

Recursive Transformers for BabyLM 2026

Serdar Gülbahar¹

Alexander Fraser^{1,2,3}

Lukas Edman^{1,2}

¹School of Computation, Information and Technology, TU Munich

²Munich Center for Machine Learning

³Munich 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,

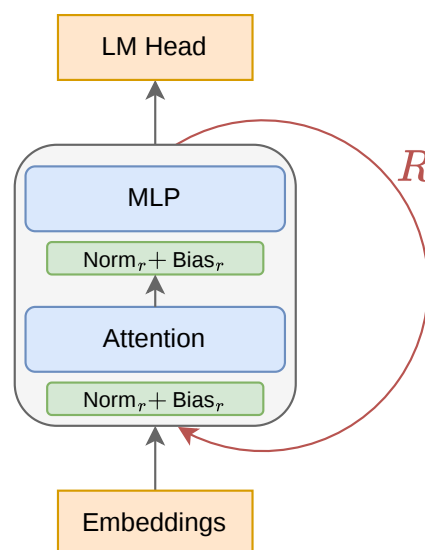


Figure 1: Recursive transformer architecture, adapted from Gülbahar et al. (2026). One attention/MLP block is reused for R recurrent steps, with step-specific normalization and bias; embeddings and the LM head remain outside the recursion.

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

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.,

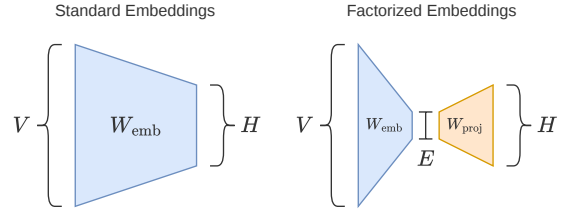


Figure 2: Standard and factorized vocabulary maps, adapted from Gülbahar et al. (2026). The factorized version uses embedding dimension E as an intermediate size rather than mapping the vocabulary directly to hidden dimension H .

Configuration	RecGPT-10M	RecGPT-100M
Recursive depth R	16	24
Hidden size H	768	1,408
Embedding size E	192	768
Heads $\times$ head dim.	12×64	22×64
MLP expansion	$16 \times$	$16 \times$
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.

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.

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 Pierce, 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

$$\hat{h}_{t+1} = \text{RMSNorm}(h_t + \Delta_{\phi}(h_t, \text{sg}(e_{t+1}))),$$

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.