

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 pre-training 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.

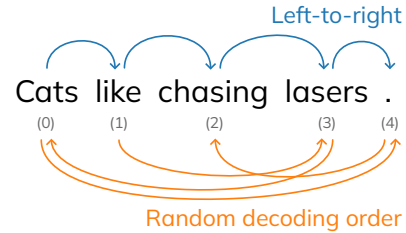


Figure 1: Left-to-right decoding (often implemented via next-token prediction) illustrated by the arrows on top. Random decoding order illustrated by the arrows on the bottom: While the decoding order is random, the tokens’ positional embedding indices (in parentheses) stay the same.

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

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 $\mathbf{x} = (x_1, \dots, x_T)$ as a forward product (Radford et al., 2018; Yang et al., 2019):

$$p(\mathbf{x}) = \prod_{t=1}^T p(x_t | x_{<t})$$

While this may be the most intuitive factorization, $p(\mathbf{x})$ 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:

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \in \mathcal{Z}_T} \left[\sum_{t=1}^T \log p_{\theta}(x_{z_t} | x_{z_{<t}}) \right]$$

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

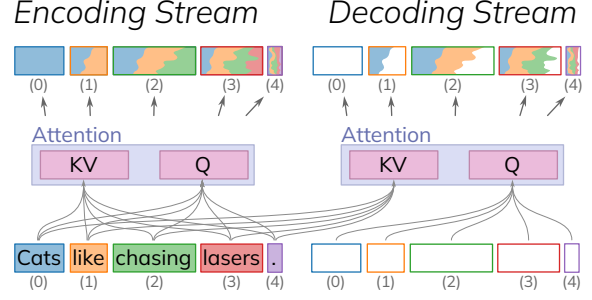


Figure 2: Dual stream attention (here with left-to-right decoding order): The encoding stream initializes the residual stream with the current tokens’ embedding, and the decoding stream initializes the residual stream with zero. This way, the encoding stream aggregates information about the current token *and* the context, while the decoding stream aggregates information *only* about the context to predict the current token.

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.

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.

¹The 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} 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%).

	Epochs	2	5	7	10	12	15	20
Model	Ratio							
one-stream	-	64.52 $\pm$ 0.16	66.90 $\pm$ 0.56	65.80 $\pm$ 0.32	63.96 $\pm$ 0.57	63.11 $\pm$ 0.21	62.39 $\pm$ 0.08	61.93 $\pm$ 0.74
two-stream	0.00	65.08 $\pm$ 0.77	66.56 $\pm$ 0.53	65.89 $\pm$ 0.51	63.82 $\pm$ 0.20	63.15 $\pm$ 0.72	61.14 $\pm$ 1.08	62.94 $\pm$ 0.65
	0.10	64.93 $\pm$ 1.51	66.33 $\pm$ 1.78	66.44 $\pm$ 1.36	63.55 $\pm$ 0.93	63.20 $\pm$ 0.50	62.67 $\pm$ 1.13	60.90 $\pm$ 0.55
	0.25	64.30 $\pm$ 0.93	67.25 $\pm$ 0.95	67.16 $\pm$ 0.59	65.85 $\pm$ 0.95	64.28 $\pm$ 0.26	62.91 $\pm$ 0.70	61.95 $\pm$ 0.78
	0.50	63.54 $\pm$ 0.79	65.69 $\pm$ 0.47	66.33 $\pm$ 0.69	65.24 $\pm$ 0.77	64.67 $\pm$ 0.46	64.39 $\pm$ 0.62	62.27 $\pm$ 0.17
	0.75	62.98 $\pm$ 0.60	63.79 $\pm$ 0.57	65.05 $\pm$ 1.28	65.22 $\pm$ 0.70	65.61 $\pm$ 0.28	64.93 $\pm$ 0.78	64.30 $\pm$ 1.36
	0.90	61.46 $\pm$ 0.90	63.32 $\pm$ 0.85	63.73 $\pm$ 1.23	63.46 $\pm$ 1.45	64.28 $\pm$ 0.49	63.49 $\pm$ 0.87	63.32 $\pm$ 0.40
	1.00	60.14 $\pm$ 1.47	60.30 $\pm$ 1.40	60.58 $\pm$ 0.84	60.92 $\pm$ 0.62	61.03 $\pm$ 1.21	61.30 $\pm$ 2.38	61.01 $\pm$ 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

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).

(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.

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 SuperGIUE (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 ± 0.59	58.88 ± 0.90	51.96 ± 0.46	19.41 ± 0.56	51.77 ± 0.44
	two-stream (25%)	68.40 ± 0.27	58.00 ± 1.19	52.15 ± 0.08	18.15 ± 0.60	51.76 ± 0.74
	two-stream (75%)	66.36 ± 0.42	55.25 ± 1.43	52.92 ± 0.18	17.87 ± 0.19	52.19 ± 0.40
small	one-stream	68.69 ± 0.44	60.64 ± 1.04	52.19 ± 0.50	19.69 ± 0.46	51.49 ± 0.59
	two-stream (25%)	69.57 ± 0.57	58.37 ± 0.68	52.51 ± 0.62	18.13 ± 0.63	51.72 ± 1.21
	two-stream (75%)	66.68 ± 0.30	55.29 ± 1.79	52.95 ± 0.11	19.33 ± 0.96	52.29 ± 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

	Task	BoolQ	CoNLL-2003	MNLI	MRPC	MultiRC	QQP	Avg.
	Metric	Acc.	F_1	Acc.	F_1	F_1	F_1	-
Size	Model							
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%)	<u>69.34</u> $\pm$ 0.43	<u>82.82</u> $\pm$ 0.17	<u>48.18</u> $\pm$ 0.67	<u>82.57</u> $\pm$ 0.99	<u>49.15</u> $\pm$ 7.44	66.25 $\pm$ 0.59	<u>66.38</u>
	two-stream (75%)	70.30 $\pm$ 0.46	83.69 $\pm$ 0.21	48.97 $\pm$ 0.50	82.58 $\pm$ 0.13	<u>51.03</u> $\pm$ 1.64	<u>65.07</u> $\pm$ 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	<u>50.41</u> $\pm$ 0.99	62.15 $\pm$ 0.25	64.75
	two-stream (25%)	<u>69.75</u> $\pm$ 0.37	<u>83.24</u> $\pm$ 0.34	<u>48.51</u> $\pm$ 0.54	<u>82.51</u> $\pm$ 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	<u>48.09</u> $\pm$ 2.54	<u>65.00</u> $\pm$ 1.44	<u>66.47</u>

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 variant³, 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 MP-Net, 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>

⁴Allen-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 \(EWOKE\): 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 fatori, Charles Lovering, farzanehnakhaee70, Jason Phang, Anish Thite, Fazz, Aflah, Niklas Muenighoff, Thomas Wang, sdtbck, nopperl, gakada, tttyuntian, researcher2, Julen Etxaniz, Chris, Han-wool 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 $\pm$ 0.05	52.64 $\pm$ 0.10	52.52 $\pm$ 0.02	52.47 $\pm$ 0.16	52.28 $\pm$ 0.13	52.15 $\pm$ 0.09	52.39 $\pm$ 0.06
0.10	53.72 $\pm$ 0.19	54.07 $\pm$ 0.30	54.12 $\pm$ 0.34	53.67 $\pm$ 0.30	53.43 $\pm$ 0.01	52.72 $\pm$ 0.37	52.40 $\pm$ 0.12
0.25	54.26 $\pm$ 0.03	55.09 $\pm$ 0.09	55.27 $\pm$ 0.33	54.78 $\pm$ 0.07	54.71 $\pm$ 0.23	54.38 $\pm$ 0.14	53.84 $\pm$ 0.50
0.50	54.79 $\pm$ 0.04	55.77 $\pm$ 0.23	56.07 $\pm$ 0.13	55.96 $\pm$ 0.22	55.82 $\pm$ 0.21	55.69 $\pm$ 0.33	54.95 $\pm$ 0.26
0.75	55.22 $\pm$ 0.13	56.23 $\pm$ 0.26	56.61 $\pm$ 0.10	56.79 $\pm$ 0.15	56.75 $\pm$ 0.43	56.64 $\pm$ 0.25	56.34 $\pm$ 0.37
0.90	55.13 $\pm$ 0.13	56.52 $\pm$ 0.19	56.82 $\pm$ 0.16	56.83 $\pm$ 0.25	57.17 $\pm$ 0.40	56.75 $\pm$ 0.35	56.68 $\pm$ 0.29
1.00	55.13 $\pm$ 0.25	56.17 $\pm$ 0.33	56.41 $\pm$ 0.35	56.74 $\pm$ 0.11	56.30 $\pm$ 0.27	56.63 $\pm$ 0.28	56.43 $\pm$ 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 $\pm$ 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
		bi. (last)	67.48 $\pm$ 0.53	-	47.72 $\pm$ 0.63	80.28 $\pm$ 1.66	48.56 $\pm$ 1.75	63.45 $\pm$ 0.41	61.50
		uni.	68.42 $\pm$ 0.93	65.74 $\pm$ 0.23	51.44 $\pm$ 1.23	81.87 $\pm$ 1.02	50.90 $\pm$ 1.25	67.55 $\pm$ 0.53	64.32
	two-stream (25%)	bi.	69.34 $\pm$ 0.43	<u>82.82$\pm$0.17</u>	48.18 $\pm$ 0.67	<u>82.57$\pm$0.99</u>	49.15 $\pm$ 7.44	<u>66.25$\pm$0.59</u>	<u>66.38</u>
		bi. (last)	68.79 $\pm$ 0.25	-	49.71 $\pm$ 0.48	81.06 $\pm$ 1.03	50.60 $\pm$ 2.28	65.86 $\pm$ 0.51	63.20
		uni.	66.18 $\pm$ 0.66	65.17 $\pm$ 0.48	46.84 $\pm$ 0.48	78.87 $\pm$ 1.29	44.98 $\pm$ 6.40	60.50 $\pm$ 0.42	60.42
	two-stream (75%)	bi.	70.30 $\pm$ 0.46	83.69 $\pm$ 0.21	48.97 $\pm$ 0.50	82.58 $\pm$ 0.13	<u>51.03$\pm$1.64</u>	65.07 $\pm$ 0.99	66.94
		bi. (last)	68.97 $\pm$ 0.82	-	<u>50.28$\pm$0.33</u>	82.00 $\pm$ 0.69	46.80 $\pm$ 4.92	65.39 $\pm$ 0.60	62.69
		uni.	65.12 $\pm$ 0.89	65.81 $\pm$ 0.11	42.43 $\pm$ 0.99	78.53 $\pm$ 1.52	41.76 $\pm$ 0.73	54.43 $\pm$ 0.24	58.01
small	one-stream	bi.	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
		bi. (last)	67.87 $\pm$ 0.09	-	47.24 $\pm$ 1.26	82.29 $\pm$ 1.79	52.24 $\pm$ 1.08	64.87 $\pm$ 0.44	62.90
		uni.	68.44 $\pm$ 0.80	64.96 $\pm$ 0.64	50.67 $\pm$ 0.93	82.00 $\pm$ 1.96	49.55 $\pm$ 2.42	68.64 $\pm$ 0.21	64.04
	two-stream (25%)	bi.	69.75 $\pm$ 0.37	<u>83.24$\pm$0.34</u>	48.51 $\pm$ 0.54	<u>82.51$\pm$0.76</u>	<u>50.72$\pm$4.11</u>	65.84 $\pm$ 1.33	66.76
		bi. (last)	68.91 $\pm$ 0.67	-	49.47 $\pm$ 0.36	81.67 $\pm$ 0.54	49.48 $\pm$ 1.08	<u>66.25$\pm$0.47</u>	63.15
		uni.	66.54 $\pm$ 0.16	65.38 $\pm$ 0.38	47.10 $\pm$ 0.36	79.09 $\pm$ 0.93	43.68 $\pm$ 0.71	60.90 $\pm$ 0.41	60.45
	two-stream (75%)	bi.	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	<u>66.47</u>
		bi. (last)	68.93 $\pm$ 0.83	-	<u>50.20$\pm$0.22</u>	80.54 $\pm$ 0.07	49.36 $\pm$ 1.68	66.14 $\pm$ 0.33	63.04
		uni.	66.91 $\pm$ 0.56	65.55 $\pm$ 0.04	41.80 $\pm$ 0.96	81.08 $\pm$ 0.27	29.18 $\pm$ 9.31	55.71 $\pm$ 0.60	56.71

Table 8: Performance of the models after fine-tuning on some GLUE tasks and CoNLL-2003.

Size	Model	Task Metric Variant	Acc.	BestQ F_1	MCC	CoNLL-2003 F_1	MNLI Acc.	MRPC F_1	MCC	Acc.	MultiRC F_1	MCC	Acc.	QQP F_1	MCC	Acc.	RTE F_1	MCC	Acc.	WSC F_1	MCC
unipau	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	62.48±0.04	38.87±0.72	53.86 ±4.35	17.22±3.26	67.31 ±0.00	43.49±11.19	27.92 ±2.07
		bi, (lst)	67.88±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	63.49±0.41	41.29±0.53	58.51±3.69	43.89±8.94	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	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
		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	<u>66.25±0.59</u>	<u>44.75±0.63</u>	54.44±0.83	18.78±32.53	63.46±1.92	14.13±12.82	12.01±11.03
		two-stream (25%)	bi, (lst)	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
	two-stream (75%)	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	52.59±4.01	50.15±5.04	64.74±5.55	43.02±10.73	20.88±13.66
		bi	70.30 ±0.46	79.19 ±0.34	31.14 ±1.17	83.09 ±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	65.38±3.33	32.20±28.06	18.33±15.89
		bi, (lst)	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	66.2±6.53	64.10±1.11	23.50±10.58
		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	63.46±3.85	50.15 ±16.30	23.99±14.66
	llmus	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	55.80 ±1.50	44.83±8.29	63.46±1.92	18.86±24.88	13.67±12.17
		bi, (lst)	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±5.20	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±3.59	49.09±3.59	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	62.18±1.11	39.80±7.08	15.32±3.24
		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	63.84±1.33	44.54±1.74	54.68±2.59	20.13±34.87	66.67 ±2.94	40.50 ±28.40	29.30 ±10.62
		two-stream (25%)	bi, (lst)	68.91±0.67	77.70±0.89	28.55±1.54	-	49.47±0.36	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	45.59±0.38	55.64±0.42	54.03 ±2.08	11.44 ±0.87	62.18±1.11	8.00±13.86
small	two-stream (25%)	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	63.7±4.69	57.69±3.33	15.30±7.29
		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
	two-stream (75%)	bi, (lst)	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	43.75±0.56	55.16±0.42	22.44±24.01	9.35±4.00	60.26±1.11	12.42±14.65
		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	58.97±2.22	29.39±12.18	4.89±8.41

Table 9: Complete fine-tuning results.