistics.

Haris Jabbar. 2023. Morphpiece : Moving away from statistical language representation. *CoRR*, abs/2307.07262.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. Muon: An optimizer for hidden layers in neural networks.

Jaap Jumelet, Michael Hanna, Marianne de Heer Kloots, Anna Langedijk, Charlotte Pouw, and Oskar van der Wal. 2023. ChapGTP, ILLC's attempt at raising a BabyLM: Improving data efficiency by automatic task formation. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 74–85, Singapore. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Zilong Li. 2024. Team ryu's submission to SIGMORPHON 2024 shared task on subword tokenization. *CoRR*, abs/2410.17094.

Tomasz Limisiewicz, Terra Blevins, Hila Gonen, Orevaoghene Ahia, and Luke Zettlemoyer. 2024. MYTE: Morphology-driven byte encoding for better and fairer multilingual language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15059–15076, Bangkok, Thailand. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, Shijin Wang, and Guoping Hu. 2020. CharBERT: Character-aware pre-trained language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 39–50, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Michael Mayor, editor. 2009. *Longman Dictionary of Contemporary English*, 5th edition. Pearson Longman, Harlow, England.

Sherin Muckatira, Vijeta Deshpande, Vladislav Lialin, and Anna Rumshisky. 2024. Emergent abilities in

reduced-scale generative language models. In *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, volume NAACL 2024 of *Findings of ACL*, pages 1242–1257. Association for Computational Linguistics.

Antoine Nzeyimana and Andre Niyongabo Rubungo. 2022. KinyaBERT: a morphology-aware Kinyarwanda language model. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5347–5363, Dublin, Ireland. Association for Computational Linguistics.

Iñigo Parra. 2024. Morphological typology in BPE subword productivity and language modeling. *CoRR*, abs/2410.23656.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Jonne Saleva and Constantine Lignos. 2021. The effectiveness of morphology-aware segmentation in low-resource neural machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 164–174, Online. Association for Computational Linguistics.

Claudia H Sánchez-Gutiérrez, Hugo Mailhot, S Hélène Deacon, and Maximiliano A Wilson. 2018. Morpholex: A derivational morphological database for 70,000 english words. *Behavior research methods*, 50(4):1568–1580.

Lane Schwartz, Francis M. Tyers, Lori S. Levin, Christo Kirov, Patrick Littell, Chi-kiu Lo, Emily Prud'hommeaux, Hyunji Hayley Park, Kenneth Steimel, Rebecca Knowles, Jeffrey Micher, Lonny Strunk, Han Liu, Coleman Haley, Katherine J. Zhang, Robbie Jimerson, Vasilisa Andriyanets, Aldrian Obaja Muis, Naoki Otani, and 2 others. 2020. Neural polysynthetic language modelling. *CoRR*, abs/2005.05477.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24, Gothenburg, Sweden. Association for Computational Linguistics.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864.

Cagri Toraman, Eyup Halit Yilmaz, Furkan Sahinuç, and Oguzhan Ozcelik. 2023. Impact of tokenization on language models: An analysis for turkish. *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, 22(4):116:1–116:21.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Saketh Reddy Vemula, Sandipan Dandapat, Dipti Sharma, and Parameswari Krishnamurthy. 2025. Rethinking tokenization for rich morphology: The dominance of unigram over BPE and morphological alignment. In *The 14th International Joint Conference on Natural Language Processing and The 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 232–252, Mumbai, India. Association for Computational Linguistics.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, and 12 others. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

A Intrinsic tokenizer comparison

Table 4 reports the intrinsic properties of the six tokenizers on the BabyLM 10M corpus (vocabulary size, fertility, and out-of-vocabulary rate), referenced from Section 5.1.

Tokenizer	Vocab	Fallback	Fertility	OOV (tok)	OOV (word)	tok/type
Ours (curated morph)	36,638	no	1.177	0.99%	1.16%	1.57
BPE	32,000	yes	1.107	0.00%	0.00%	2.37
Unigram	32,000	yes	1.329	0.00%	0.00%	2.90
WordPiece	32,000	yes	1.112	0.00%	0.00%	2.46
Morfessor 2.0	68,874	yes	1.015	0.00%	0.00%	1.64
MorphPiece (raw MorphyNet)	26,869	yes	1.287	0.00%	0.00%	2.53

Table 4: Intrinsic comparison on the BabyLM 10M `strict-small` corpus. *Fertility* is emitted tokens/word (lower = more compression). BPE/Unigram/WordPiece/Morfessor/MorphPiece carry a subword or BPE *fallback*, so they are 0% OOV *by construction*; **Ours has no fallback**, so its 1% `<unk>` tail is the price of never emitting a non-morpheme.

B Full ablation results

Beyond the controlled tokenizer comparison (Section 5.2), we ran an extensive search over training recipe and data-side interventions to build the submitted system. Table 5 collects every 10M model we evaluated. All scores are from our own reproduction of the BabyLM zero-shot scorer (BLiMP main + supplement, COMPS, EWoK, entity-tracking) and the finetuned (Super)GLUE macro-average. Dashes mark evaluations we did not run for that condition — we scored BLiMP for every model, the full suite for the core family, and (Super)GLUE for the leaderboard-relevant set.

The submitted system is **m20-ms**. "m20" denotes that base recipe (pruned morphological tokenizer + Muon + masked-next-token prediction at a 0.20 masking fraction, 135M parameters, trained on the raw 10M corpus with no data augmentation). Every "m20-∗" row adds one or more data-side ingredients on top of that identical base.

C Submitted recipe tokenizer comparison

Table 3 uses a plain autoregressive objective to isolate the effects of the tokenizer. Table 6 repeats the comparison under the full submitted recipe (Muon + MNTP at 20% masking), which is the recipe behind the submitted model. **The pruned morphological tokenizer (the submission configuration) leads BLiMP by 4.4–6.5 points over every un-pruned tokenizer.** Among un-pruned tokenizers, Unigram slightly outperforms Ours-full and BPE on BLiMP, while Morfessor leads on the BLiMP supplement. Three of the five un-pruned conditions (BPE, Unigram, MorphPiece) outperform the pruned submission on GLUE.

D Effect of model size

Table 7 reports the submitted recipe (pruned morphological tokenizer) at six model sizes from 8M to 135M parameters. BLiMP improves smoothly (+2.9 from 8M to 135M), with the largest step at 14M→28M. GLUE does not improve with size and is noisy (61–65).

E Extended related work

Statistical subword tokenizers. The dominant tokenizers – BPE (Sennrich et al., 2016), WordPiece (Wu et al., 2016), and Unigram LM (Kudo, 2018) – share an objective (compress the corpus under a vocabulary budget) that is agnostic to morphology. Bostrom and Durrett (2020) show BPE is suboptimal: at a fixed vocabulary, Unigram LM matches or outperforms it downstream, while inducing segments that align more closely with morphology. Vemula et al. (2025) also find that, in their comparison of BPE and Unigram across Telugu, Hindi, and English on classification and structure prediction tasks, Unigram consistently outperforms BPE.

However, while Unigram improves statistical boundary alignment, it still fragments words across meaning, rendering e.g. `singleton` as `single t on` and `caliphate` as `caliph a t e` in our experiments, and it does not unify `-ise`/`-ize` or an inflectional paradigm onto one unit. Our tokenizer abandons the compression objective entirely for an explicit, semantically-constrained morphological one.

Unsupervised and resource-based morphological segmentation. Morfessor (Smit et al., 2014) induces a morphological lexicon from raw text under a minimum description length (MDL) prior. It is unsupervised and language-agnostic but optimizes for description length, not meaning, and

Model	= base submitted (m20) +	BLiMP	Supp.	COMPS	EWoK	Entity	GLUE
m20-ms (submitted)	— (submitted run)	**72.53**	58.44	54.66	52.98	20.31	63.9
m20-all	filter + synthetic + swap + curriculum	**72.60**	—	—	—	—	—
m20-super	all augmentations	72.51	58.02	56.26	52.86	—	—
m20-s3	seed 3	72.47	55.48	54.70	53.60	19.69	64.1
grand-1	filter + synthetic + swap	72.46	—	—	—	—	—
m20	— (base submitted)	72.36	58.45	54.59	52.93	20.27	63.9
m20-ns	name-swap	72.36	56.28	53.93	54.15	21.16	63.8
m20-amlm	Adaptive MLM	72.24	—	—	—	—	63.7
m10	mask-frac 0.10 (90/10)	72.18	—	—	—	—	—
m30	mask-frac 0.30 (70/30)	71.76	56.64	54.35	53.65	20.90	64.1
m15	mask-frac 0.15 (85/15)	71.57	55.73	54.15	53.42	20.11	**64.9**
m20-s2	seed 2	71.33	58.64	53.73	53.17	20.15	—
m20-58m	58M parameters	71.28	55.94	54.35	52.95	21.06	64.6
m25	mask-frac 0.25 (75/25)	71.15	56.19	53.88	52.85	21.41	64.8
super-58m-c512	all augment., 58M + ctx 512	70.93	57.51	56.14	53.11	22.04	—
m20-c512	context length 512	70.64	—	—	—	—	—
m20-v8nc	name-collapse	70.25	55.06	53.68	52.52	19.89	63.7
m20-cur	curriculum (alone)	69.68	56.73	53.91	52.22	20.59	63.6

Table 5: Full ablation over recipe and data-side interventions on BabyLM 10M `strict-small`. **m20** is the base recipe, with our pruned morphological tokenizer, Muon, MNTP on 20% of microbatches, 135M parameters, context length of 256, and the cleaned 10M corpus. Every other row changes one or more ingredients relative to it. ***Data-side:*** *filter* drops documents with >5% out-of-vocabulary tokens; *synthetic* adds Qwen-generated targeted text (Appendix F); *swap* remaps names onto evaluation-frequent names; *collapse* merges consecutive name tokens into one; *curriculum* orders the first epoch by OOV rate; *all augmentations* is all five together. ***Modeling-side:*** *mask-frac x (a/b)* trains MNTP on a fraction x of microbatches and causal LM on the rest; *Adaptive MLM* is from Edman and Fraser (2025); *seed 2/3* re-run the base recipe with a different random seed; *58M* and *ctx 512* change model size and context length (base: 135M, 256). **m20-ms** is a second training run of the base recipe (m20) and is the submitted model.

Tokenizer	BLiMP	Supp.	EWoK	COMPS	GLUE
Ours pruned 28k (submitted)	**72.53**	58.38	53.08	54.61	63.87
Ours un-pruned 41k	67.71	58.36	**53.82**	**55.41**	63.56
BPE	67.11	56.81	52.72	53.99	64.02
Unigram	67.97	56.45	50.97	54.17	**64.95**
MorphPiece (raw MorphyNet)	65.89	58.59	53.52	54.61	**64.95**
Morfessor 2.0	66.63	**60.38**	50.80	54.45	63.49

Table 6: Tokenizer comparison under the submitted recipe (Muon + MNTP@20%), the same recipe as the submitted model. Only the tokenizer varies.

Params	BLiMP	Supp.	EWoK	COMPS	GLUE
8M	69.43	56.42	52.16	52.79	61.43
14M	69.47	57.29	52.35	53.60	**65.00**
28M	71.28	56.59	53.41	53.94	63.92
58M	71.32	56.07	52.85	54.33	63.52
100M	71.81	**58.82**	53.08	54.11	64.22
135M	**72.53**	58.38	53.08	**54.61**	63.63

Table 7: Training the submitted recipe with various model sizes, with the pruned morphological tokenizer. All other hyperparameters held fixed. The 135M parameter model is the one we submitted to the competition.

(as we find in Section 5.1) barely segments at its default setting. Supervised morpheme segmentation, e.g. the SIGMORPHON 2022 shared task (Batsuren et al., 2022), and large morphological databases such as MorphyNet (Batsuren et al., 2021) provide gold or silver segmentations for many languages. We build on MorphyNet but do not take its segmentations as ground truth: it encodes historical and derivational structure that our criterion often rejects (Section 3), which is why the tokenizer required substantial hand curation.

Morphology-aware tokenizers for LMs. MorphPiece (Jabbar, 2023) combines a rule-based morphological table with a BPE fallback and reports gains at GPT-2 scale; it is the closest engineering precedent to our tokenizer, and we reimplement it as a baseline. FLOTA (Hofmann et al., 2022) keeps a standard vocabulary but selects the longest available segments from it, which tend to

better preserve morphological structure.

A complementary line constrains statistical training rather than replacing it: MorphBPE (Asgari et al., 2025) forbids BPE merges that cross morpheme boundaries. Byte-level schemes such as MYTE (Limisiewicz et al., 2024) apply morphology to balance sequence lengths across languages rather than to model monolingual morphology. Closest to us in setting, two BabyLM submissions train child-scale models on morphologically segmented vocabularies: Bölücü and Can (2025), using unsupervised Morfessor and a rule-based affix-stripper, and Fusco et al. (2025), who compare BPE against paradigm- and syllable-based tokenizers and conclude that no single tokenizer dominates, with morphology aiding semantics but not syntax. Unlike this work, both segment automatically and at the surface, without any semantic-relatedness constraint.

Evaluating tokenizers on morphology. A parallel line asks how well existing tokenizers respect morphology and whether it predicts model quality. Hofmann et al. (2021) show that derivationally segmented input improves generalization on rare complex words; MorphScore (Arnett and Bergen, 2025) and its 70-language extension (Arnett et al., 2025) quantify morphological alignment across tokenizers and languages; Parra (2024) relates BPE subword productivity to morphological typology across six languages. Notably, Arnett et al. (2025) find that morphological alignment explains little of the variance in downstream performance. One possible reason is that a morphologically aligned split can still be semantically misleading. Whether semantically constrained splits predict downstream gains across languages remains to be tested.

Low-resource and child-scale modeling. Several works find tokenization and vocabulary especially consequential in low-data settings: (Muckatira et al., 2024; Finke et al., 2025; Goriely et al., 2024; Li, 2024; Jumelet et al., 2023). For morphologically rich languages the stakes are higher still: morphology-aware tokenization has been reported to help across Finnish and Russian (Hou et al., 2023), Hebrew (Gueta et al., 2023), Latin (Hudspeth et al., 2026), Slovak (Drzík and Kapusta, 2026), Kinyarwanda (Nzeyimana and Niyongabo Rubungo, 2022), and polysynthetic languages (Schwartz et al., 2020), though some of the evidence is mixed (Saleva and Lignos, 2021; Toraman et al., 2023; Vemula et al., 2025), and several reported gains are in perplexity or convergence rather than downstream accuracy.

F Other intervenions

Most interventions that are intuitive or that helped in other settings did not improve our system; one did. We report them because negative results are useful to the community.

Name remapping is neutral. We remapped names in the training data onto the names that recur in the evaluation suites, consistently within each training sequence, and only within gender and person/place pools,[8] expecting better performance since the model is not wasting capacity learning a plethora of unnecessary names. It was neutral on the headline metrics: identical BLiMP (72.36), unchanged (Super)GLUE (63.8 vs. 63.9), better on EWoK/entity ($+1.2$/$+0.9$) but worse on the BLiMP supplement (-2.2).

Curriculum does not help. We tried a curriculum which orders the first epoch's documents by OOV tier (fraction of <UNK> tokens under the pruned vocabulary, bucketed into four tiers from 0% to 5%, easy-to-hard); it also drops all documents with >5% OOV, removing 3.5% of documents (7.3% of tokens) relative to the full-corpus baseline.

The combined effect costs -2.7 BLiMP (72.36→69.68), though this conflates reordering with the data reduction. Comparing two runs that both drop the same documents and differ only in ordering, reordering alone is approximately neutral (grand-1 72.46 → m20-all 72.60, $+0.14$). Curriculum's harm is offset once synthetic data is added (m20-all recovers to 72.60), but it never helps, consistent with prior BabyLM findings that curriculum learning rarely pays off (Warstadt et al., 2023).

Longer context hurts the strong recipe. Raising the context length from 256 to 512 cost

[8] Names were bucketed into four pools (male, female, gender-neutral person, place) and a name was only ever replaced by a name from the same pool, so anaphor agreement (*John…himself*) and person/place distinctions were never corrupted. The mapping was drawn fresh for each training sequence and held fixed within it, so repeated mentions of a name in one sequence stayed consistent while the same corpus name landed on different targets elsewhere. Ambiguous word-names such as *bill* or *frank* could be targets but were never replaced, so their common-noun uses were left intact.

−1.7 BLiMP (72.36→70.64) under the submitted recipe.

Multi-stream input fusion improves perplexity but not grammar. Fusing character-, morpheme-, and word-level embeddings at the input (Botha and Blunsom, 2014; Ma et al., 2020) produced the lowest validation losses of any run (down to 3.29) yet BLiMP at or below baseline (67.8–68.8). This is consistent with Edman and Fraser (2025), who found that adding sub-token embeddings lowered BLiMP accuracy by 2 to 3 points while leaving (Super)GLUE fine-tuning performance unchanged.

Synthetic data is the one data-side win. The one data-side intervention that helped was adding targeted synthetic text. We used Qwen2.5-7B-Instruct (a ≤9B teacher, which the challenge FAQ permits) to generate short, simple, child-level sentences aimed at the base model's measured weaknesses. A first tranche (∼354k words) targeted the BLiMP phenomena on which the base model erred most, with each phenomenon's share proportional to its error rate (NPI licensing 52%, filler–gap 13%, determiner–noun agreement 11%, quantifiers 10%, reflexives 7%, irregular participles 7%); a second tranche (∼458k words) exercised property inheritance (for COMPS), object-location tracking (for entity tracking), and everyday physical and social knowledge (for EWoK). The synthetic text was interleaved into the filtered corpus. Every run that included it matched or exceeded the base model on BLiMP (grand-1 72.46, m20-super 72.51, m20-all 72.60 vs. 72.36), whereas the surface-normalization interventions (name remapping, name collapse) did not. The submitted model does not use synthetic data (Appendix G).

G Training data and budget compliance

Training corpus. We train exclusively on a *cleaned* version of the official BabyLM 2026 STRICT-SMALL corpus; no other text corpus is used. Cleaning is deterministic and reproducible from the released cleaning script: the raw corpus (10,000,019 words by `wc -w`) reduces to **9,732,593** words (−2.7%). The removed material is transcription and formatting junk rather than child-directed speech: 40,073 CHILDES transcriber-comment lines (the dominant removal), OpenSubtitles mojibake/OCR artifacts, subtitle markup and watermark lines (∼2,500 lines), and Simple English Wikipedia HTML entities. Post-cleaning line counts per source: BNC spoken 65,220; CHILDES 491,232; Gutenberg 59,934; OpenSubtitles 385,975; Simple English Wikipedia 58,709; Switchboard 2,892.

Word-budget accounting. Training text is 9,732,593 words (<10M); ten epochs give ∼97.3M cumulative words seen (under the 100M exposure cap). The submitted model uses no synthetic data, no external model, and no distillation. The 19 required intermediate checkpoints (every 1M words seen to 10M, every 10M to 100M) are published as revisions of the model repository.

Tokenizer resources. The morphological analyzer is a lookup/rule system; no component of it is trained on external text. It consults (i) hand-curated morpheme tables created by the authors (28,544 entries) and (ii) the MorphyNet English inflectional/derivational lexicon, used in full (the analyzer tables contain 187,132 form→segmentation entries and a 617,042-item known-word list of single-word spellings — every lemma/form spelling in MorphyNet, consulted for known-word decisions during segmentation). All corpus statistics used in vocabulary construction (frequency pruning, character backoff) derive from the training corpus itself. During training the analyzer can only consult entries for words occurring in the corpus (135,085 types); this effective slice is 59,285 known-word entries, 16,968 inflection entries, and 21,535 curated entries — at most 168,757 words counted maximally strictly, so training text plus consulted lexicon annotations total 9,901,350 words, under the 10M budget even on that conservative accounting.

H MorphoLex comparison

As an independent quality check, we compared our tokenizer's segmentations against MorphoLex (Sánchez-Gutiérrez et al., 2018), an expert-built derivational morphological database of ∼70,000 English words from the English Lexicon Project.[9] MorphoLex records only derivation on lemmas (no inflection), so we strip the inflectional ending from our output before comparing derivational structure. Table 8 shows the full breakdown.

[9] https://github.com/hugomailhot/MorphoLex-en

Bucket	Words	%	Definition
Identical	17,497	25.5	Both produce the same pieces
Agree on derivation	16,743	24.4	Same derivational split; we additionally peel off the inflectional suffix that MorphoLex does not represent
They split more	25,937	37.8	MorphoLex decomposes into bound Latin/Greek roots (*abduct*→`ab duct`) we deliberately keep whole
We split more	686	1.0	We over-segment relative to MorphoLex (includes some by-design cases such as suppletives)
Convention diff	2,538	3.7	Same piece count; one piece has a different canonical spelling (e.g., MorphoLex `accuse ion` vs. ours `accuse ation`; `im legal` vs. `in legal`)
We OOV	5,216	7.6	Our tokenizer cannot analyze the word; MorphoLex can (mostly proper names and rare tokens outside our lookup tables)
Total	**68,617**	**100**	

Table 8: Comparison of our segmentations against MorphoLex on all 68,617 words in MorphoLex (after deduplication and dropping 6 data-error rows). The 68,617 is the full MorphoLex database, not filtered to any corpus.

The comparison shows that where the two resources disagree, it is overwhelmingly by design: the 37.8% "they split more" bucket is almost entirely MorphoLex's etymological decomposition into bound roots (*abase*→`a base`, roots like *brevi*, *duct*, *plic*) that our synchronic criterion rejects for the same reason we do not split *remember*→`re member`. The comparison also confirmed that our productive-affix coverage matches MorphoLex, with zero missed *-ness/-ment/-ful/-less/-able* splits.

I Per-phenomenon acquisition curves

Figure 2 plots the per-phenomenon BLiMP accuracy of the submitted model (`morpheus-10M-v5`) across 19 official intermediate checkpoints (1M to 100M cumulative words seen, evaluated with the official `fast_eval` protocol). The phenomena separate into distinct acquisition classes:

- **Early risers (mastered by ∼3–5M words):** plain determiner–noun agreement (95+ by 3M words); the irregular and adjective-intervening variants reach 80–90 by 5M and continue to climb.
- **Slow climbers:** anaphor number agreement reaches 95+; anaphor gender agreement (∼70 at 5M → 96) and subject–verb agreement (65–76 at 5M → 82–88), together with passives, show steady growth through 100M words.
- **Late bloomers (emerge only after ∼30–50M words, i.e. epochs 4+):** *wh vs. that with gap* (20→55), *left branch island simple question* (20→65), *coordinate structure constraint complex left branch* (25→52)— the long-distance dependency family, which requires many exposures.
- **Never learned (flat or at chance through 100M words):** matrix-question NPI licensing (∼10), NPI-present (∼20–40), `only`-NPI licensing (wild oscillation), complex-NP island (∼40), tough-vs-raising-1 (∼30), Principle-A reconstruction (∼35). These are predominantly NPI licensing and hard island phenomena requiring abstract scope/licensing relations that 10M words and 135M parameters never acquire.

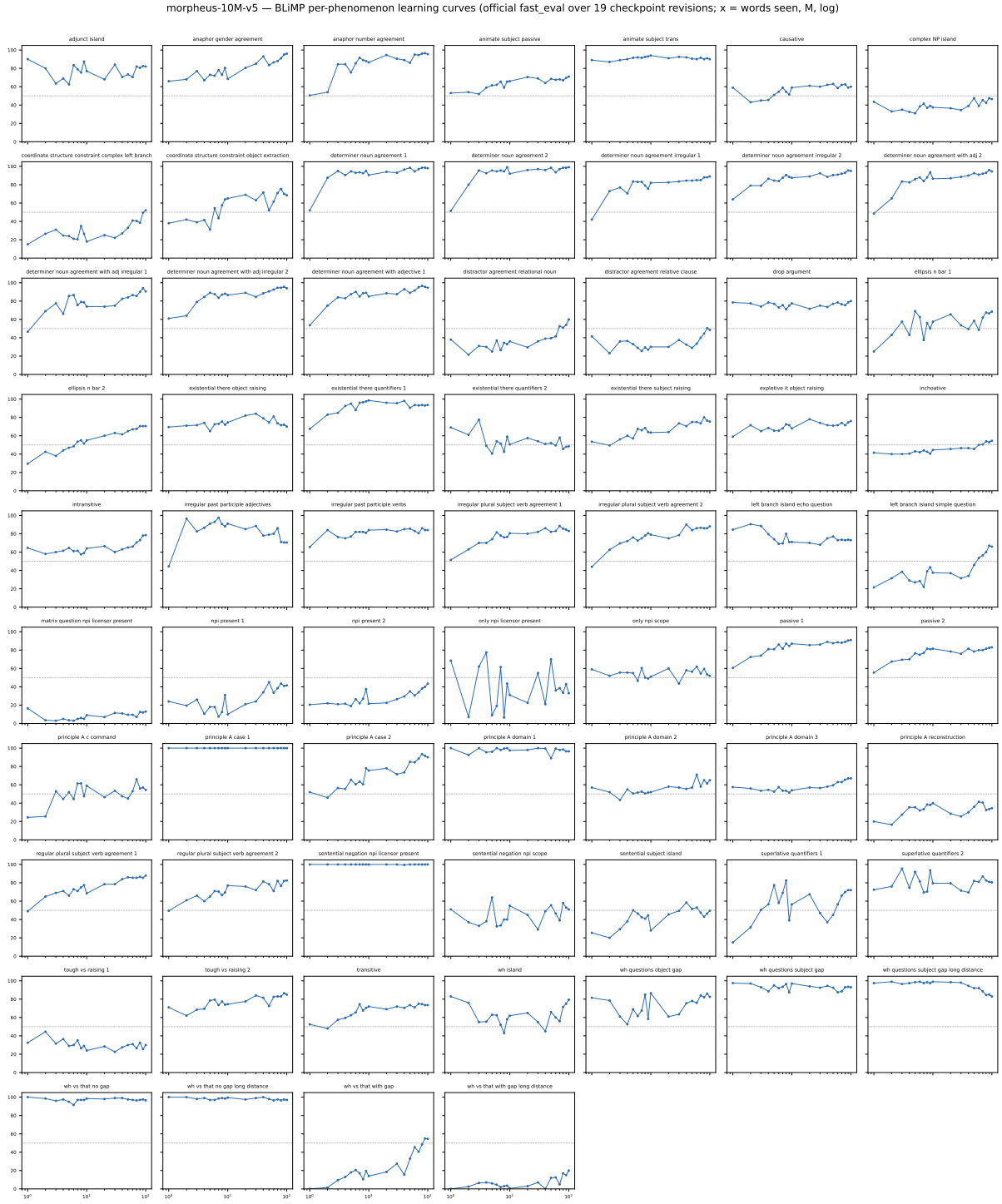


Figure 2: Per-phenomenon BLiMP learning curves for the submitted model across 19 official checkpoints (x-axis: cumulative words seen, log scale). Each panel is one BLiMP phenomenon. Phenomena separate into early risers (determiner–noun agreement, mastered by 3–5M words), slow climbers, late bloomers (long-distance dependencies, emerging after 30–50M words), and never-learned (NPI licensing, flat at chance).

What Does Input-Dependent Depth Routing Buy in Data-Limited Language Modeling?

Yupei Wang
University of California San Diego
yuw306@ucsd.edu

Abstract

Learned weighting of earlier layers, as used in ELC-BERT (Charpentier and Samuel, 2023), is a promising direction for sample-efficient language modeling. Attention Residuals (AttnRes; Kimi Team et al., 2026) extend this idea by computing input-dependent weights separately for each token, with reported gains in large-scale autoregressive models. We present a controlled study of whether such input-dependent depth routing helps in the data-limited regime where static routing first proved useful. Across the 10M- and 100M-word BabyLM budgets and three paired seeds, AttnRes consistently lowers test negative log-likelihood (NLL) compared to a residual baseline. It reaches the same development-set (dev) NLL after seeing fewer training words. In the 10M L32 control, AttnRes also lowers test NLL relative to static routing in all three seeds. Behavioral differences between AttnRes and the baseline are small and mixed. The BLiMP (Warstadt et al., 2020) advantage at equal training exposure disappears after matching on dev NLL. The trained 10M-word AttnRes models make causal use of routes from nonrecent depth blocks, but we find no clear evidence that categories with greater dependence on these sources achieve larger BLiMP gains over the baseline.

🤗 **AttnRes for BabyLM 2026** on Hugging Face (models, tokenizers, and checkpoints)
Training and analysis code on GitHub

1 Introduction

Residual connections in pre-norm Transformers accumulate earlier layer outputs with fixed unit weights. A growing line of work replaces this fixed accumulation with learned selection over depth. In the BabyLM setting, ELC-BERT (Charpentier and Samuel, 2023) won the strict and strict-small tracks of the BabyLM 2023 Challenge (Warstadt et al., 2023). It gives every layer a learned convex combination of all earlier layer outputs, with one static weight per pair of layers. Attention Residuals (AttnRes; Kimi Team et al., 2026) make this selection *input-dependent*: a depth-wise softmax computes routing weights over preceding representations separately for each token. At large scale, Block AttnRes, the variant that routes among groups of consecutive sublayers called *blocks*, matches the loss of a baseline trained with 1.25× more compute. The same work applies AttnRes to a Kimi Linear model with 48B total and 3B activated parameters trained on 1.4T tokens, improving downstream performance across many tasks.

Static routing already helped under BabyLM limits, and input-dependent routing helped at large scale. These results leave open whether input-dependent depth routing adds anything in a much smaller, data-limited setting, and what it adds. The BabyLM 2026 strict-small and strict tracks limit training data to 10M and 100M words, respectively (Choshen et al., 2026). We trained models with 62–116M parameters, far below the scale of the reported Kimi experiment. In this setting, it remains unclear whether input-dependent routing improves linguistic performance at the same development-set (dev) negative log-likelihood (NLL). It is also unclear whether any linguistic gains are associated with reliance on nonrecent depth routes.

We therefore ask three questions. **RQ1:** Does AttnRes improve language-model sample efficiency under the BabyLM data limits? **RQ2:** Does the lower NLL associated with AttnRes translate into better linguistic behavior beyond overall training progress? **RQ3:** Do the trained models rely on nonrecent depth routes, and is stronger dependence associated with larger BLiMP gains over the baseline?

Proceedings of the Second BabyLM Workshop, pages 892–903
October 29, 2026 ©2026 Association for Computational Linguistics

We compare residual baselines and Block AttnRes models under matched architectures, optimization, and training schedules, using three paired seeds. Our endpoint comparisons on the 10M track use 16- and 32-layer models. We also evaluate the 32-layer models at multiple checkpoints during training on both tracks. We also evaluate a static-routing control in the 10M L32 setting. We separate equal-exposure comparisons from within-seed matched-NLL comparisons, then compare the effect of nonrecent-source masking and count-matched random masking.

We found that AttnRes lowers test NLL at the final checkpoint in all nine paired comparisons and has a positive NLL advantage throughout most of training, providing evidence of greater sample efficiency. However, the endpoint downstream differences are small and mixed, and the broad BLiMP advantage at equal exposure disappears after matching on dev NLL. Finally, we mask selected depth routes in the trained model at evaluation time. This intervention shows that routes from nonrecent blocks causally affect the model's predictions, but the categories that rely more on those routes are not necessarily the categories with larger AttnRes gains over the baseline.

2 Block Attention Residuals

2.1 From Fixed Residual Accumulation to Input-Dependent Depth Routing

In a standard Transformer language model, each layer contains one self-attention sublayer and one feed-forward network sublayer. A sublayer ℓ receives the sum of all earlier residual contributions as follows:

$$\boldsymbol{h}_\ell = \sum_{i=0}^{\ell-1} \boldsymbol{v}_i, \quad \boldsymbol{v}_i = \begin{cases} \boldsymbol{h}_1, & i = 0, \\ f_i(\boldsymbol{h}_i), & 1 \le i < \ell, \end{cases} \tag{1}$$

where $\boldsymbol{h}_1$ is the embedding, $\boldsymbol{v}_i$ is an earlier residual contribution, and $f_i(\cdot)$ is the transformation applied by sublayer i. Every earlier contribution enters with a fixed weight of one. Throughout the paper, we call the earlier contributions $\boldsymbol{v}_i$ *sources*. A learned selection over sources is *routing*, and a weighted connection from one source to a target sublayer is a *route*.

AttnRes instead learns weights over sources along the *depth* dimension (Kimi Team et al., 2026). For target sublayer ℓ, the routed input is

$$\boldsymbol{h}_\ell = \sum_{i<\ell} \alpha_{i\to\ell} \boldsymbol{v}_i, \tag{2}$$

$$\alpha_{i\to\ell} = \mathrm{softmax}_i\left(\boldsymbol{q}_\ell^\top \mathrm{RMSNorm}(\boldsymbol{v}_i)\right). \tag{3}$$

Here, the query $\boldsymbol{q}_\ell$ is a sublayer-specific learnable vector in $\mathbb{R}^d$, so different tokens can use different mixtures of earlier depths.

We use Block AttnRes, which routes among the embedding, completed-block representations, and the current partial block. A block groups a fixed number of consecutive sublayers. A completed block's source is the sum of the sublayer outputs produced within that block, not the residual-stream state at the block boundary, and the current partial block is the running sum of sublayer outputs in the block so far. A final routing mix with its own query is applied over all sources before the language-model head.

2.2 Relation to ELC-BERT Layer Weighting

ELC-BERT (Charpentier and Samuel, 2023) replaces each layer's residual input with a convex combination of all earlier layer outputs, using softmax-normalized weights learned during pre-training. The core difference is whether the learned weights depend on the input token. In ELC-BERT, each "source–target" pair has a single static scalar weight shared across all tokens and inputs while being updated during training. AttnRes instead computes input-dependent weights at run time from each token's source representations in eq. (3), so two tokens in the same sentence can mix earlier depths differently. Static routing was a deliberate choice in ELC-BERT, which describes itself as reintroducing highway-style layer weighting (Srivastava et al., 2015) without the cost of a conditional gating mechanism. AttnRes introduces input dependence through attention over depth.

Two further differences matter for our setup. ELC-BERT weights every individual layer output, while Block AttnRes routes among *block-level* sources. ELC-BERT is an encoder-only masked language model, while we study autoregressive language models. Notably, ELC-BERT's own comparison against its architecture-matched LTG-BERT backbone showed only small and mixed benchmark differences, and its larger margins came from baselines that differ in architecture and

training recipe. We therefore ask whether making depth routing input-dependent improves sample efficiency.

3 Experimental Setup

3.1 Data and Models

We use the official BabyLM 2026 training, development, and test splits of strict-small 10M-word and strict 100M-word corpora (Choshen et al., 2026). We clean the training, development, and test splits independently with the same source-specific pipeline. The cleaning pipeline removed speaker labels and annotation-only lines, normalized Unicode punctuation and whitespace, and applied source-specific rules while preserving document boundaries. We trained byte-level BPE tokenizers with a 16k vocabulary for each corpus separately, and only on its cleaned training split.

All models have a maximum context length of 512 tokens, an embedding dimension of 512, eight attention heads, RMSNorm (Zhang and Sennrich, 2019), SwiGLU with an $8/3$ multiplier (Shazeer, 2020), rotary position embeddings (Su et al., 2021), and Qwen3-Next-style gated attention (Qiu et al., 2025). We apply dropout 0.1 to the input embeddings, attention weights and outputs, and the SwiGLU output. The controlled endpoint experiment crosses 16-layer (L16) and 32-layer (L32) depth with the residual baseline and Block AttnRes on the 10M track. The developmental analyses use the L32 pairs on both tracks. We also train L32 static-routing models on the 10M track as controls for the L32 AttnRes models. The control replaces the input-dependent routing scores with a learned logit for each source at each routing site, shared across tokens. Both routing variants start from uniform mixing weights.

We trained models on both tracks for 10 epochs, and sampled nonoverlapping context windows in a new shuffled order each epoch. Every update contains 262,144 tokens.[1] We trained models of each setting with seeds 1337, 1338, and 1339, using the same value for the model and sampler seed.

3.2 Route Blocks and Parameter Cost

Our L16 model groups four sublayers, or two Transformer layers, per route block. Our L32 model groups eight sublayers, or four Transformer layers, per route block. Both models therefore have eight completed route blocks over their full depth.

The routing parameters add little model capacity. At L16, the parameter size rises from 62.21M for the baseline model to 62.24M for the AttnRes model, an increase of 0.048%. At L32, it rises from 116.23M for the baseline model to 116.30M for the AttnRes model, an increase of 0.060%. Static routing adds 585 parameters, so its total rounds to the same 116.23M as the baseline. All comparisons therefore add less than 0.1% parameters. However, parameter size overhead does not capture runtime cost; we report measured update times in section 3.4.

Model	Params	Time (s)	tokens/s	Ratio
Baseline	116.23M	6.237 ± 0.025	42,029	1.00×
Static	116.23M	8.862 ± 0.280	29,582	1.42×
AttnRes	116.30M	10.230 ± 0.006	25,625	1.64×

Table 1: Runtime measurements for the strict-small L32 configuration on one RTX 3090. Time is seconds per complete optimizer update, reported as the mean $\pm$ sample standard deviation of three repetition means. Each repetition uses 10 warmup and 20 measured updates; each update processes 262,144 tokens. Ratio is relative to the baseline update time.

3.3 Optimization and Checkpoint Selection

We use AdamW with peak learning rate 6×10^{-4}, minimum learning rate 6×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay 0.1. The learning rate follows linear warmup and cosine decay. Warmup lasts 40 updates on 10M and 100 updates on 100M. These optimization-related hyperparameters are fixed in advance and are not tuned separately for different models. The full configuration is reported in table A1.

We use the final scheduled checkpoint in all endpoint comparisons to keep training exposure fixed. We do not select checkpoints by dev NLL.

3.4 Runtime Measurements

Under the configuration described in section 3.3, we benchmark the three model variants separately on the same RTX 3090 using the 10M L32 configuration. The results are shown in table 1. AttnRes takes $1.64\times$ as long per update as the baseline and $1.15\times$ as long as the static-routing model. Appendix D describes the measurement protocol.

[1] The 10M runs use microbatch size 8 and 64 accumulation steps, while the 100M runs use microbatch size 16 and 32 accumulation steps.

Depth	Model	Test NLL ↓	BLiMP ↑	Zero-shot ↑
L16	Baseline	4.1370 ± 0.0017	67.47 ± 0.75	46.02 ± 0.06
L16	AttnRes	$\mathbf{4.1109 \pm 0.0048}$	$\mathbf{68.20 \pm 0.89}$	$\mathbf{46.18 \pm 0.43}$
L32	Baseline	4.1075 ± 0.0111	$\mathbf{68.66 \pm 0.74}$	$\mathbf{46.81 \pm 0.01}$
L32	Static	4.1081 ± 0.0100	68.43 ± 0.56	46.54 ± 0.35
L32	AttnRes	$\mathbf{4.0807 \pm 0.0051}$	68.50 ± 0.33	46.59 ± 0.45

Table 2: Strict-small endpoint results over three paired seeds, reported as mean ± sample standard deviation. Zero-shot is the six-suite mean defined in section 3.5.

3.5 Evaluation

We report test NLL and the official BabyLM zero-shot suites (Choshen et al., 2026). For the zero-shot aggregate, we first average the two GlobalPIQA splits, then report the unweighted mean of the six suites: BLiMP (Warstadt et al., 2020), BLiMP Supplement (Warstadt et al., 2023), EWoK (Ivanova et al., 2025), Entity Tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), and GlobalPIQA (Chang et al., 2025). We report SuperGLUE separately. We run SuperGLUE (Wang et al., 2019), Reading (De Varda et al., 2024), and Age of Acquisition (AoA; Chang and Bergen, 2022) for the L32 baseline and AttnRes models on both tracks, with complete results for all three seed pairs.

4 Endpoint Performance

4.1 Held-Out Language Modeling

Across both tracks, AttnRes lowers test NLL in all nine paired endpoint comparisons.[2] As shown in table 2, AttnRes reduces mean test NLL at both L16 and L32 on the 10M track compared to the baseline model. At L32, AttnRes has lower test NLL than static routing in all three paired seeds. At 100M, the mean L32 reduction is 0.0219 nats, as reported in table 3. The gain is similar across the two depths even though the deeper model has almost twice as many parameters.

The 62.24M-parameter L16 AttnRes model approximately matches the L32 baseline's test NLL. Its test NLL is only 0.0034 higher than the L32 baseline's, which is smaller than the L32 baseline's across-seed standard deviation (0.0111). It does so with half the depth and 46% fewer parameters.

4.2 Endpoint Behavior

The endpoint benchmark differences are small and mixed. As shown in table 2, at L16, AttnRes raises mean BLiMP by 0.73 points and the six-suite mean by 0.16 points from the baseline. At L32, the differences reverse to -0.16 and -0.23 points, respectively.[3] The endpoint results across all three model variants show no evidence of a broad downstream improvement.

Table 3 reports the full three-seed L32 results on both tracks. At 10M, AttnRes is higher on EWoK, Entity Tracking, and COMPS, but lower on BLiMP, the Supplement, and GlobalPIQA. Mean SuperGLUE rises from 63.36 to 63.92, but the paired difference is positive for two seeds and negative for one. The complete NLP and Overall aggregates are slightly lower for AttnRes, while the Human-like aggregate is nearly unchanged.[4] At 100M, AttnRes is higher on BLiMP, EWoK, Entity Tracking, COMPS, GlobalPIQA, and SuperGLUE, but lower on the Supplement and Reading. AoA and the AoA-dependent Human-like aggregate are also lower, although the limited AoA fitting coverage makes these differences difficult to interpret. The 100M NLP aggregate rises from 54.51 to 55.09.

4.3 NLL by Context Position

We test whether the final NLL gain is concentrated at tokens with long preceding contexts. For each final 10M checkpoint, we compute token-weighted dev-NLL differences in the context-position bins 1–4, 5–16, 17–64, 65–256, and 257–512.

In fig. 1, the improvement is approximately flat across positions 5–512, while positions 1–4 show a smaller gain. Therefore, we find no evidence that

[2] We report paired differences and descriptive variability rather than significance tests: each configuration has only three training seeds, and comparisons across configurations reuse the same seed labels and data.

[3] The differences are computed before rounding the aggregate scores.

[4] The AoA scorer used here assigns zero when the fitted correlation does not pass its p-value threshold ($p > 0.1$).

Model	Test NLL ↓	Overall	NLP	SuperGLUE	Human-like	Reading	AoA
Strict-small (10M)							
Baseline	4.1075 ± 0.0111	39.03 ± 0.03	49.18 ± 0.04	63.36 ± 0.30	3.51 ± 0.02	7.01 ± 0.04	0.00 ± 0.00
ATTNRES	**4.0807 ± 0.0051**	38.94 ± 0.32	49.06 ± 0.42	63.92 ± 0.84	3.52 ± 0.06	7.04 ± 0.13	0.00 ± 0.00
Strict (100M)							
Baseline	3.1088 ± 0.0006	43.15 ± 0.17†	54.51 ± 0.18	67.52 ± 0.50	3.43 ± 0.20†	6.89 ± 0.34	−0.04 ± 0.06†
ATTNRES	**3.0868 ± 0.0012**	43.56 ± 0.27†	55.09 ± 0.28	68.18 ± 0.49	3.21 ± 0.26†	6.67 ± 0.13	−0.25 ± 0.43†

Model	BLiMP	Supplement	EWoK	Entity	COMPS	GlobalPIQA
Strict-small (10M)						
Baseline	68.66 ± 0.74	57.07 ± 0.43	50.15 ± 1.10	19.74 ± 0.52	51.13 ± 0.08	34.12 ± 0.50
ATTNRES	68.50 ± 0.33	56.48 ± 1.05	51.14 ± 0.57	20.18 ± 0.71	51.71 ± 0.29	31.48 ± 0.32
Strict (100M)						
Baseline	77.13 ± 0.38	66.60 ± 0.97	56.08 ± 0.65	17.50 ± 1.74	57.57 ± 0.08	39.13 ± 3.45
ATTNRES	78.48 ± 0.60	65.43 ± 1.78	56.46 ± 0.89	19.10 ± 0.26	57.69 ± 0.15	40.28 ± 1.25

Table 3: Full L32 endpoint results on both BabyLM tracks. All values are three-seed means ± sample standard deviations. The upper panel reports test NLL, leaderboard aggregates, and individual SuperGLUE, Reading, and AoA scores, while the lower panel reports the six zero-shot suites. GlobalPIQA first averages its parallel and nonparallel splits. NLP averages the six zero-shot suites with SuperGLUE, Human-like averages Reading with AoA, and Overall averages all nine components. Four of the six 100M AoA evaluations fit only 5–8 words; † marks AoA and aggregates that depend on it.

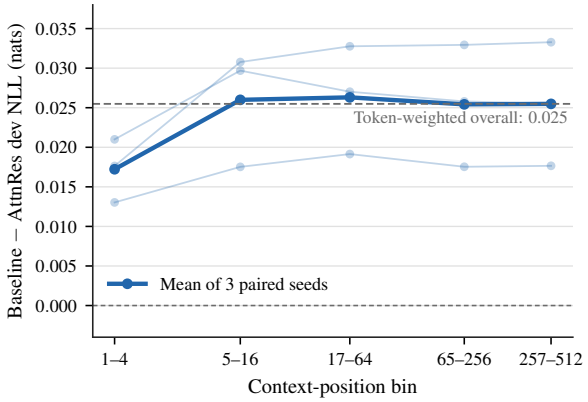


Figure 1: Final 10M dev-NLL differences across token context positions. Thin lines are paired seeds and the thick line is their mean. The lower dashed line marks zero; the upper dashed line is the token-weighted overall mean.

the gain increases with position within the 512-token context.

5 Developmental Analysis

The endpoint results leave two different explanations open: **(1)** ATTNRES may obtain lower NLL after seeing the same amount of training data, or **(2)** it may show different BLiMP behavior even when the two models have the same dev NLL. To separate these two questions, we compare ATTNRES and baseline models throughout pre-training through the lens of **equal exposure**, which holds the number of words seen fixed, and **matched NLL**, which holds dev NLL fixed.

5.1 Dev NLL at Equal Exposure

For seed s and exposure t, we define

$$\Delta L_s(t) = L_{\text{baseline},s}(t) - L_{\text{ATTNRES},s}(t), \quad (4)$$

where $L_{m,s}(t)$ is the mean token-level NLL of model m, with seed s, on the fixed dev set after exposure t. We call $\Delta L_s(t)$ the NLL advantage. A positive value means that ATTNRES has lower dev NLL than the paired baseline after seeing the same number of words. A larger value indicates a larger NLL reduction at that exposure.

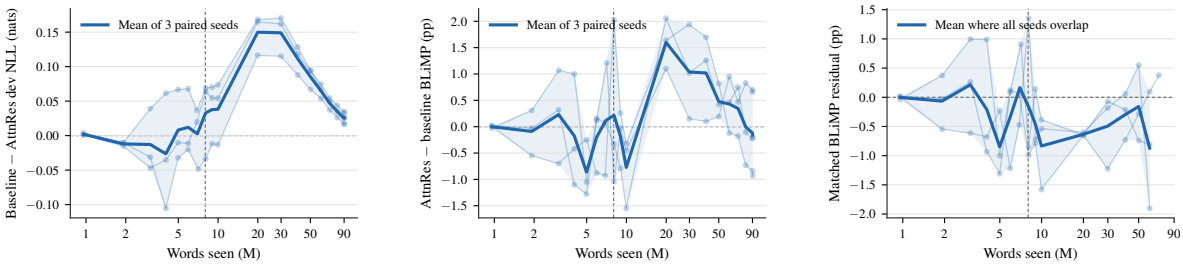
The NLL advantage is not a single checkpoint spike. In fig. 2a, it forms a mid-training plateau. It peaks around 20–30M words at 0.12–0.17 nats across paired seeds, then narrows to 0.018–0.033 nats by the final checkpoint (per-seed values in table B1). Because the NLL advantage remains positive over most of training, ATTNRES reaches comparable dev NLL values after seeing fewer training words. This provides evidence of greater sample efficiency under the matched training setup.

5.2 BLiMP at Equal Exposure

We define the paired BLiMP difference as

$$\Delta B_s(t) = B_{\text{ATTNRES},s}(t) - B_{\text{baseline},s}(t), \quad (5)$$

where $B_{m,s}(t)$ is the overall BLiMP accuracy for model m with seed s after exposure t. At



(a) Equal-exposure dev NLL (b) Equal-exposure BLiMP (c) Matched-NLL BLiMP

Figure 2: Developmental results for the 10M L32 models at 19 paired checkpoints. These checkpoints span the full ten-epoch training run over the strict-small corpus. Thin lines show the three paired seeds, thick lines show their mean, and bands show the seed range rather than a confidence interval. Vertical dashed lines mark the end of warmup. Panel (c) plots all valid within-seed matches, but its thick mean uses only checkpoint labels with overlap for all three seeds.

equal exposure, overall BLiMP is higher for AttnRes through much of the mid-to-late trajectory in fig. 2b. The difference is most consistent from 20M to 50M words. All three seed pairs favor AttnRes at these four checkpoints, with mean differences of 1.60, 1.04, 1.02, and 0.48 percentage points. The difference narrows after 50M words and becomes mixed across seeds at the final checkpoint. However, the seed trajectories are noisy, and the endpoint difference is not stable.

5.3 BLiMP at Matched Dev NLL

We next compare checkpoints at the same dev NLL within each seed. We sort the baseline checkpoints by exposure. When the baseline NLL has a local upward fluctuation, we apply equal-weight pool-adjacent-violators regression (PAVA; Ayer et al., 1955)[5] to make only the NLL axis non-increasing. Without the monotonicity correction, the baseline NLL curve may cross a specific intermediate point multiple times, and then eq. (6) would give multiple results for one L_A. In the 10M L32 trajectories, this correction is applied only to seed 1337 and changes only its final two baseline NLL values. We do not smooth the BLiMP values because a model's BLiMP performance is not assumed to improve monotonically.

For an AttnRes checkpoint with dev NLL L_A, if L_A lies strictly between two distinct adjacent baseline NLL values, we linearly interpolate the baseline BLiMP score:

$$\widehat{B}_B(L_A) = B_{B,j} + \frac{\widetilde{L}_{B,j} - L_A}{\widetilde{L}_{B,j} - \widetilde{L}_{B,j+1}} \left(B_{B,j+1} - B_{B,j}\right), \tag{6}$$

where $\widetilde{L}_{B,j}$ denotes the adjusted dev NLL of baseline checkpoint j, and $B_{B,j}$ denotes its original BLiMP score. If L_A equals an adjusted NLL value shared by multiple checkpoints on a PAVA plateau, we instead use the mean of their original BLiMP scores. The matched residual is $B_A - \widehat{B}_B(L_A)$. We never match across seeds and never extrapolate outside the observed baseline NLL range. The final AttnRes checkpoint is below the minimum baseline NLL for all three seeds, so it has no matched final estimate.

Figure 2c shows that the broad BLiMP advantage *disappears* after matching on dev NLL. The all-point mean residuals are -0.382, -0.290, and -0.281 percentage points for seeds 1337, 1338, and 1339. The valid matched counts and monotonicity adjustments are listed in table B1. Together with the equal-exposure result in section 5.1, this shows that AttnRes reaches comparable dev NLL at lower training exposure, but we find no broad BLiMP advantage at the same dev NLL.

5.4 Matched-NLL Results Across BLiMP Categories

Overall BLiMP can hide differences among linguistic categories. We therefore run a post-hoc analysis of all 13 categories on the 100M trajectories. We use the 100M track for this analysis because the 10M–70M checkpoints occur during its

[5] A correction method that finds the monotonic sequence closest to the original sequence under squared-error loss.

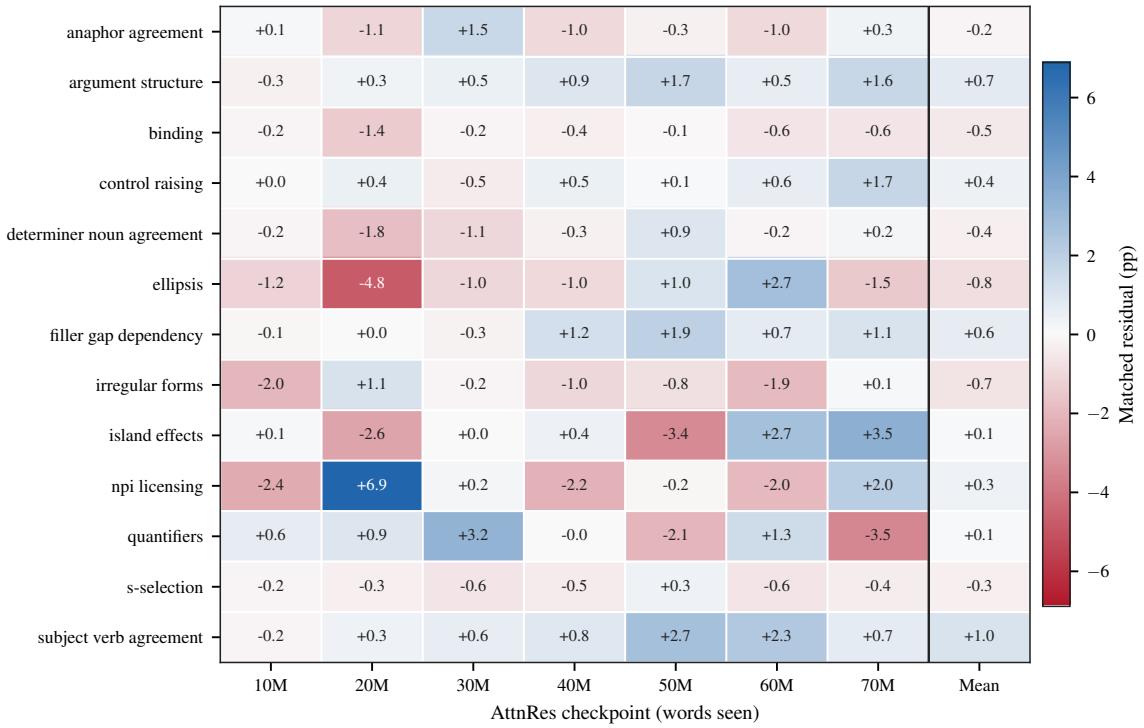


Figure 3: Matched-NLL residuals across all 13 BLiMP categories in the 100M trajectories. Each column identifies the exposure of the AttnRes checkpoint. For each category and seed, baseline accuracy is linearly interpolated at the same dev NLL and may come from a different exposure. Each cell shows AttnRes accuracy minus the interpolated baseline accuracy, averaged over three paired seeds, in percentage points. The final column averages the seven checkpoint-level values from 10M to 70M words seen. Positive values favor AttnRes.

first training epoch. This limits repeated exposure to the same training data. For every category and seed, we match the 10M–70M AttnRes checkpoints to the paired baseline at equal dev NLL. The 80M and 90M AttnRes checkpoints have lower dev NLL than any of the first-epoch baseline checkpoints (10M–90M), so we do not extrapolate matched BLiMP scores for them.

The residuals in fig. 3 vary across both categories and checkpoints. Argument structure and subject–verb agreement are positive at most checkpoints, while binding and S-selection are mostly negative. Several other categories change sign over training, and their checkpoint-level residuals can be much larger than their trajectory means. Across the full 10M–70M window, the category means range from -0.8 to $+1.0$ percentage points. Because this post-hoc analysis has 13 categories, three paired seeds, and no multiplicity-adjusted confirmatory test, it does not establish a broad or category-specific AttnRes advantage.

6 Depth-Routing Analysis

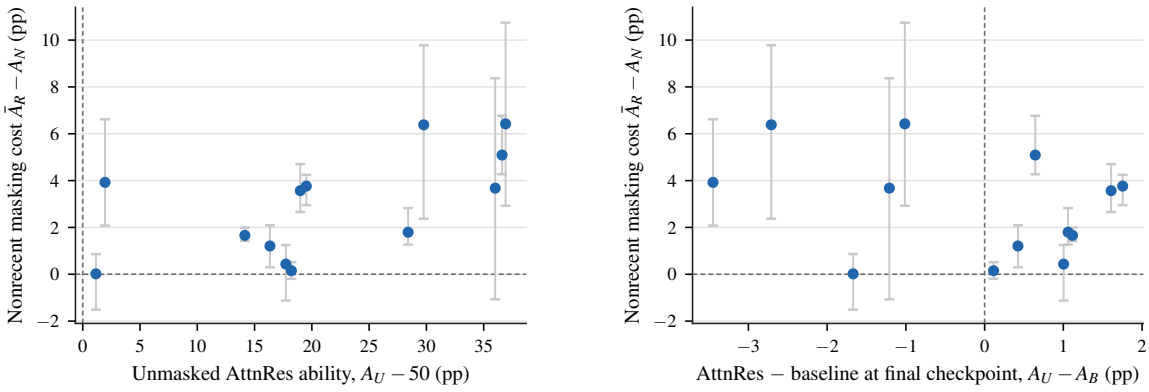
The previous matched-NLL analysis asks whether linguistic behavior differs between models and whether this difference changes over the course of pre-training. However, it does not show whether the trained AttnRes model uses representations from earlier route blocks. We address that question with a masking intervention on the final 10M L32 models after ten epochs of training.[6]

6.1 Masking Procedure and Analysis Measures

Following deletion-style analyses of attention heads (Michel et al., 2019; Voita et al., 2019), we test which depth routes the trained model relies on by masking selected sources at evaluation time.

For each training seed and BLiMP linguistic category, let A_U be unmasked AttnRes accuracy and A_B be the paired baseline accuracy. Let A_N be AttnRes accuracy when we mask *nonrecent* completed-block sources. Specifically, in an AttnRes model, the most recently completed route block has age 1, the preceding block has age 2, and so on. We mask nonrecent completed-block

[6]Unlike the category analysis in the previous section, which used the 100M trajectories, every quantity in this section comes from the final 10M models; their exposure of roughly 90M words is unrelated to the 100M track's 90M checkpoint.



(a) Absolute ability

(b) Relative gain over baseline

Figure 4: Nonrecent-route dependence across the 13 BLiMP linguistic categories. Points average three training seeds. Vertical whiskers show the range across five fixed random-mask draws after averaging each draw across training seeds, not a training-seed confidence interval. Panel (a) compares the cost with unmasked ability above chance, while panel (b) compares it with the ATTNRES gain over the paired baseline. Draw-level values are reported in table C1.

sources with age at least 2, while retaining the embedding, the current partial block, and the most recently completed block. We set the logits of masked sources to $-\infty$ before the depth-wise softmax, so the weights of the remaining sources are renormalized to sum to one. For control draw d, let $A_{R,d}$ be accuracy after masking the same number of *randomly chosen* completed-block sources, with the most recently completed block also eligible for masking. Let $\bar{A}_R$ be the mean accuracy over five fixed random-mask draws. Each draw masks, at every target sublayer, the same number of completed-block sources as the nonrecent mask at that sublayer. We run this analysis on BLiMP because its 13 linguistic categories support category-level contrasts and its zero-shot format is cheap to re-evaluate under every mask. The remaining suites are reported at endpoint in table 3.

We report three analysis measures:

$$\text{absolute ability} = A_U - 50, \quad (7)$$

$$\text{relative gain} = A_U - A_B, \quad (8)$$

$$\text{nonrecent masking cost} = \bar{A}_R - A_N. \quad (9)$$

Absolute ability measures unmasked performance above the 50% chance level. Relative gain compares unmasked ATTNRES with its paired baseline. Nonrecent masking cost compares the nonrecent mask with count-matched random masking. We compute these measures within each training seed and linguistic category before averaging across seeds.

6.2 Nonrecent Sources Affect Predictions

Masking nonrecent completed-block sources is more damaging than masking the same number of randomly selected sources. Overall BLiMP accuracy falls from 68.5% without masking to 55.1% under nonrecent-source masking and 57.9% under count-matched random masking. The equally averaged excess cost across the 13 categories is approximately 2.93 percentage points.[7] The category-specific nonrecent masking costs range from 0.02 to 6.42 percentage points, as detailed in table 4. Because the intervention directly changes the available routes in a trained ATTNRES model, it shows that **nonrecent sources make a causal contribution to that model's predictions**.

In fig. 4a, nonrecent masking cost increases with the model's absolute ability above chance. Using the mean nonrecent masking cost across the five random masks, the Pearson correlation across the 13 categories is $r = 0.600$. This relationship suggests a headroom effect: **categories with stronger unmasked performance can show a larger measurable masking cost.** When each random mask is analyzed separately, the correlations range from 0.156 to 0.842, so the exact strength depends on the random control.

6.3 Route Dependence and Relative Gain

We find no clear evidence of a positive association between nonrecent-route dependence and the

[7] The count-weighted average cost is 2.80 percentage points.

BLiMP category	Ability	Gain	Nonrecent cost
Anaphor agreement	36.91	−1.01	6.42 ± 3.72
Argument structure	18.99	1.61	3.57 ± 0.86
Binding	18.22	0.11	0.15 ± 0.27
Control and raising	14.15	1.12	1.66 ± 0.23
Determiner–noun agreement	36.60	0.64	5.09 ± 1.02
Ellipsis	17.73	1.00	0.43 ± 0.98
Filler–gap dependency	19.52	1.75	3.77 ± 0.55
Irregular forms	36.00	−1.21	3.68 ± 4.08
Island effects	1.15	−1.67	0.02 ± 0.92
NPI licensing	1.95	−3.45	3.92 ± 1.92
Quantifiers	29.76	−2.71	6.38 ± 2.87
S-selection	28.40	1.06	1.79 ± 0.66
Subject–verb agreement	16.34	0.42	1.21 ± 0.76

Table 4: Routing contrasts for the final 10M L32 models, in percentage points. Ability is $A_U - 50$, gain is $A_U - A_B$, and nonrecent cost is $\bar{A}_R - A_N$. The first two columns are means over three training seeds. The final column reports the mean ± sample standard deviation over five fixed random masks after averaging each mask over training seeds.

ATTNRES gain over the baseline on BLiMP. Using the mean nonrecent masking cost across the five random masks, the Pearson correlation across the 13 categories in fig. 4b is $r = -0.326$. The individual-draw correlations range from -0.532 to 0.058. Therefore, categories that rely more on nonrecent routes are not necessarily the categories with larger ATTNRES improvements over the baseline.

7 Conclusion

ATTNRES consistently improves held-out language modeling in data-limited BabyLM training while adding less than 0.1% parameters. Under the shared training recipe, it lowers test NLL in all nine paired endpoint comparisons at the same training exposure. We further separate static routing from input-dependent routing: in the 10M L32 setting, ATTNRES also achieves lower test NLL than static routing in all three paired seeds. Its positive NLL advantage over most of training provides evidence of greater sample efficiency, although it takes 1.64 times as long per update as the baseline in our implementation. Moreover, the 16-layer ATTNRES model approximately matches the 32-layer baseline's test NLL with 46% fewer parameters.

These language-modeling gains do not produce a broad downstream advantage. Endpoint benchmark differences between ATTNRES and baseline models are small and mixed. The same holds for ATTNRES versus static routing on the zero-shot suites. The overall BLiMP advantage observed at equal exposure disappears at within-seed matched dev NLL.

The trained ATTNRES models nevertheless make causal use of nonrecent completed-block sources. Masking them is more damaging than masking the same number of randomly selected sources. However, we find no clear evidence that categories with greater dependence on nonrecent sources achieve larger BLiMP improvements over the baseline.

Limitations

All comparisons use three paired training seeds, so the reported standard deviations and ranges are descriptive and do not support fine-grained significance claims. In the 100M category analysis, we exclude the 80M and 90M ATTNRES checkpoints because their dev NLL is below the range of the first-epoch baseline checkpoints.

We evaluate static routing only in the 10M L32 setting, using three seeds. We use the same optimization settings for all model variants rather than tuning each variant separately. Our runtime measurements compare update speed; we did not train the models for the same amount of GPU time.

At each routing site with completed-block sources, the nonrecent mask retains only the most recently completed block. The random control masks the same number of completed-block sources at that site, leaving one randomly selected block. The two masks are identical when the random control retains the most recent block.

The random masking control matches the *number* of sources masked by the nonrecent intervention, but not their routing probability mass or expected severity. The routing intervention applies only to the final 10M L32 ATTNRES models and establishes causal use within those models.

Acknowledgements

The author would like to thank Alex Warstadt for his suggestions on the experimental design and manuscript writing. The author used OpenAI Codex to assist with planning and refining parts of the experimental code, checking analyses, and revising manuscript text, including methodological descriptions and appendices. The author reviewed and revised the outputs and takes responsibility for the experiments, results, and final manuscript.

References

Miriam Ayer, H Daniel Brunk, George M Ewing, William T Reid, and Edward Silverman. 1955. An empirical distribution function for sampling with incomplete information. *The annals of mathematical statistics*, pages 641–647.

Tyler A Chang, Catherine Arnett, Abdelrahman Sadallah, Abdelrahman Eldesokey, Abeer Kashar, Abolade Daud, Abosede Grace Olanihun, Adamu Labaran Mohammed, Adeyemi Praise, Adhikarimayum Meerajita Sharma, et al. 2025. Global PIQA: Evaluating commonsense reasoning across 100+ languages and cultures. *arXiv preprint arXiv:2510.24081*.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Georges Gabriel Charpentier and David Samuel. 2023. Not all layers are equally as important: Every layer counts BERT. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 238–252, Singapore. Association for Computational Linguistics.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Andrea Gregor De Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 english sentences, aligned with existing eeg and reading time data. *Behavior Research Methods*, 56(5):5190–5213.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, Yutian Chen, Junjie Yan, Ming Wei, Y. Zhang, Fanqing Meng, Chao Hong, Xiaotong Xie, Shaowei Liu, Enzhe Lu, and 18 others. 2026. Attention residuals. *Preprint*, arXiv:2603.15031.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949, Dubrovnik, Croatia. Association for Computational Linguistics.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Component	Strict-small 10M	Strict 100M
Training data	Official BabyLM 2026 training split	Official BabyLM 2026 training split
Tokenizer	16k-vocabulary byte-level BPE	16k-vocabulary byte-level BPE
Context length	512 tokens	512 tokens
Width and heads	$d = 512$, 8 heads	$d = 512$, 8 heads
Depths	L16 and L32	L32
Shared architecture	RMSNorm, SwiGLU $8/3$, RoPE, attention gate	RMSNorm, SwiGLU $8/3$, RoPE, attention gate
Dropout	0.1	0.1
Route block size	4 sublayers at L16; 8 sublayers at L32	8 sublayers
Sampler	shuffled nonoverlapping windows	shuffled nonoverlapping windows
Microbatch / accumulation	8 / 64	16 / 32
Tokens per update	262,144	262,144
Updates	471	4,797
Optimizer	AdamW	AdamW
Peak / minimum learning rate	6×10^{-4} / 6×10^{-5}	6×10^{-4} / 6×10^{-5}
Adam β_1, β_2	0.9, 0.95	0.9, 0.95
Weight decay	0.1	0.1
Schedule	40-update warmup, then cosine decay	100-update warmup, then cosine decay
Training and sampler seeds	1337, 1338, 1339	1337, 1338, 1339
Endpoint selection	final checkpoint	final checkpoint

Table A1: Experimental configuration. The L16 AttnRes block contains two Transformer layers, and the L32 block contains four Transformer layers. Both settings therefore expose eight completed route blocks over the full network.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in neural information processing systems*, 32.

A Experimental Configuration

Table A1 reports the complete training configuration. Values in each track column apply to every model variant unless the row states otherwise.

B Checkpoint Results

Table B1 records the 10M L32 checkpoint results used in the main text.

Seed	Peak ΔL	Peak	Matched n	Matched ΔB	PAVA
1337	0.1703	30M	15	-0.382	yes
1338	0.1648	20M	15	-0.290	no
1339	0.1164	20M	16	-0.281	no

Table B1: Summary of the 10M L32 checkpoint analyses. Peak values are in nats, and matched ΔB is the mean over all valid matched checkpoints, in percentage points. Matched n counts the AttnRes checkpoints, out of 19, that have a valid estimate after matching within the seed on dev NLL (section 5.3). The final AttnRes checkpoint falls below the minimum baseline dev NLL in every seed.

C Random-Mask Control Variability

The five draws vary only the randomly masked sources. Their range gives the vertical whiskers in fig. 4.

BLiMP category	$A_U - 50$	$A_U - A_B$	**Draw 1**	**Draw 2**	**Draw 3**	**Draw 4**	**Draw 5**	$\bar{A}_R - A_N$
Anaphor agreement	36.91	−1.01	5.21	10.74	3.23	2.93	10.01	6.42
Argument structure	18.99	1.61	4.04	2.78	2.66	3.66	4.70	3.57
Binding	18.22	0.11	0.05	0.52	0.32	−0.19	0.07	0.15
Control and raising	14.15	1.12	1.78	1.99	1.53	1.42	1.58	1.66
Determiner–noun agreement	36.60	0.64	5.25	4.30	4.27	4.87	6.77	5.09
Ellipsis	17.73	1.00	−1.12	0.17	0.67	1.21	1.25	0.43
Filler–gap dependency	19.52	1.75	2.95	4.25	3.43	4.05	4.14	3.77
Irregular forms	36.00	−1.21	7.34	0.81	−1.07	2.94	8.37	3.68
Island effects	1.15	−1.67	0.50	−0.06	0.31	0.86	−1.51	0.02
NPI licensing	1.95	−3.45	5.22	2.63	3.07	6.62	2.08	3.92
Quantifiers	29.76	−2.71	9.78	4.77	2.37	7.18	7.81	6.38
S-selection	28.40	1.06	1.43	2.09	1.27	1.36	2.83	1.79
Subject–verb agreement	16.34	0.42	0.30	2.09	0.94	0.80	1.90	1.21

Table C1: Complete routing intervention audit over 13 BLiMP linguistic categories. Each value is first averaged across three training seeds. Draw columns report $A_{R,d} - A_N$ for five fixed draws of count-matched random masking, in percentage points. The final column is their mean, $\bar{A}_R - A_N$.

D Runtime Measurement Protocol

We tested the three strict-small L32 models on the same RTX 3090. We used PyTorch 2.7.1+cu128, BF16 autocast, and `torch.compile`. Each optimizer update accumulates gradients over 64 microbatches of eight 512-token sequences.

We repeat the benchmark three times for each model and vary the model order across runs. Each run uses 10 warmup updates followed by 20 timed updates. We synchronize CUDA before and after each timed update.

One timed update includes the forward and backward passes, gradient clipping, and the fused AdamW update. We use synthetic inputs already on the GPU. Timing excludes input generation and transfer, compilation, validation, and checkpoint writing.

Language Exposure Matters: Distribution and Order in Multilingual BabyLM Pre-training

JiWoo Hwang[1,2] **Lukas Edman**[1,2] **Alexander Fraser**[1,2,3]

[1]School of Computation, Information and Technology, TU Munich
[2]Munich Center for Machine Learning
[3]Munich Data Science Institute

jiwoo.hwang@tum.de, lukas.edman@tum.de

Abstract

We investigate two aspects of multilingual pre-training with limited data, namely how language exposure is distributed across English, Dutch, and Chinese, and how that exposure is ordered during training. In particular, we compare balanced multilingual exposure to language-dominant exposure, where one language is exposed with more training data than the others. We also compare *mixed training*, where multilingual training data is used throughout training, with *curriculum training*, where monolingual exposure of the dominant language is followed by balanced multilingual exposure. Our results show that multilingual BabyLM performance varies with both the amount and order of language exposure. Mixed training outperforms curriculum learning across all 15 matched configurations, indicating that simultaneous multilingual exposure is more effective than staged exposure in our setting. Several language-dominant mixed training configurations achieve higher scores than the balanced configuration, suggesting that a balanced distribution is not necessarily the highest-performing configuration in our setting. Interestingly, the Dutch-primary configurations achieve the best overall performance.

1 Introduction

While the performance of multilingual pre-trained models has improved over the recent years, billions of tokens are required to achieve these results. In the context of the BabyLM Challenge (Choshen et al., 2026), the focus lies on the learning abilities of language models from a limited amount of training data, thereby shifting attention from scaling pre-training toward efficient language learning. For the multilingual track, this brings attention to how data should be distributed among languages, as well as the order of language exposure.

In this paper, we address these questions in a controlled setting of multilingual exposure with English, Dutch and Chinese. We compare balanced exposure among the three languages with a setup where one language is exposed more than others. We also compare two training schemes: *mixed* and *curriculum* training. In mixed training, the full multilingual dataset is used throughout training. In curriculum training, the same dataset is used, however split into two stages: the model first trains on a monolingual primary language subset, then continues on a balanced multilingual subset containing all three languages. Our contributions are:

- We examine the effect of the amount of language exposure in multilingual BabyLM pre-training by comparing balanced exposure with language-dominant exposure across English, Dutch, and Chinese.
- We explore the effect of exposure order by comparing mixed training with a two-stage curriculum learning strategy, in which monolingual primary language exposure is followed by balanced multilingual exposure.
- We investigate whether strong overall multilingual performance requires balanced training by comparing settings where each language is dominant.

A natural baseline for multilingual training is a

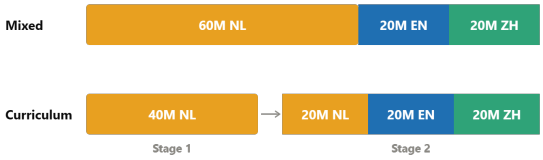


Figure 1: Mixed and curriculum training for the Dutch-primary 60/20/20 configuration. Mixed training shuffles all 100M tokens in a single stage, whereas curriculum training first uses 40M Dutch tokens and then a balanced multilingual stage containing 20M tokens from each language. For this configuration, mixed training achieves a higher multilingual average than curriculum training and produces the best overall result in our experiments.

Proceedings of the Second BabyLM Workshop, pages 904–911
October 29, 2026 ©2026 Association for Computational Linguistics

balanced distribution across languages, however, our results show that this assumption does not hold in our setting[1].

2 Related Work

Multilingual data distribution. One of the issues in multilingual learning is whether training should be balanced across different languages. Prior research on multilingual machine translation has demonstrated that the language sampling distribution has a considerable impact on overall performance. Wang et al. (2020) show that learning how to rebalance training data across languages can improve performance across languages.

Although balanced exposure to multiple languages serves as a natural baseline, previous work has shown that imbalance is occasionally beneficial for cross-lingual transfer. Schäfer et al. (2024) investigate controlled experiments using cloned languages that separate the effects of imbalance from linguistic factors. They show that the dominance of one language positively affects the performance of other minority languages and results in better cross-lingual alignment, an effect that is more prominent in larger models and with more training time. We address a similar problem using the BabyLM framework but with real languages.

Exposure order. Bengio et al. (2009) introduced curriculum learning, which sorts training examples in a meaningful order, from simple to complex or from one domain to another, in order to boost learning efficiency. In the case of multilingual settings, the ordering may define the moment the model forms shared multilingual representations early or first specializes in one language. Previous research on few-shot cross-lingual transfer has demonstrated that the mixed multilingual training procedure outperforms staged target adaptation (Xu and Murray, 2022).

Our comparison between mixed and curriculum training is also motivated by bilingual language acquisition, in which both the amount of exposure and the time at which that exposure takes place are important. A distinction is made between simultaneous bilingual language acquisition, in which more than one language is learned from an early age, and sequential bilingual language acquisition, in which another language is acquired later (Johnson and Newport, 1989). Although our experimental setup cannot be considered as direct representations of bilinguals, this distinction provides a useful motivation for comparing simultaneous multilingual exposure with staged curriculum learning.

3 Experimental Setup

The detailed training setup can be found in Appendix A.

3.1 Data

The training corpus is constructed from Wikipedia dumps[2] for the three languages: English, Dutch, and Chinese in the BabyLM 2026 multilingual track. Wikipedia dumps are used in order to keep the source domain as comparable as possible across the languages, since the BabyLM corpora contain different mixtures of domains. As per the constraint of the track, each configuration contains 100M tokens, where the number of tokens is calculated based on the byte premium (Arnett et al., 2024) of each language.

We use six data configurations, where each configuration has different proportions allocated to each language. The baseline configuration contains an equal number of tokens in all three languages. The five language-dominant configurations allocate the primary language with a higher percentage while the rest is divided equally among the other two languages. We have chosen primary language percentages of 50%, 60%, 70%, 80%, and 90%, resulting in five different configurations as shown in Table 1.

Configuration	Primary	Other 1	Other 2
Balanced	33%	33%	33%
50/25/25	50%	25%	25%
60/20/20	60%	20%	20%
70/15/15	70%	15%	15%
80/10/10	80%	10%	10%
90/5/5	90%	5%	5%

Table 1: Data proportion configurations. Each language-dominant configuration is instantiated with English, Dutch, and Chinese as the primary language respectively.

For each language-dominant split and primary-language pairing, we generate two further subsets for curriculum training (see Section 3.3): a Stage 1 subset containing only primary-language

[1] Code and experimental resources are available at https://github.com/jinx2321/babylm26.

[2] https://huggingface.co/datasets/wikimedia/wikipedia

text and a Stage 2 subset containing the remaining primary-language text together with all non-primary-language text, shuffled randomly. The split is constructed so that each language contributes an equal number of tokens in Stage 2. For instance, in the 60/20/20 configuration, Stage 1 contains 40M tokens from the primary language, while Stage 2 contains the remaining 60M tokens, with 20M tokens from each language, as shown in Figure 1.

3.2 Model

We train a decoder-only transformer from scratch using the Qwen3 architecture (Yang et al., 2025), which results in ca. 110M parameters (768 hidden dimension, 12 layers, 12 attention heads, 2048 feed-forward dimension, 1024 context length). We train a separate byte-level BPE tokenizer for each data configuration, using the training data of the configuration as the corpus for tokenization. This guarantees that the language distribution utilized for tokenization is equal to the distribution seen during the pre-training process.

3.3 Training Strategies and Evaluation

We compare the mixed and curriculum learning strategies across all five language-dominant configurations, as shown in Figure 1. In mixed training, all 100M tokens from the corpus are shuffled in random order and used during the 10-epoch training. The model is exposed to all three languages throughout all training iterations. Curriculum training involves two consecutive training stages using the Stage 1 and Stage 2 subsets from Section 3.1. In Stage 1, the model trains on the subset that contains only the primary language for 10 epochs, and then Stage 2 continues training on the multilingual balanced subset for another 10 epochs, initialized from the Stage 1 checkpoint, including the optimizer and learning rate scheduler states. Stage 1 and Stage 2 together cover the whole 100M-token dataset configuration. Thus, each language is seen the same number of times across both strategies and only the order of exposure differs.

For all models, the official BabyLM evaluation pipeline (Choshen et al., 2026) was used for evaluation, which contains zero-shot and fine-tuning tasks in English, Dutch, and Chinese. The zero-shot tasks include BLiMP (Warstadt et al., 2020), GlobalPIQA (Chang et al., 2025), HellaSwag (Zellers et al., 2019), MultiBLiMP (Jumelet et al., 2026), Winogrande (Sakaguchi et al., 2021), XStoryCloze (Lin et al., 2022b), and XCOMPS (He et al., 2025), while fine-tuning evaluation cover tasks such as ARC (Clark et al., 2018), Belebele (Bandarkar et al., 2024), BMLAMA (Qi et al., 2023), MNLI (Williams et al., 2018), SIB-200 (Adelani et al., 2024), TruthfulQA (Lin et al., 2022a), XNLI (Conneau et al., 2018), and INCLUDE (Romanou et al., 2025). The zero-shot average, fine-tuning average, and overall average scores of each language are reported for all models.

4 Results and Discussion

The full results can be found in Appendix B.

4.1 Impact of Language Exposure Proportion

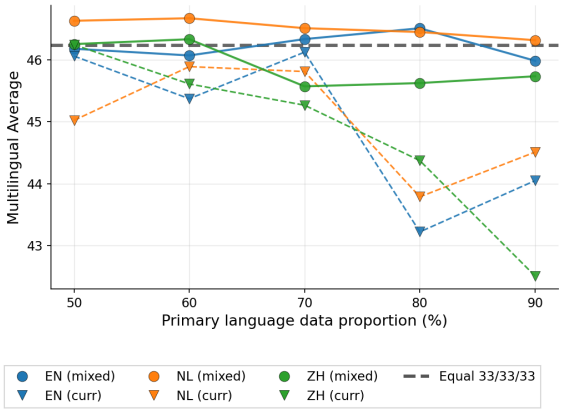


Figure 2: Multilingual average performance across primary-language data proportions, grouped by the language designated as primary and by training strategy. Each point averages performance across English, Dutch, and Chinese, while the dashed horizontal line marks the balanced 33/33/33 baseline.

We analyze the effect of language exposure from two complementary perspectives. Figure 2 compares complete data-allocation configurations using their multilingual average, showing whether language-dominant exposure improves overall performance relative to the balanced baseline. Figure 3 instead examines each language separately by relating its proportion in the training data to performance in that same language.

We first compare the language-dominant configurations with the balanced 33/33/33 mixed training baseline, which obtains a multilingual average of 46.23. For English-primary mixed training, the 70/15/15 and 80/10/10 configurations outperform the balanced baseline, with performance peaking at 46.51 for 80/10/10.

Dutch-primary mixed training performs more

consistently: all five Dutch-primary configurations outperform the balanced baseline. The highest multilingual average is 46.67 at 60/20/20, an improvement of 0.44 points over the baseline and the best overall result among all configurations. Chinese-primary training provides smaller benefits. The 50/25/25 and 60/20/20 configurations slightly exceed the baseline, reaching a maximum of 46.33 at 60/20/20, whereas configurations with higher Chinese proportions fall below it.

These comparisons show that the balanced configuration is not the highest-scoring configuration in this setting, although the benefit of language dominance depends on both the primary language and the degree of imbalance. English-primary performance peaks at 80/10/10, while Dutch-primary and Chinese-primary performance peak at 60/20/20. At 90/5/5, only the Dutch-primary model remains above the balanced baseline, indicating that extreme imbalance is generally less effective.

This finding is important since a balanced distribution is a natural baseline when all three evaluation languages are weighted equally. Nevertheless, for each primary language, at least one language-dominant mixed training configuration outperforms the balanced model. The advantage is therefore not limited to English dominance as the Dutch-primary 60/20/20 configuration achieves the best overall result, while the Chinese-primary 50/25/25 and 60/20/20 configurations also surpass the balanced baseline. At the same time, the weaker results under the most extreme allocations suggest a trade-off between increasing exposure to the primary language and maintaining sufficient exposure to the other languages.

4.2 Mixed vs. Curriculum Learning

Next, we compare mixed training with the curriculum learning strategy. As shown in Figure 2, mixed training achieves a higher multilingual average than curriculum training in all 15 matched language-dominant configurations. The difference is generally larger at high levels of dominance, although it does not increase monotonically. For example, at 80/10/10, mixed training outperforms curriculum training by 3.29 points in the English-primary setting and by 2.66 points in the Dutch-primary setting.

The major weakness of curriculum training is its decline in performance of non-primary languages under high dominance conditions. In the English-primary 80/10/10 case, the Dutch and Chinese average scores of mixed training models are 46.59 and 44.42, respectively, whereas the corresponding figures for curriculum models are 40.81 and 40.19. This suggests that the large monolingual first stage encourages specialization on the primary language, and the subsequent multilingual stage does not fully compensate for this, consistent with findings that simultaneous multilingual exposure outperforms staged adaptation (Xu and Murray, 2022).

4.3 Language-Specific Effects of Exposure

Concerning the language-specific results in Figure 3, the clearest relationship between data proportion and performance appears in zero-shot evaluation. The linear trends are positive for all three languages, with R^2 values of 0.504 for English, 0.502 for Dutch, and 0.516 for Chinese. These similar values indicate that the proportion of a language in the pre-training data has a moderate relationship with zero-shot performance in that same language.

The relationship is weaker for finetuned performance. English and Dutch show similar R^2 values of 0.369 and 0.367, respectively, whereas the relationship is much weaker for Chinese, with $R^2 = 0.140$. This difference is also visible in the trend lines as English and Dutch finetuned performance generally increases with additional same-language exposure, while Chinese finetuned scores vary less consistently with the proportion of Chinese data.

The language-specific overall averages combine the zero-shot and finetuned results. The relationship is strongest for Dutch ($R^2 = 0.483$), followed by English ($R^2 = 0.451$), and is weaker for Chinese ($R^2 = 0.276$). Overall, Figure 3 shows that same-language data proportion is most consistently related to zero-shot performance. The dispersion around the trend lines also shows that data proportion alone does not fully account for the differences among configurations.

5 Conclusion

Our work studies the impact of language distributions and order of exposure on BabyLM multilingual pre-training with English, Dutch, and Chinese on a 100M-token budget. By comparing balanced and language-dominant data distributions with mixed and curriculum training strategies, we examined how both the amount and order of multilingual exposure shape evaluation performance.

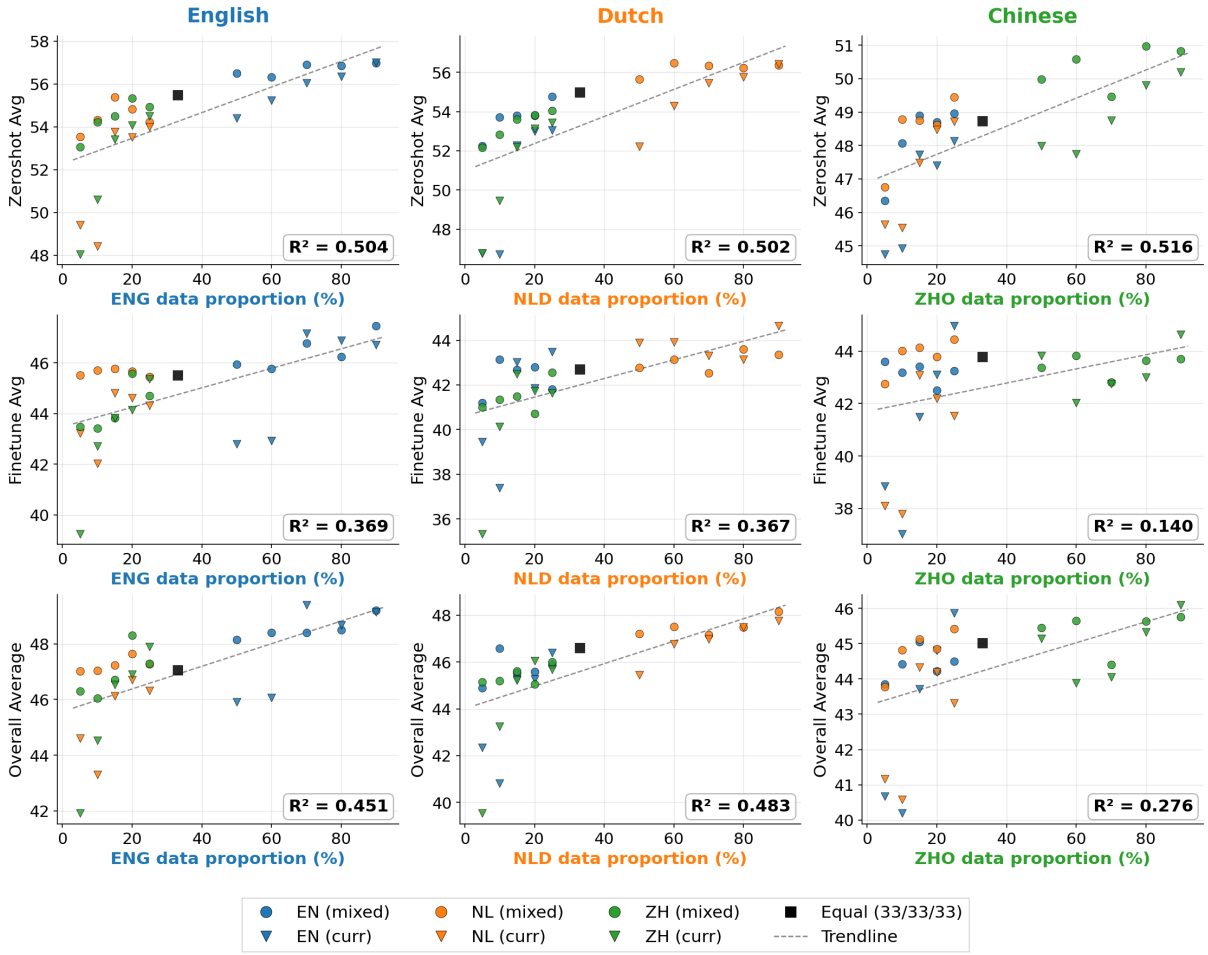


Figure 3: Relationship between language-specific data proportion and evaluation performance. Columns show English, Dutch, and Chinese data proportions, while rows show zero-shot, finetuned, and overall average performance for each language. Dashed lines indicate linear trends across all models, with R^2 reported in each panel. The black square represents the balanced 33/33/33 baseline.

We find that mixed training outperforms curriculum learning. Although the size of the difference does not increase monotonically, the gap is generally larger under highly imbalanced allocations, where curriculum training shows substantial declines in non-primary-language performance. Hence, simultaneous multilingual training should be preferred to sequential monolingual specialization and multilingual adaptation.

In addition, perhaps more surprisingly, pretraining on balanced data is not the optimal approach in our experiments. For mixed training, even though we consider a balanced configuration as the baseline, a certain level of imbalance in favor of one language improves the performance of all three primary languages, suggesting that the effect of imbalance is not language-specific. However, the advantage decreases for extremely large imbalances, indicating a trade-off between strengthening the dominant language and preserving cross-lingual coverage. However, since we evaluate multiple language-dominant configurations against a single balanced configuration and conduct only one run per setting, these results should be interpreted descriptively rather than as evidence that imbalanced training is generally preferable to balanced training.

Limitations

Our study is limited to only three languages (English, Dutch, and Chinese) within one domain (Wikipedia), thus it is uncertain whether our findings will apply to other languages or domains. All experiments use a fixed size of 100M tokens and a single model architecture (Qwen3). Results may vary for other scales and other architectures. The training procedure uses a separate tokenizer for every experiment based on the particular training

dataset. Hence, some variations may be due to tokenizer settings in addition to differences in the data distribution. However, this choice was made in order to ensure that the language distribution for tokenization and model pre-training is the same. Moreover, due to limited computational resources, we conduct only one training procedure per configuration with a fixed random seed. Therefore, we do not estimate variance across training runs. The curriculum learning strategy we explore is applied using a fixed protocol with two stages, thus different protocols such as gradual mixing or additional stages may produce different results.

Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project b279bb. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

This work was co-funded by the European Union (ERC, EPICAL, 101141712). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. Sib-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245.

Catherine Arnett, Tyler A Chang, and Benjamin Bergen. 2024. A bit of a problem: Measurement disparities in dataset sizes across languages. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages@ LREC-COLING 2024*, pages 1–9.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Tyler A Chang, Catherine Arnett, Abdelrahman Eldesokey, Abdelrahman Sadallah, Abeer Kashar, Abolade Daud, Grace Olanihun Abosede, Labaran Mohammed Adamu, Adeyemi Praise, Meerajita Sharma Adhikarinayum, and 1 others. 2025. Global piqa: Evaluating physical commonsense reasoning across 100+ languages and cultures.

Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop. *Preprint*, arXiv:2602.20092.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2475–2485.

Linyang He, Ercong Nie, Sukru Samet Dindar, Arsalan Firoozi, Adrian Florea, Van Nguyen, Corentin Puffay, Riki Shimizu, Haotian Ye, Jonathan Brennan, and 1 others. 2025. Xcomps: A multilingual benchmark of conceptual minimal pairs. In *Proceedings of the 7th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 75–81.

Jacqueline S Johnson and Elissa L Newport. 1989. Critical period effects in second language learning: The influence of maturational state on the acquisition of english as a second language. *Cognitive psychology*, 21(1):60–99.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2026. Multiblimp 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 3214–3252.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, and 1 others.

2022b. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 9019–9052.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2023. Cross-lingual consistency of factual knowledge in multilingual language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10650–10666.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Zeming Chen, Mohamed Haggag, Alfonso Amayuelas, and 1 others. 2025. Include: Evaluating multilingual language understanding with regional knowledge. In *International Conference on Learning Representations*, volume 2025, pages 83291–83322.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Anton Schäfer, Shauli Ravfogel, Thomas Hofmann, Tiago Pimentel, and Imanol Schlag. 2024. The role of language imbalance in cross-lingual generalisation: Insights from cloned language experiments. *arXiv preprint arXiv:2404.07982*.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 8526–8537.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pages 1112–1122.

Haoran Xu and Kenton Murray. 2022. Por qué não utiliser alla språk? mixed training with gradient optimization in few-shot cross-lingual transfer. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2043–2059.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 4791–4800.

A Training Details

Table 2 summarizes the shared training setup used for all models. All configurations use the same model size, optimization settings, context length, and random seed.

Setting	**Value**
Model size	$\sim$110M parameters
Tokenizer	Byte-level BPE
Vocabulary size	32,000
Context length	1024
Optimizer	AdamW
Learning rate	6×10^{-4}
(β_1, β_2)	$(0.9,\ 0.95)$
Weight decay	0.1
LR schedule	Cosine with warmup
Warmup	5% of steps
Batch size	128 sequences
Epochs (mixed)	10 over 100M tokens
Epochs (curriculum)	10 per subset
Random seed	42
Hardware	1$\times$ NVIDIA H100

Table 2: Training setup used for all models.

B Full Results

Table 3 reports the complete results for all 31 models. We include zero-shot, finetuned, language-specific average, and multilingual average scores for each configuration to show how language distribution and training strategy affect both individual languages and overall multilingual performance.

Model		English			Dutch			Chinese			Multi
Config	**Strategy**	ZS	FT	Avg	ZS	FT	Avg	ZS	FT	Avg	Avg
Balanced baseline											
33/33/33	Mixed	55.48	45.52	47.06	54.99	42.70	46.61	48.73	43.78	45.03	46.23
English-primary											
50/25/25	Mixed	56.50	45.94	48.15	54.77	41.80	45.89	48.96	43.25	44.50	46.18
	Curr	54.39	42.79	45.91	53.05	43.47	46.39	48.13	**44.97**	45.87	46.06
60/20/20	Mixed	56.31	45.78	48.40	53.80	42.80	45.59	48.70	42.50	44.22	46.07
	Curr	55.23	42.93	46.06	52.99	41.86	45.25	47.41	43.11	44.80	45.37
70/15/15	Mixed	56.91	46.78	48.41	53.79	42.68	45.53	48.90	43.41	45.06	46.33
	Curr	56.04	47.17	**49.39**	52.26	43.03	45.26	47.73	41.49	43.72	46.12
80/10/10	Mixed	56.87	46.25	48.51	53.72	43.13	46.59	48.07	43.19	44.42	46.51
	Curr	56.35	46.88	48.67	46.72	37.40	40.81	44.92	37.04	40.19	43.22
90/5/5	Mixed	56.98	**47.47**	49.21	52.24	41.20	44.89	46.34	43.60	43.85	45.98
	Curr	**57.00**	46.71	49.14	46.77	39.44	42.34	44.75	38.84	40.68	44.05
Dutch-primary											
50/25/25	Mixed	54.23	45.46	47.27	55.65	42.76	47.20	49.45	44.46	45.42	46.63
	Curr	54.00	44.31	46.32	52.23	43.88	45.44	48.72	41.54	43.31	45.02
60/20/20	Mixed	54.82	45.66	47.65	**56.48**	43.14	47.51	48.62	43.79	44.85	**46.67**
	Curr	53.51	44.61	46.71	54.29	43.91	46.77	48.48	42.20	44.19	45.89
70/15/15	Mixed	55.40	45.77	47.24	56.35	42.52	47.16	48.75	44.14	45.13	46.51
	Curr	53.77	44.81	46.13	55.45	43.30	46.98	47.48	43.09	44.33	45.81
80/10/10	Mixed	54.32	45.70	47.05	56.24	43.61	47.48	48.78	44.02	44.82	46.45
	Curr	48.42	42.02	43.31	55.75	43.13	47.48	45.55	37.79	40.58	43.79
90/5/5	Mixed	53.54	45.51	47.01	56.37	43.37	**48.16**	46.77	42.75	43.77	46.32
	Curr	49.42	43.22	44.61	56.42	**44.66**	47.76	45.63	38.10	41.17	44.51
Chinese-primary											
50/25/25	Mixed	54.93	44.70	47.30	54.05	42.56	46.00	49.98	43.37	45.46	46.25
	Curr	54.49	45.38	47.90	53.43	41.65	45.69	47.98	43.83	45.14	46.24
60/20/20	Mixed	55.33	45.57	48.30	53.83	40.70	45.05	50.58	43.82	45.65	46.33
	Curr	54.06	44.14	46.90	53.14	41.73	46.05	47.74	42.02	43.88	45.61
70/15/15	Mixed	54.49	43.83	46.70	53.61	41.49	45.61	49.46	42.81	44.40	45.57
	Curr	53.40	43.83	46.54	52.20	42.49	45.22	48.75	42.74	44.05	45.27
80/10/10	Mixed	54.21	43.41	46.05	52.83	41.35	45.19	**50.97**	43.65	45.63	45.62
	Curr	50.61	42.71	44.54	49.47	40.13	43.25	49.80	42.99	45.33	44.37
90/5/5	Mixed	53.06	43.48	46.30	52.17	41.02	45.14	50.82	43.70	45.75	45.73
	Curr	48.05	39.25	41.91	46.77	35.33	39.53	50.19	44.64	**46.09**	42.51

Table 3: Full results for all 31 models. ZS = zero-shot average, FT = finetune average, Avg = combined average for that language, Multi = multilingual average across all three languages. Bold indicates the best score in each column.

Looped GPT-BERT: Trading Parameters for Computation in Small Language Modeling

Tingshuo Fan **Hongtao Mu** **Tianyu Zhou** **Hansen Liu** **Tao Ji**✉
College of Foreign Languages and Literature, Fudan University
tsfan24@m.fudan.edu.cn taoji@fudan.edu.cn

Abstract

When training data are limited, increasing parameter count is not the only way to improve language-model performance. A small parameter set, when repeatedly applied, can also deliver comparable performance. We study Looped GPT-BERT in the BabyLM 2026 Strict-small setting, combining GPT-BERT's masked next-token and causal language-modeling objectives with depth-wise parameter sharing. We train on a preprocessed 7.48M-word English corpus and compare objective ratios, non-looped and looped architectures, and loop counts. Our final 4×12 model uses four physical layers for twelve recurrent traversals and contains 12.18M parameters. The BabyLM 2026 leaderboard reports an Overall Average of 35.42 and an NLP Average of 48.48. Compared with public BabyLM 10M Strict-small GPT-2 and GPT-BERT baselines, it achieves comparable performance on selected linguistic and downstream metrics, including BLiMP and GLUE, with fewer parameters. The loop ablations show that additional recurrent computation can improve training and preserve strong performance on selected linguistic tasks, whereas poorer performance on other tasks may reveal an inherent limitation of the looped design: using only a few physical layers restricts the model's representational space.

1 Introduction

Language-model performance has commonly improved through jointly scaling model size, data, and training computation. Scaling-law studies describe predictable power-law relations between loss, parameter count, data, and compute (Kaplan et al., 2020), while compute-optimal analyses emphasize that parameter and token budgets should be allocated jointly (Hoffmann et al., 2022). These results largely concern much larger autoregressive models, where model capacity and per-example computation grow together. Under constrained data and parameter budgets, however, they need not: repeatedly applying the same parameters can create a deeper computational path without proportionally increasing the number of learned weights. We therefore ask whether recurrent computation induced by shared parameters can compensate for reduced physical depth and parameter capacity in low-resource language-model pretraining.

The BabyLM Challenge provides a controlled setting for this question by restricting pretraining text to a developmentally plausible scale and evaluating linguistic, downstream, and human-like behavior (Warstadt et al., 2023). We work in the BabyLM 2026 Strict-small track on a cleaned 7.48M-word English corpus. Our starting point is GPT-BERT (Charpentier and Samuel, 2024), which trains masked next-token prediction (MNTP) and causal language modeling (CLM) in one Transformer parameter stack. It consequently supports both bidirectional masked prediction and autoregressive scoring or generation. This hybrid objective gives a further dimension to study: when the backbone is looped, does the relative amount of masked and causal supervision change the capabilities the architecture acquires?

We introduce depth-wise recurrent parameter sharing on top of GPT-BERT. Universal Transformers repeatedly share transformations along depth (Dehghani et al., 2019), and ALBERT shows that cross-layer sharing can substantially reduce encoder parameters (Lan et al., 2020). Most directly, Looped Transformers inject a fixed input and the prior loop state into a shared backbone, increasing effective computational depth through iteration (Yang et al., 2024). Their strongest evidence, however, comes from in-context algorithm-learning tasks such as linear functions, decision trees, and neural-network regression. Natural-language pretraining must simultaneously acquire lexical knowledge, syntactic regularities, world knowledge, and discourse state; whether those computations can

✉Corresponding author.

Proceedings of the Second BabyLM Workshop, pages 912–921
October 29, 2026 ©2026 Association for Computational Linguistics

be supported by repeated shared layers remains an open empirical question.

We address three questions: how the BERT:GPT training ratio changes the training objective and downstream abilities; whether effective depth formed by repeatedly applying a small number of physical layers can retain language ability with substantially fewer parameters; and whether training and downstream tasks continue to benefit as the number of loops increases. We summarize our contributions as follows:

- We introduce Looped GPT-BERT, combining GPT-BERT's hybrid MNTP/CLM objective with depth-wise recurrent parameter sharing.
- We study how the BERT:GPT objective ratio affects training and downstream behavior in the looped architecture.
- We compare a 12-layer non-looped model with four physical layers applied 1/3/6/12 times, showing how recurrent computation closes the gap from fewer learned layers and where additional loops yield diminishing returns.
- We evaluate the final model in the BabyLM Challenge, where its 12.18M parameters achieve comparable performance to public Strict-small baselines on linguistic and downstream tasks.

2 Related Work

Data-Efficient LM Pretraining BabyLM shifts attention from performance under web-scale training to learning efficiency under a controlled data budget. The first challenge showed that architecture, pretraining objective, preprocessing, and curriculum can all substantially affect small-model performance, with no single approach dominating every downstream task (Warstadt et al., 2023; Hu et al., 2024). Subsequent work further found that the effect of data composition depends on model scale: a genre mixture suitable for one model size need not be optimal for another (Yam and Paek, 2024). Other BabyLM studies explore variation sets, child-inspired data and vocabulary choices, explicit linguistic information, and self-distillation as complementary routes to sample-efficient pretraining (Haga et al., 2024; Ghanizadeh and Dousti, 2024; Edman et al., 2024; Nair et al., 2024). These approaches primarily improve sample efficiency through data selection, ordering, or objectives. Complementarily, we study parameter efficiency: keeping the training corpus and model width fixed, we replace part of the independently parameterized depth with repeated computation through shared physical layers.

Hybrid Causal and Masked Modeling CLM predicts the next token from left context, whereas conventional MLM predicts selected tokens at their own positions from bidirectional context. The two objectives therefore assign different meanings to the same output position in a decoder-style model. GPT-BERT resolves this mismatch with MNTP (Charpentier and Samuel, 2024): when token x_{k+1} is masked, supervision is shifted left so that the hidden state at position k predicts the original x_{k+1}. Consequently, causal and masked examples share a single next-token vocabulary projection and loss interface. GPT-mode rows retain their original tokens and use a lower-triangular attention mask; MNTP-mode rows corrupt selected input tokens, permit bidirectional attention, and compute loss only at the corresponding shifted next-token positions. Each training row is assigned one mode, rather than receiving both losses. AntLM similarly combines causal and masked language-modeling objectives in a BabyLM setting, alternating between them during training (Yu et al., 2024).

Recurrent Depth and Parameter Sharing Universal Transformers update representations recurrently along depth while retaining parallel computation over sequence positions (Dehghani et al., 2019); ALBERT uses cross-layer parameter sharing primarily for parameter efficiency and shows that the sharing strategy affects downstream behavior (Lan et al., 2020). Looped Transformers use the recurrence $z_{t+1} = f(x + z_t)$, where x is a fixed input embedding, z_t is the loop state, and f is a shared Transformer backbone (Yang et al., 2024). This makes effective computational depth grow with loop count while the parameter count is governed mainly by the number of physical layers. Kohli et al. (2026) provide a more direct natural-language reference: recurrent-depth Transformers can improve systematic generalization and depth extrapolation in implicit multi-hop reasoning, but excessive loops can also reduce prediction quality through overthinking. Neither line of work establishes that shared loops are uniformly suitable for lexical, syntactic, world-knowledge, and discourse-state learning. We therefore compare equal-application and equal-parameter configurations across linguistic and state-tracking tasks.

Cleaning operation	Before	After
Style and grammar normalization	*Where are you at?*	*Where are you?*
Disfluency removal	*a pound, or a hundred pounds today, is not the same*	*A hundred pounds today is not worth the same amount later.*
Short-content filtering	*Right.*	Removed when it lacks independent semantic content.
Case normalization	*LET'S GIVE IT UP FOR DANNY TANNER*	*Let's give it up for Danny Tanner.*
Structural-noise removal	*=== Bullet the Blue Sky ===*	Header removed; article text retained.
Stage-direction removal	*I'll be right back. [leaves room.]*	*I'll be right back.*
Speaker normalization	**MOT: more what? *CHI: more tapioca.*	*A: More what? B: More tapioca.*
Consecutive-turn merging	*B: Yeah. B: I'm in Texas.*	*B: Yeah, I'm in Texas.*
Duplicate removal	Repeated or near-identical sentences	One normalized instance.

Table 1: Representative corpus-cleaning operations. Examples are shortened for presentation.

3 Method

3.1 Corpus Preparation and Tokenization

We construct our training corpus by selecting and cleaning six source corpora: BNC Spoken, CHILDES, Project Gutenberg, OpenSubtitles, Simple English Wikipedia, and Switchboard. The resulting `new_data1` corpus is produced by our rule-based normalization and filtering pipeline, which removes redundancy, short or low-quality content, case and structural noise, and normalizes dialogue formatting. The preprocessing code and source files are available in our preprocessing repository. **Table 1** summarizes the operations and representative input–output examples. After cleaning, the six selected corpora yield 7,482,189 whitespace-delimited words. We use this complete post-cleaning corpus directly for pretraining rather than adding or padding data to reach the 10M-word upper bound. Their source-wise counts are reported in **Table 2**.

We use two byte-pair encoding (BPE) tokenizers. The official 16k GPT-BERT tokenizer is used only for early non-looped ratio experiments, while the 8k tokenizer trained on the `new_data1` corpus is used for the main and loop experiments. BPE provides a subword representation that can handle words beyond a fixed vocabulary (Sennrich et al., 2016). We choose the 8k vocabulary as a parameter-allocation trade-off for this small-data setting: with hidden size 384, its tied embedding table contains $8{,}192 \times 384 \approx 3.15$M parameters. In our corpus, reducing the vocabulary from 16k to 8k increases the measured fertility from 1.4381 to 1.4787 tokens per whitespace-delimited word, a 2.82% increase. This indicates a modest change in word-level context coverage for the fixed 128-token input, while avoiding a larger embedding allocation.

Source	Words
BNC Spoken	619,847
CHILDES	1,779,027
Gutenberg	2,502,709
OpenSubtitles	1,174,391
Simple Wikipedia	1,386,065
Switchboard	20,150
Total	7,482,189

Table 2: Word counts after preprocessing.

3.2 Hybrid GPT-BERT Objective

Let $x = (x_1, \ldots, x_T)$ be a token sequence and θ the shared model parameters. For GPT-mode rows, a causal attention mask is used and every non-padding next token is supervised:

$$\mathcal{L}_{\text{GPT}} = -\sum_{t=1}^{T-1} \log p_\theta(x_{t+1} \mid x_{\leq t}). \qquad (1)$$

For BERT-mode rows, predictable positions are sampled with a masking probability that decreases linearly from 0.3 to 0.15. Of the selected tokens, 80% are replaced by the mask token, 10% by a random token, and 10% are left unchanged. Bidirectional attention is permitted. The target is shifted to the preceding hidden-state position so that it remains aligned with the decoder-style next-token head. If $\tilde{x}$ denotes the corrupted input, the objective is

$$\mathcal{L}_{\text{BERT}} = -\sum_{t \in M} \log p_\theta(x_t \mid \tilde{x}_{\setminus t}). \qquad (2)$$

Only shifted positions corresponding to selected masks contribute to this loss; all other labels are set to the ignore index. Thus, unmasked BERT positions never enter the loss. A batch-level ratio of $r_{\text{B}} : r_{\text{G}}$ assigns complete rows to the two modes, and the optimized objective averages valid supervised tokens with a z-loss regularizer weighted by

10^{-4}. Our final setting uses 1:3, so one quarter of rows use MNTP and three quarters use CLM.

3.3 Looped Transformer Backbone

The non-looped baseline contains twelve independent Transformer layers. The looped backbone instead contains four independently parameterized physical layers, each with attention and feed-forward sublayers. Each loop adds the static token representation x to the preceding loop state z_t and applies the same four-layer stack F_θ:

$$\begin{aligned} z_0 &= 0, \\ z_{t+1} &= F_\theta(x + z_t), \quad t = 0, \ldots, L-1. \end{aligned} \tag{3}$$

Thus, z_{t+1} is the final representation produced by the current traversal and is simply carried into the next pass; it is not an additional network operation. Within physical layer i, attention and feed-forward transformations are applied sequentially:

$$\begin{aligned} u_t^{(i)} &= \mathrm{DWA}_{i,A}\left(h_t^{(i)} + \mathrm{Attn}_i(h_t^{(i)})\right), \\ h_t^{(i+1)} &= \mathrm{DWA}_{i,F}\left(u_t^{(i)} + \mathrm{FFN}_i(u_t^{(i)})\right). \end{aligned} \tag{4}$$

GPT-BERT's dynamic weighted accumulation (DWA) is a learnable short-range residual mixer, not an additional recurrent state. We restrict DWA to representation fusion within each physical layer and reinitialize it on every loop; information across loops is carried exclusively by z_t. A $4 \times L$ model therefore has four sets of Transformer parameters but performs $4L$ layer applications. The 4×3, 4×6, and 4×12 variants perform 12, 24, and 48 layer applications, respectively, while keeping the same parameter count. This "effective depth" counts layer applications only: repeatedly applying shared parameters is not equivalent to a non-shared Transformer with the same number of layers.

3.4 Training Configuration

Table 3 lists the main configuration. Beyond the structural hyperparameters shown there, optimization uses LAMB (You et al., 2020), a peak learning rate of 0.0141, a minimum learning rate of 0.00141, and cosine decay for 2,600 steps followed by a constant minimum rate. We select the peak learning rate and decay horizon through the controlled studies in Section 5. All pretraining and fine-tuning runs use random seed 42.

Hyperparameter	Value
Physical layers	4
Loop iterations	12
Hidden size	384
FFN size	1,280
Attention heads	6
Vocabulary	8,192
Maximum sequence length	128
Batch size per GPU	32
Number of GPUs	8
Global batch (tokens)	32,768
Epochs	10
Optimizer	LAMB
Peak learning rate	0.0141
Decay steps	2,600
Minimum learning rate	0.00141
Gradient clipping	2.0
Parameters	12.18M

Table 3: Main pretraining configuration.

4 Experiments

4.1 Evaluation Protocol

We follow the BabyLM 2026 Strict-small evaluation pipeline. The final checkpoint receives the full evaluation: the zero-shot suite includes BLiMP and BLiMP Supplement minimal-pair syntax judgments (Warstadt et al., 2020), EWoK world knowledge (Ivanova et al., 2024), Entity Tracking, COMPS conceptual properties, GlobalPIQA commonsense reasoning, Reading, and Age of Acquisition (AoA). Fine-tuning covers BoolQ, MNLI, MRPC, MultiRC, QQP, RTE, and WSC, the GLUE/SuperGLUE-style natural-language-understanding subset used by the challenge (Wang et al., 2019). We report the leaderboard aggregates, where NLP Average pools NLP tasks and Human-like Average includes Reading, AoA, and related measures. We use the official causal backend for all reported GPT-BERT results.

Beyond the final *main* revision, we retain intermediate checkpoints at one-million-word intervals from *chck_1M* through *chck_10M*, and at ten-million-word intervals from *chck_20M* through *chck_70M*. We run the official fast evaluation on each of these checkpoints. The ten-epoch training run contains approximately 74.8M word presentations, so *chck_70M* is the last complete ten-million-word milestone. The final model and all intermediate revisions are released in one public Hugging Face repository.

4.2 BabyLM 2026 Leaderboard Results

Table 4 reports the final 4×12 Looped GPT-BERT alongside two publicly released BabyLM Strict-small references: the official GPT-2 baseline and

Model	Params	Context	Overall	NLP	BLiMP	Sup.	EWoK	Entity	COMPS	GPIQA	GLUE
Official GPT-2 baseline	124M	512	37.38	48.99	65.23	57.25	50.63	19.10	51.81	35.09	63.80
Official GPT-BERT causal-focus	31M	128→512	28.46	35.64	71.66	63.21	49.49	–	–	–	65.13
Looped GPT-BERT (ours)	**12.18M**	**128**	35.42	48.48	71.19	55.06	49.10	15.78	51.01	34.67	62.55

Table 4: BabyLM 2026 Strict-small leaderboard comparison on aggregate and selected task-level metrics. "Context" denotes the maximum training sequence length, "Sup." denotes BLiMP Supplement, "Entity" denotes Entity Tracking, "GPIQA" denotes GlobalPIQA, and "GLUE" denotes the official (Super)GLUE aggregate. Dashes indicate results not reported for the corresponding public entry. Aggregate scores for entries with unreported tasks are not directly comparable.

the GPT-BERT causal-focus baseline. The leaderboard snapshot was accessed on July 21, 2026.[1]

With 12.18M parameters, the final model is substantially smaller than both public references. It reaches 71.19 on BLiMP, compared with 65.23 for GPT-2 and 71.66 for GPT-BERT causal-focus, while its GLUE aggregate is 62.55, compared with 63.80 and 65.13. This pattern illustrates the trade-off studied here: at the cost of additional computation, repeated application of shared layers can retain strong performance on selected tasks with a smaller independently parameterized model, with gains varying across tasks.

4.3 Effect of the BERT:GPT Ratio

We examine the BERT:GPT ratio under two tokenizers. The official 16k tokenizer is used for early non-looped ratio exploration, whereas the 8k tokenizer trained on the `new_data1` corpus is used for all main and loop experiments. The 15:1 configuration is BERT-heavy, 1:1 is balanced, and 1:3 is GPT-heavy. The two settings test whether the ratio trend is consistent; they are not used to compare tokenizer quality. **Tables 5 and 6** report the principal 8k comparison, including training endpoints and available zero-shot results.

BERT:GPT	Params	Loss	Acc.
1:1	29.9M	3.9925	30.03
1:3	29.9M	**3.3625**	**36.47**

Table 5: Non-looped training results across objective ratios with the custom 8k tokenizer.

For context, **Table 7** reports the corresponding early experiments with the official 16k tokenizer. Increasing the proportion of GPT rows lowers the final mixed training loss in both tokenizer settings, with 1:3 giving the strongest endpoint.

[1] https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026

BERT:GPT	BLiMP	Sup.	COMPS	Entity
1:1	63.77	55.06	50.87	**38.66**
1:3	**70.54**	**57.19**	**51.41**	22.87

Table 6: Non-looped zero-shot results across objective ratios with the custom 8k tokenizer.

BERT:GPT	Final loss	Token accuracy
15:1	4.0039	33.42
1:1	3.9711	31.50
1:3	**3.7925**	32.37

Table 7: Non-looped training results across objective ratios with the official 16k tokenizer.

The 1:3 ratio improves BLiMP, BLiMP Supplement, and COMPS under the 8k tokenizer, but reduces Entity Tracking from 38.66 to 22.87. This pattern suggests that objective mixing changes the distribution of learned capabilities rather than merely the aggregate loss. More CLM supervision repeatedly trains left-to-right next-token prediction and is therefore closely aligned with sequential generation, local dependencies, and causal syntactic scoring. More MNTP supervision requires reconstructing a target from both sides of its context and may better support integrating multiple positions and relations, which is useful for tracking entities and their states. We therefore use 1:3 as the main configuration for subsequent loop experiments targeting generative and syntactic ability.

4.4 Effect of Looped Depth

We first examine training loss and token accuracy under the 1:3 objective. **Figure 1** plots how these two measures change over training for the 12-layer non-looped model and for four-layer models with 1, 3, 6, or 12 applications per forward pass. The 4×3 and 12-layer non-looped models both execute twelve layer applications per forward pass, while

BERT:GPT = 1:3					
Metric	Non-loop	4×1	4×3	4×6	4×12
BLiMP	70.54	65.08	**71.21**	70.49	71.19
BLiMP Sup.	**57.19**	55.90	55.73	55.37	55.06
COMPS	51.41	51.21	51.40	**51.69**	51.01
Entity Tracking	22.87	17.38	16.47	**28.66**	15.78
Eye Tracking	8.61	**8.62**	7.85	**8.62**	8.25
Self-paced Reading	4.03	3.88	3.68	4.16	**4.24**
BERT:GPT = 1:1					
Metric	Non-loop	4×1	4×3	4×6	4×12
BLiMP	63.77	62.28	64.45	**69.38**	64.21
BLiMP Sup.	55.06	**57.03**	54.96	56.11	56.07
COMPS	50.87	50.44	50.95	**51.42**	50.03
Entity Tracking	**38.66**	17.52	16.60	13.10	17.15
Eye Tracking	7.86	7.64	**7.88**	7.62	7.55
Self-paced Reading	3.80	**3.84**	3.57	3.56	3.63

Table 8: Zero-shot comparison across objective ratios and loop counts.

the looped model uses only four independently parameterized layers.

The 4×1 control is visibly less stable at the beginning of training: it maintains a higher loss and lower token accuracy than the models with repeated layer applications. Increasing the loop count lowers the training loss and generally raises token accuracy, bringing the looped models closer to the 12-layer non-looped trajectory. The largest improvement occurs when moving from 4×1 to 4×3; the changes from 4×3 to 4×6 and from 4×6 to 4×12 are smaller. Thus, repeated computation recovers much of the training behavior of the non-looped model, while additional loops provide diminishing returns.

We next turn to the zero-shot task results in **Table 8**. The table compares the 12-layer non-looped model with four physical layers applied 1, 3, 6, or 12 times, using BERT:GPT ratios of 1:3 in the upper block and 1:1 in the lower block. Under 1:3, increasing the loop count from 4×1 to 4×3 substantially improves BLiMP, from 65.08 to 71.21, slightly exceeding the non-looped score of 70.54. Increasing the depth beyond 4×3 does not produce a consistent additional gain: BLiMP is 70.49 for 4×6 and 71.19 for 4×12. Under 1:1, 4×6 gives the highest BLiMP score (69.38), compared with 63.77 for the non-looped model and 62.28–64.45 for the other looped settings. Overall, the looped models preserve or improve several syntactic and reading-related scores despite using fewer independently parameterized layers, while the effect of recurrent depth varies across tasks.

Unlike the syntactic measures above, Entity Tracking is more sensitive to parameter sharing. Overall, the looped variants are weaker than the corresponding non-looped model on this task, with one exception: under 1:3, 4×6 reaches 28.66, compared with 22.87 for the non-looped model. Even in this setting, the score falls to 15.78 with 4×12, so the additional computation from six to twelve applications does not preserve the 4×6 improvement. Under 1:1, all looped variants remain below the non-looped score of 38.66, with scores between 13.10 and 17.52. One possible explanation is that entity tracking requires distinguishing and updating multiple entities, their states, and state changes (Kim and Schuster, 2023). Reusing four attention/FFN parameter sets rather than twelve independent sets may leave less representational space for the layer-specific transformations needed to maintain multiple entities and relations. This contrast suggests that recurrent depth can compensate for reduced parameterization on some linguistic tasks, but cannot uniformly replace the representational flexibility of independently parameterized layers. Considering both downstream performance and inference cost, 4×6 is a practical intermediate configuration, but the table also shows that the preferred loop depth depends on the objective ratio and task.

4.5 Inference Cost

We measure inference cost for the 12-layer non-looped model and the 4×3, 4×6, and 4×12 looped models with the same batch size of 32, sequence length of 128, 20 warmup steps, and 100 timed forward passes on one RTX 3090. As shown in **Table 9**, 4×3 is better than the 12-layer non-looped model on every measured systems metric: it has lower latency, higher token throughput, and lower peak memory use. Increasing the loop count from 4×3 to 4×6 and 4×12 roughly doubles the latency at each step, while throughput decreases and memory use increases. Considering both downstream performance and inference speed, 4×6 provides a practical compromise: it is slower than 4×3 but performs better on some downstream tasks, while remaining substantially faster than 4×12.

5 Optimization Ablations

5.1 Selecting the Peak Learning Rate

We first select a peak learning rate that lowers the training objective quickly without producing clear instability. Before full training, we run four-epoch non-looped sweeps at 0.0075, 0.01, 0.0141, and 0.02 with all other settings fixed. As **Ta-**

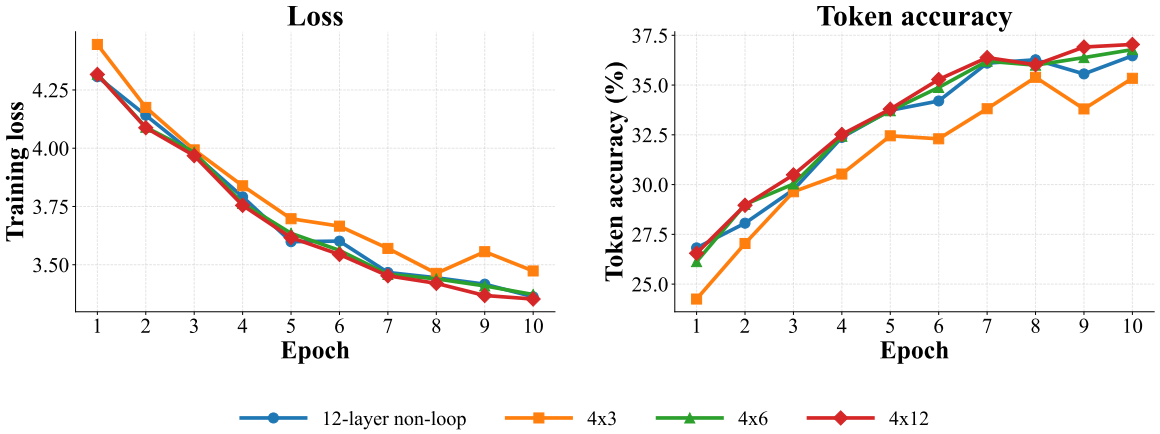


Figure 1: Training loss and token accuracy for the 8k-tokenizer, 1:3 objective across loop depths. The looped models use four physical layers; the non-looped model uses twelve independently parameterized layers.

Model	Latency (ms)	Tokens/s	Memory (GiB)
12-layer non-loop	58	70,571	0.68
4×3	**55**	**74,128**	**0.51**
4×6	108	37,966	0.72
4×12	212	19,295	1.15

Table 9: Inference speed measured with batch size 32, sequence length 128, 20 warmup steps, and 100 timed forward passes on one RTX 3090.

ble 10 shows, the first three settings decrease stably, whereas 0.02 degrades after approximately step 600 and produces gradient-norm spikes. Although 0.01 has the highest final token accuracy, 0.0141 achieves the lowest final loss (3.6847) and is selected as the peak learning rate for the main experiments. This sweep provides a preliminary learning-rate choice based on the non-looped model and does not fully optimize the rate for looped models. We examine the effect of decay speed on training stability separately below.

Peak LR	Final loss	Final accuracy
0.0075	3.7442	32.66
0.0100	3.6931	**33.07**
0.0141	**3.6847**	32.95
0.0200	4.5864	22.84

Table 10: Four-epoch peak learning-rate sweep.

5.2 Learning-Rate Decay and Stability

We select two candidate decay lengths from the training budget: 4,080 steps spans the full optimizer-step budget of the run, whereas 2,600 steps reaches the minimum earlier and remains there for the rest of training. With total training steps fixed, *lr_schedule_steps* determines how quickly cosine decay reaches its minimum: a smaller value decays earlier and remains at the minimum longer, while a larger value retains a higher learning rate later in training. We compare these decay rates in 4×6 and 4×12 models. The main 2,600-step schedule reaches 10% of the peak learning rate and then holds 0.00141; the slower 4,080-step schedule maintains a higher learning rate into the later stages. Because shared physical layers are repeatedly applied across loops, the effect of one parameter update can be propagated through multiple layer applications. A high early learning rate may therefore amplify update oscillations, while earlier decay may help shared parameters enter a useful region more smoothly.

Figure 2 shows that the 2,600-step schedule has its clearest advantage in early and middle training: loss declines faster and token accuracy rises sooner. Earlier decay, however, can reduce the opportunity to explore alternative parameter regions and leave the model near a suboptimal solution. The 4,080-step schedule gradually catches up later, consistent with its higher late-stage learning rate preserving additional exploration. Nevertheless, the 2,600-step curves are smoother overall and show no clear final-metric disadvantage. We therefore choose 2,600 steps as a practical compromise between stability and late-stage exploration under the current compute budget and single-seed setting, not as a schedule that is uniformly superior at every point in training.

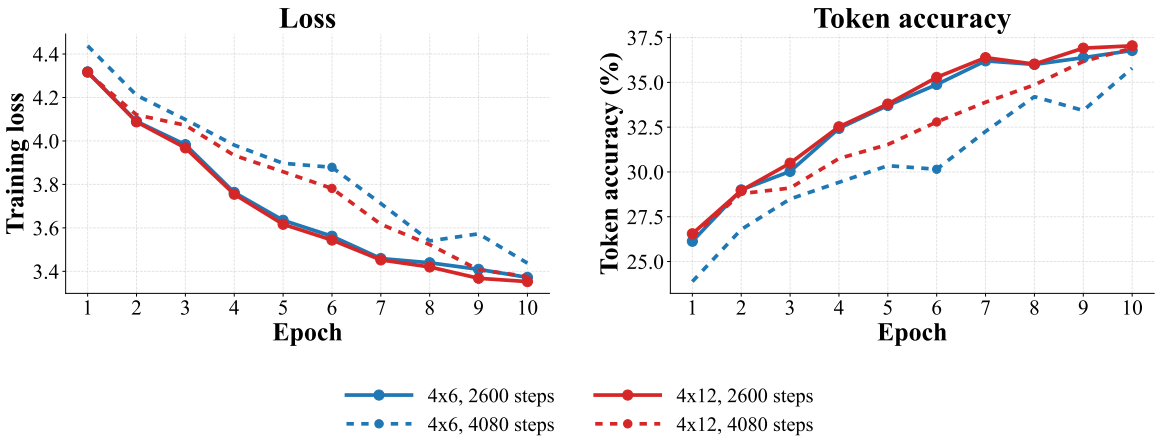


Figure 2: Training loss and token accuracy for the learning-rate decay ablation. Solid and dashed lines denote 2,600-step and 4,080-step decay, respectively; colors distinguish 4×6 and 4×12.

5.3 Qualitative Generation Examples

The causal interface also supports open-ended completion. **Table 11** presents selected completions from the final 1:3, 4×12 model using temperature 0.9, top-k 40, top-p 0.95, and seed 42. The examples illustrate locally coherent continuation and appropriate termination in short responses; they are qualitative examples rather than a controlled generation evaluation.

Prompt	Sampled completion
I want to eat some	*dinner, all right?*
After the rain stopped, the children	*began to run for a walk.*
She put the toy in the	*drawer, and gave the doll a glass of milk.*
The bird flew over the	*mountains and in the morning the rain came in.*

Table 11: Selected sampled completions from the final 1:3, 4×12 model.

6 Conclusion

We combine GPT-BERT's hybrid CLM/MNTP training with depth-wise recurrent parameter sharing to study a "fewer parameters, more computation" language-model design under BabyLM Strict-small constraints. Reusing four physical layers for twelve loops yields a 12.18M-parameter model whose training objective is close to that of a 29.9M non-looped model. The final model reaches 71.19 on BLiMP and 62.55 on GLUE, compared with 71.66 and 65.13 for the official GPT-BERT causal-focus reference, despite its smaller parameter budget. On the reported task-level metrics, looped models preserve or improve selected syntactic abilities, while Entity Tracking becomes weaker as a result of parameter sharing. The ablations further show that gains from additional loops saturate quickly and differ across tasks. Looped GPT-BERT is therefore an architecture with clear advantages and limitations: under constrained data and parameter budgets, it can support syntactic understanding and generation, but its objective ratio and recurrent depth require further adjustment.

Limitations

- **Data and tokenizer.** The main experiments use one cleaned corpus and one 8k tokenizer. We do not independently ablate raw data, rule-based cleaning, alternative cleaning procedures, and tokenization.
- **Training variance.** Each model is trained with a single random seed, so sub-percentage-point differences may fall within training variance.
- **Compute matching.** The 4×12 model performs substantially more layer applications and computation than the 12-layer non-looped baseline. This is a parameter-efficiency study, not a FLOP-matched comparison.
- **Objective mixing.** Changing the BERT:GPT ratio also changes supervision density and attention visibility; mixed training losses are therefore not perfectly homogeneous across ratios.
- **Capability boundary.** The final model is weak on AoA and Entity Tracking, indicating limited cognitive-similarity and state-tracking ability.
- **Scope.** Our conclusions are limited to approximately 12M parameters, a 7.48M-word corpus, ten epochs, and a maximum sequence length of

128; they should not be directly extrapolated to larger models or longer contexts.

Ethics Statement

The model uses English text supplied by, or derived from, the BabyLM Challenge and introduces no newly collected personal data. The sources include books, subtitles, encyclopedic text, and dialogue, so they may retain social biases or inappropriate content from the original material; automated cleaning cannot guarantee their removal. Cleaning may also remove dialectal, conversational, or minority-language patterns and introduce stylistic preferences through normalization. This small model is intended for research and should not be deployed directly in high-stakes applications.

Reproducibility Statement

The Hugging Face repository contains the final model and sixteen intermediate checkpoints. The *main* revision stores the final model; *chck_1M–chck_10M* are one-million-word checkpoints, and *chck_20M–chck_70M* are ten-million-word checkpoints. It also includes *configuration_gpt_bert.py*, *modeling_gpt_bert.py*, and tokenizer files, allowing the model to be loaded with *trust_remote_code=True*.[2] Training configurations and scripts are available in the accompanying code repository,[3] and the data-cleaning workflow is documented separately.[4]

Acknowledgments

The authors thank the reviewers for their helpful comments and suggestions, as well as the BabyLM organizers for maintaining the datasets, evaluation pipeline, and leaderboard. This work was partially funded by the National Natural Science Foundation of China (No. 62506079).

References

Lucas Georges Gabriel Charpentier and David Samuel. 2024. GPT or BERT: Why not both? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283, Miami, FL, USA. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

Lukas Edman, Lisa Bylinina, Faeze Ghorbanpour, and Alexander Fraser. 2024. Are BabyLMs second language learners? In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 166–173, Miami, FL, USA. Association for Computational Linguistics.

Mohammad Amin Ghanizadeh and Mohammad Javad Dousti. 2024. Towards data-efficient language models: A child-inspired approach to language learning. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 22–27, Miami, FL, USA. Association for Computational Linguistics.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM challenge: Exploring the effect of variation sets on language model training efficiency. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 252–261, Miami, FL, USA. Association for Computational Linguistics.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030.

Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Leshem Choshen, Ryan Cotterell, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21, Miami, FL, USA. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, et al. 2024. Elements of world knowledge (EWoK): A cognition-inspired framework for evaluating basic world knowledge in language models. *arXiv preprint arXiv:2405.09605*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages

[2] https://huggingface.co/Survivor613/BabyLM2026-Strict-Small-looped_GPT-BERT

[3] https://github.com/Survivor613/BabyLM-2026

[4] https://github.com/JT-Ushio/babylm26-nlp-spring

3835–3855, Toronto, Canada. Association for Computational Linguistics.

Harsh Kohli, Srinivasan Parthasarathy, Huan Sun, and Yuekun Yao. 2026. Loop, think, & generalize: Implicit reasoning in recurrent-depth transformers. *arXiv preprint arXiv:2604.07822*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Aakarsh Nair, Alina Hancharova, Mayank Kumar, and Ali Gharaee. 2024. BabyLM challenge: Experimenting with self-distillation and reverse-distillation for language model pre-training on constrained datasets. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 28–36, Miami, FL, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Hong Meng Yam and Nathan Paek. 2024. What should baby models read? exploring sample-efficient data composition on model performance. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 284–291, Miami, FL, USA. Association for Computational Linguistics.

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. 2024. Looped transformers are better at learning learning algorithms. In *International Conference on Learning Representations*.

Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations*.

Xinru Yu, Bin Guo, Shiwei Luo, Jie Wang, Tao Ji, and Yuanbin Wu. 2024. AntLM: Bridging causal and masked language models. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 324–331, Miami, FL, USA. Association for Computational Linguistics.

Author Index

Akbik, Alan, 75, 299, 722
Aksoy, Ömer Faruk, 15
Alvarez-Melis, David, 750

Badawy, Mohamed, 199
Barak, Libby, 279
Beauchemin, David, 369
Beinborn, Lisa, 209, 805
Belinkov, Yonatan, 750
Benedikt, Martin, 631
Bohn, Jeremias, 843
Brantley, Kianté, 750
Brasoveanu, Adrian, 260
Bylinina, Lisa, 165

Cacioli, Jon-Paul, 240
Cai, Alexander Dazhen, 750
Cai, Qing, 779
Can, Burcu, 227
Chavales, Georgios, 413
Chen, Yijing, 309
Chen, Yubei, 823
Chesi, Cristiano, 659
Chiang, Po-Han, 563
Choshen, Leshem, 479
Cotterell, Ryan, 479
Cygan, Hanna B., 659

Dalar, Yağız Ekrem, 15
Dev, Atul, 767
Dewan, Divija, 767
Ding, Chenyi, 250
Doggett, Erika Varis, 702
Dostál, Ivan, 631
Dotlacil, Jakub, 260
Duong, Lam, 93
Dusek, Ondrej, 678

Edman, Lukas, 598, 611, 624, 639, 904

Fahim, Tamjid Hasan, 859
Fan, Tingshuo, 912
Fang, Andrew, 93
Feiertag, Sebastian, 843
Feng, Gefei, 831
Filandrianos, Giorgos, 320
Finegan-Dollak, Catherine, 388
Formichi, Guido, 659
Fraser, Alexander, 199, 598, 611, 624, 639, 904
Furr, Sam, 514
Fusco, Achille, 659

Gaines, Dylan, 689
Goda, Koki, 735
Golde, Jonas, 75, 722
Groh, Georg, 843
Gul, Mustafa Omer, 479
Gülbahar, Serdar, 639

Haberer, Janek, 33
Haider, Mozafar, 227
Haller, Patrick, 75, 722
Han, Phu, 93
He, Danni, 779
Hojjat, Ali, 33
Hotta, Daichi, 543
Houghton, Zachary Nicholas, 250
Huang, Zhuojing, 805
Huda, Imrul, 179
Hwang, Haerim, 50
Hwang, JiWoo, 904

Islam, Sadri, 689
Iwasawa, Yusuke, 543

Jain, Soham, 767
Jasbi, Masoud, 250
Ji, Tao, 309, 912
Jumelet, Jaap, 479

Kamzela, Wiktor, 678
Ke, Alan Hezao, 426
Kemme, Oscar, 33
Kennedy, Mary, 702
Kim, Heesu, 105
Klubarski, Robin, 631
Kobashi, Yohei, 543
Kojima, Takeshi, 543
Komachi, Mamoru, 105
Kordjamshidi, Parisa, 426
Kosireddy, Tagore Rao, 689
Ku, Bill K. S., 148
Kyo, Seiji, 735

Laitenberger, Filipe, 75, 722
Landsiedel, Olaf, 33

Lango, Mateusz, 678
Lanz, Vojtěch, 575
Lautenschlager, Jonathan, 209
Li, Kevin, 689
Li, Zy, 93
Liao, Hanyu, 309
Lien, Hsuan, 598
Linzen, Tal, 479
Liu, Hansen, 912
Liu, Jianghao, 779
Liu, Yihan, 403
Lymperaiou, Maria, 320

Maheshwari, Yash, 1
Marco, Marion Di, 199
Martins, Jonas Mayer, 209
Matsuo, Yutaka, 543
Melkersson, Valdemar, 504
Meyer, Francois, 585
Mikat, Aftab Hossain, 859
Miles, Sebastian, 525
Mishra, Ujjwal, 646
Mita, Masato, 187
Mori, Naoki, 735
Mu, Hongtao, 912
Mueller, Aaron, 479
Murillo, Jorge C, 823
Mustafa, Noor Eidirhn, 504
Müller-Eberstein, Max, 187

Nugues, Pierre, 403, 413, 504

Oseki, Yohei, 187
Oytaç, Doğa, 358
Öztürk, Ahmet Rıfat, 15

Palomino, Alonso, 493
Paudel, Prashanna Mani, 134
Perdijk, Sabijn, 459
Pierrehumbert, Janet B., 148
Ploner, Max, 299
Pohlmann, Luise, 805
Pu, Shihao, 831
Pulipaka, Venkata Naga Sai Vishnu Rohit, 279

Raube, Nicolas, 843
Ren, Pengfei, 65
Rodriguez, Juan Diego, 875
Rooryck, Dries, 750
Rossi, Sarah, 659
Roy, Sudipta, 646

Salhan, Suchir, 479
Salihoğlu, Feyzi Arda, 15
Sasidhar, Aditya, 123
Satheesh, Siddhi, 823
Sato, Shota, 105
Schwenger, Emil, 843
Sezer, Nedim Mutlu, 15
Sgrizzi, Tommaso, 659
Shah, Raj Sanjay, 479
Sheshappanavar, Shivanand Venkanna, 134
Singh, Harsh, 767
Song, Ming, 779
Srivastav, Arnav, 289
Szymanik, Jakub, 575
Štěpánková, Barbora, 575

Takmaz, Ece, 260
Taylor, Josh, 24
Theodoropoulos, Nikitas, 320
Tran, Trinh, 93

Vaira, Edoardo, 413
van Duijn, Max J., 459
Varghese, Dheeraj, 333
Vazquez Martinez, Hector Javier, 438
Veitsman, Yana, 209
Verberne, Suzan, 459

Wang, Yitong, 779
Wang, Yupei, 892
Warstadt, Alex, 479
Wasserman, Adam Zachary, 369
Wei, Zichao, 539
Wijnholds, Gijs, 459
Wilcox, Ethan Gotlieb, 479
Wu, Shaoxiang, 611

Xia, Ellie, 426
Xu, Yizhe, 779

Yamaguchi, Misaki, 187
Yang, Charles, 438
Yang, Fuming, 598
Yang, Kai-hao, 598
Yang, Yiming, 831
Yang, Yuwei, 831

Zanollo, Asya, 659
Zhang, Tongjing, 403
Zhang, Wen, 348

Zhou, Tianyu, 912