Sample-Efficient Language Modeling with Linear Attention and Lightweight Enhancements

Patrick Haller **Jonas Golde** **Alan Akbik**

Humboldt-Universität zu Berlin
{patrick.haller.1, jonas.max.golde, alan.akbik}@hu-berlin.de

Abstract

We study architectural and optimization techniques for sample-efficient language modeling under the constraints of the BabyLM 2025 shared task. Our model, **BLaLM**, replaces self-attention with a linear-time mLSTM token mixer and explores lightweight enhancements, including short convolutions, sliding window attention with dynamic modulation, and Hedgehog feature maps. To support training in low-resource settings, we curate a high-quality corpus emphasizing readability and pedagogical structure. Experiments across both STRICT and STRICT-SMALL tracks show that (1) linear attention combined with sliding window attention consistently improves zero-shot performance, and (2) the Muon optimizer stabilizes convergence and reduces perplexity over AdamW. These results highlight effective strategies for efficient language modeling without relying on scale.

1 Introduction

Training language models under strict resource constraints remains a central challenge, both for advancing theoretical understanding and for enabling practical deployment on limited hardware. The BabyLM shared task provides a unique opportunity to evaluate models in a controlled setting, where participants are restricted to training on at most 10 million (STRICT-SMALL) or 100 million (STRICT) words for a maximum of 10 epochs. This environment encourages the development of sample-efficient algorithms rather than scale-dependent strategies.

Our submission focuses on algorithmic enhancements rather than introducing novel architectures. Specifically, we examine whether recent advancements in model design and optimization can be adapted to improve sample efficiency when applied to a standard Transformer backbone. Our contributions are as follows:

1. **Model Architecture:** We replace the self-attention mechanism in a standard Transformer with the linear-time mLSTM module, yielding an efficient subquadratic variant we refer to as BLaLM.

2. **Optimization:** We evaluate the *Muon* optimizer, a recently proposed alternative to AdamW, which introduces dynamic momentum and a decoupled weight decay schedule. We compare Muon and AdamW under identical training conditions.

3. **Architectural Enhancements:** We introduce and evaluate several lightweight modifications to the BLaLM model, including sliding window attention (SWA), short convolutional layers, and dynamic attention modulation.

4. **Corpus Construction:** We curate a high-quality corpus by filtering and modifying existing text corpora, aiming to improve training dynamics for small models. Preliminary results indicate improved downstream performance relative to unfiltered datasets.

Our experiments lead to two key findings: First, replacing self-attention with a linear-time mLSTM token mixer, especially when combined with sliding window attention and dynamic modulation, leads to strong zero-shot performance under low-resource constraints. Second, the Muon optimizer improves convergence and stability compared to AdamW, particularly for matrix-shaped parameters. Together, these results point to practical strategies for improving sample efficiency in compact language models.

2 Preliminaries and Related Work

Transformers

The Transformer architecture, proposed by Vaswani et al. (2017), has become the

Proceedings of the First BabyLM Workshop, pages 175–191

de facto standard for large-scale language modeling. Unlike recurrent neural networks (RNNs) or long short-term memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997), Transformers process sequential input in parallel through self-attention. Given query, key, and value matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n \times d}$, the self-attention output is computed as:

$$\boldsymbol{y} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}} \odot \boldsymbol{M}\right)\boldsymbol{V}, \qquad (1)$$

where $\mathbf{M}$ is a causal mask that prevents attending to future tokens. While highly expressive, self-attention incurs $\mathcal{O}(n^2 d)$ complexity in both computation and memory, which becomes a bottleneck for long sequences, especially during autoregressive decoding.

Linear Attention

To address the quadratic bottleneck, Katharopoulos et al. (2020) proposed linear attention mechanisms that replace the softmax kernel with a feature map $\phi(\cdot)$ such that:

$$\text{softmax}(\mathbf{Q}\mathbf{K}^\top) \approx \phi(\mathbf{Q})\phi(\mathbf{K})^\top. \qquad (2)$$

This formulation enables autoregressive decoding in $\mathcal{O}(nd^2)$ time by exploiting the associativity of matrix multiplication, reducing memory usage and improving scalability.

Linear attention has since been extended in numerous architectures targeting long-context modeling and efficient training (Sun et al., 2023; Poli et al., 2023). In the BabyLM 2024 shared task, Haller et al. (2024) introduced *BabyHGRN*, which leverages a recurrent HGRN2 token mixer within Transformer-style blocks. It achieved competitive results under low-resource constraints, motivating continued exploration of subquadratic alternatives.

xLSTM and mLSTM

xLSTM (Beck et al., 2024) revisits the LSTM architecture with two core innovations: exponential gating and enhanced memory structures. It defines two cells, sLSTM and mLSTM, which are assembled into residual blocks.

mLSTM extends the scalar memory c_t to a matrix memory $\mathbf{C}_t \in \mathbb{R}^{d \times d}$ that stores key-value pairs via an outer-product update. The forget gate f_t acts as a decay, while the input gate i_t controls the learning rate:

$$\mathbf{C}_t = f_t \mathbf{C}_{t-1} + i_t v_t k_t^\top, \quad n_t = f_t n_{t-1} + i_t k_t, \quad (3)$$

and retrieval is computed using:

$$h_t = o_t \odot \frac{\mathbf{C}_t q_t}{\max\{|\langle n_t, q_t \rangle|, 1\}}, \qquad (4)$$

with q_t, k_t, v_t derived from learned projections.

As with other linear-time mechanisms, mLSTM supports parallel training and linear-time autoregressive decoding. It serves as the token mixer in our model architecture.

The BabyLM Benchmark

The BabyLM initiative (Charpentier et al., 2025) introduced a suite of benchmarks for evaluating language models in low-resource conditions, with a focus on learnability, generalization, and alignment with developmental stages. The 2025 shared task continues this focus, imposing strict limits on training data and epochs to emphasize sample efficiency and high quality data curation.

Optimizers

Adaptive optimizers such as Adam (Kingma and Ba, 2017) and its decoupled variant AdamW (Loshchilov and Hutter, 2019) remain standard for LLM training due to their robustness and ease of tuning. However, their dynamics can be suboptimal for matrix-shaped parameters, especially in low-data or large-batch regimes.

Recent alternatives aim to improve convergence and stability, including Lion (Chen et al., 2023), Sophia (Liu et al., 2024), and Shampoo (Gupta et al., 2018). Muon (Keller, 2024) orthogonalizes gradient updates via a truncated Newton-Schulz iteration, improving conditioning for matrix-valued parameters with minimal overhead. It is typically used in hybrid schemes, where scalar parameters (e.g., layer norms, biases) are still optimized with AdamW.

Muon has shown benefits in both vision and language domains (AI et al., 2025; Liu et al., 2025), including better training stability, faster convergence, and improved data efficiency, which all are valuable under the constraints of BabyLM.

3 Data Curation

Data quality plays a critical role in small-scale language modeling, where noisy or incoherent samples can substantially degrade performance. In this work, we prioritize readability, coherence, and syntactic simplicity to improve learnability under low-resource constraints.

Dataset	# Words STRICT-SMALL	# Words STRICT
CHILDES Project (Child-directed speech)	2M	8.7M
Fineweb-Edu	2M	21M
TinyStories	1M	35M
Project Gutenberg, Fiction Books	1.5M	1.7M
Simple Wikipedia (English)	1.5M	22.6M
Cosmopedia		
- WikiHow	1.8M	10.1M
- Math	0.2M	0.3M
Total	$\approx$ 10M	$\approx$ 99.5M

Table 1: Token counts per data source in the curated corpus used for the STRICT-SMALL and STRICT tracks of BabyLM 2025.

Rather than relying solely on large, unfiltered corpora, we curate a dataset by filtering and modifying existing sources using heuristic and LLM-guided approaches. Our filtering pipeline targets syntactically clean, semantically rich, and pedagogically structured documents likely to be learnable by small models.

3.1 Data Sources

Our curated pretraining corpus draws from a diverse set of publicly available datasets selected for their relevance to early language acquisition, general knowledge, and structured instruction. The largest component is **FineWeb-Edu** (Lozhkov et al., 2024; Penedo et al., 2025), a filtered subset of FineWeb-2 annotated for educational value. To incorporate spoken language patterns, we include transcripts from the **CHILDES** corpus (MacWhinney, 2000), which features child-directed speech. We also leverage **TinyStories** (Eldan and Li, 2023), a synthetic story dataset designed for early learners. Fictional content is sourced from a filtered selection of English novels from **Project Gutenberg** (Gerlach and Font-Clos, 2020), while simplified encyclopedic entries come from **Simple Wikipedia**. Finally, we include domain-specific educational content from **Cosmopedia** (Ben Allal et al., 2024), which covers instructional materials such as WikiHow articles and mathematics explanations. A breakdown of word counts per dataset and track is provided in Table 1.

3.2 Filtering Pipeline

We apply dataset-specific filters to improve linguistic quality and reduce noise. Below we summarize our main filtering strategies:

FineWeb-Edu Although FineWeb-Edu is already annotated for educational value, we re-evaluate all samples using our own educational scoring prompt (Appendix A) with LLaMA 3.3–70B.

Gutenberg Fiction We discard Gutenberg entries without named entities and subsample up to 200 samples per book to ensure diversity.

TinyStories We remove template-like introductions (e.g., "Once upon a time…") to reduce repetition and increase stylistic variety.

Simple Wikipedia We retain only paragraphs with at least 15 words to remove boilerplate and fragmented content.

Cosmopedia (WikiHow & Math) We filter for content relevant to K–12 learners and remove excessively long or domain-specific passages.

CHILDES (Child-Directed Speech) CHILDES contains transcripts of parent–child dialogue, marked with speaker tags (e.g., "*MOT:" for mother). We apply:

1. **Speaker Tag Removal:** Prefixes like "*MOT:" or "*COL:" are removed.
2. **Minimum Length Filtering:** We discard utterances with fewer than 7 words.
3. **Grammar Correction:** We normalize speech using LanguageTool for improved grammaticality.

All data is tokenized and counted at the word level to ensure the final corpus respects the BabyLM 2025 limits of 10M (STRICT-SMALL) and 100M (STRICT) words.

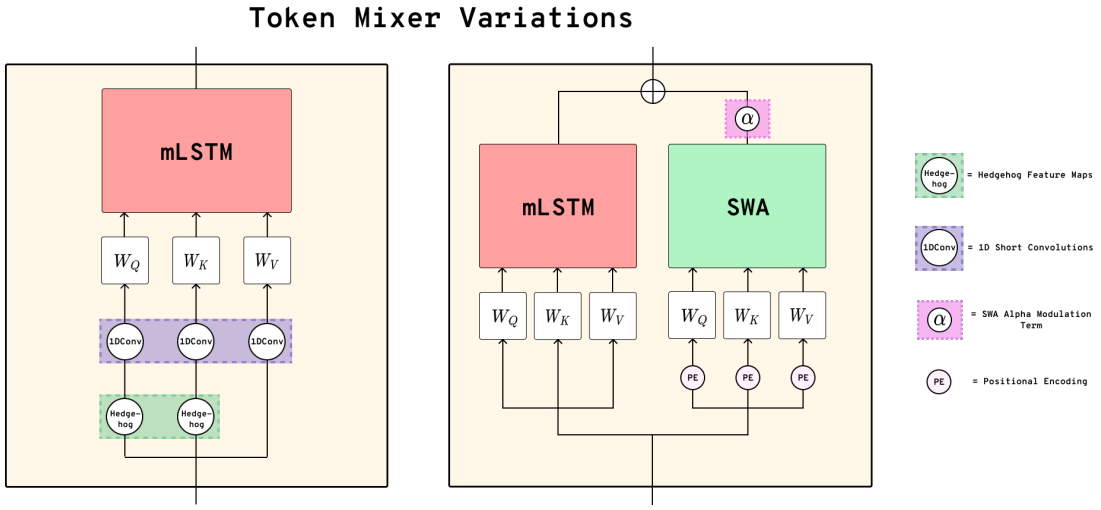

Figure 1: Overview of the BLaLM architecture. The standard self-attention module is replaced by an mLSTM token mixer. Optional enhancements such as sliding window attention (SWA) can be integrated and combined with mLSTM outputs.

4 Model Architecture and Optimization

We aim to evaluate whether architectural and optimization strategies known to improve large-scale language models can also improve sample efficiency under strict training budgets. Rather than designing a novel architecture, we incrementally modify a standard Transformer decoder to assess the contribution of individual components.

Our model, referred to as **BLaLM** (**B**aby **L**inear **A**ttention **LM**), follows the general architecture of recent Qwen models (Bai et al., 2023). It uses **pre-normalization** with **RMSNorm** (Zhang and Sennrich, 2019) for training stability, **feed-forward blocks** with SwiGLU activations, and **rotary positional embeddings (RoPE)** (Su et al., 2023) to encode position information. RoPE is used in both the self-attention baseline and the optional sliding window attention modules in BLaLM.

The key deviation from the standard Transformer lies in the **token mixer**, which is the module responsible for integrating contextual information across tokens. In Transformers, this role is fulfilled by the self-attention mechanism; in BLaLM, we replace it with **mLSTM**, a recurrent linear-time alternative. The mLSTM operates via element-wise gating (forget and input gates) and uses matrix-valued memory updates across learned projections. It supports fully parallel training and linear-time autoregressive decoding, thereby avoiding the quadratic overhead of softmax attention while maintaining expressivity.

This architectural choice preserves full compatibility with Transformer training pipelines, allowing direct comparisons between self-attention and mLSTM-based token mixing.

4.1 Architectural Enhancements

In addition to the mLSTM substitution, we introduce a set of lightweight architectural improvements aimed at enhancing sample efficiency:

- **Short Convolutions (ShortConv):** 1D depthwise convolutions are added before the token mixer on the query and key projections to enhance local inductive bias. Recently added by Gu and Dao (2024); Dao and Gu (2024); Beck et al. (2024); Lan et al. (2025); Nguyen et al. (2025).

- **Sliding Window Attention (SWA)** (Beltagy et al., 2020): A local attention mechanism with fixed-size attention window. Sliding is used in conjunction with the mLSTM token mixer. The input is passed through both modules and added together, like:

$$h_{final} = \frac{h_{LA}}{2} + \frac{h_{SWA}}{2} \tag{5}$$

- **SWA with Dynamic Modulation (DynMod):** Applies a learned gating function to modulate attention hidden states over each layer.

$$h_{total} = h_{LA} + \alpha \cdot h_{SWA} \tag{6}$$

$$h_{total} = h_{LA} + tanh(\alpha) \cdot h_{SWA} \tag{7}$$

Dataset	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
Baseline-10M	64.96	66.8	40.01	51.55	0.98	0.45	37.45
Baseline-100M	75.68	65.2	34.82	51.82	0.82	0.33	**38.11**
Our-10M	67.99	63.6	39.46	52.27	1.09	0.65	37.51
Our-100M	74.51	57.6	14.65	55.73	1.18	0.90	34.09

Table 2: Comparison of the official BabyLM dataset ("Baseline") and our curated corpus ("Ours") across both strict-small and strict tracks. We report average zero-shot performance; full results in Appendix C.

- **Hedgehog Feature Maps** (Zhang et al., 2024): A recently proposed mechanism that mimics several properties of softmax-based attention. It is applied to the query and key projections.

Each mechanism, as illustrated in Figure 1, is introduced independently and evaluated against the base BLaLM configuration to quantify its contribution under fixed training budgets.

5 Training Setup

All experiments are conducted under the BabyLM 2025 shared task constraints for the STRICT-SMALL (10M words) and STRICT (100M words) tracks, using the curated dataset described in Section 3

Sequence Length and Batching

We train with a context length of 512 tokens and an effective global batch size of 64. When hardware limitations require smaller per-device batches, we use gradient accumulation to match the target batch size. Text data is first concatenated into a continuous stream before splitting into fixed-length sequences to avoid truncation and minimize padding overhead.

Models

We use two architectural variants throughout our experiments: a baseline Transformer decoder (Qwen-style) and our proposed model, BLaLM, which replaces self-attention with an mLSTM token mixer. Both models share the same configuration where applicable; architectural differences are detailed in Appendix B.

Training Duration and Checkpointing

Each model is trained for a maximum of 10 epochs over the respective corpus. We evaluate all saved checkpoints and report results for the best-performing one based on average downstream performance.

Evaluation

All models, except the final submissions, are evaluated using the fast zero-shot evaluation suite provided by the BabyLM organizers (Charpentier et al., 2025).[1] We rely on this fast evaluation method to score all intermediate checkpoints and select the best model per run. Final submissions are evaluated on hidden tasks and additionally fine-tuned on GLUE.[2]

Learning Rate Scheduling

We use a cosine decay schedule with a 10% linear warmup phase. The learning rate used for each experiment is reported in the corresponding results section.

Optimizers

We use either AdamW or Muon, as introduced in Section 2. In Sections 6.1 and 6.2, AdamW is used as the default optimizer. Later experiments switch to Muon, which is applied to matrix-shaped parameters (e.g., projection weights, MLP layers), while AdamW handles all scalar-valued parameters (e.g., embeddings, biases, and normalization layers).

6 Experiments

6.1 Experiment 1: Dataset Performance

This experiment evaluates the impact of our curated dataset relative to the baseline corpus provided by the BabyLM organizers. Since the original training configuration of the baseline models could not

[1] Shortly before the deadline, a bug was discovered in the evaluation for the *WuG* task. Due to time constraints, we were unable to re-evaluate all models. We therefore exclude this task from our reported results.

[2] For a complete list of benchmarks and descriptions, see the official BabyLM 2025 evaluation pipeline.

Track	Model	LR	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
Strict-Small	Transformer (9)	4e-4	64.95	57.2	18.07	51.36	1.13	0.93	32.27
	BLaLM (9)	5e-4	66.72	55.6	40.93	51.0	0.91	0.60	35.96
Strict	Transformer (9)	4e-4	72.44	62.0	20.63	53.36	1.02	0.74	35.03
	BLaLM (10)	5e-4	74.49	60.4	21.99	53.91	1.03	0.71	35.42

Table 3: Zero-shot performance comparison between Transformer and BLaLM across both BabyLM tracks. Results reflect the best-performing epoch per model in brackets after the model name. See Appendix D for full details.

be fully replicated, particularly in terms of preprocessing, we train our own baseline models using their corpus under our experimental setup for a fair comparison.

Setup We use our proposed architecture (BLaLM) and train two variants on each dataset, the BabyLM-provided corpus and our curated corpus, for both the STRICT-SMALL and STRICT tracks. Each configuration is run twice with identical hyperparameters to control for variance. To keep the comparison controlled, we fix the learning rate at 4×10^{-4} for all runs.

Results Table 2 shows that in the STRICT-SMALL setting, our dataset yields slightly higher average scores (37.51 vs. 37.45), with improvements observed in BLiMP, EWoK, and Entity accuracy. In the STRICT track, the performance gap reverses, the baseline corpus outperforms ours, particularly on BLiMP Supplement and Entity.

These results suggest that dataset quality plays a stronger role in low-resource settings, where clean, coherent input provides better learning signals for small models. While the curated data does not consistently outperform the baseline at larger scales, it performs on par, and slightly better in the strict-small regime, without requiring additional sources or augmentation.

Because this dataset was specifically optimized for educational quality, readability, and structure, we use it for all subsequent experiments.

6.2 Experiment 2: Transformers vs. Linear Attention

This experiment assesses the effect of replacing the standard self-attention mechanism in a Transformer with an mLSTM-based token mixer.

Setup We compare two architectures: a baseline Transformer decoder (following the Qwen configuration) and our proposed model, BLaLM. Both models share the same configuration where applicable, differing only in the token mixer. Due to small differences in parameterization between self-attention and mLSTM, the number of layers is adjusted to keep parameter counts approximately matched. Full architectural details are provided in Appendix B.

Experiments are conducted for both the STRICT-SMALL and STRICT tracks. For each architecture, we train models using three learning rates (3e-4, 4e-4, 5e-4) to account for differences in convergence dynamics.

Results Table 3 presents the evaluation results. In the STRICT-SMALL setting, BLaLM consistently outperforms the Transformer baseline across all learning rates, with the best configuration (5e-4) improving the average score from 32.27 to 35.96.

In the STRICT track, results are more balanced. While the Transformer baseline performs better at some learning rates, BLaLM achieves the highest overall score (35.42 compared to 35.03) showing that the benefits of linear attention persist even in the presence of more data, albeit with smaller margins.

These results support the hypothesis that linear-time alternatives like mLSTM can improve sample efficiency in the low-data regime and remain competitive at larger scales, making them a viable drop-in replacement for self-attention in resource-constrained training scenarios.

6.3 Experiment 3: The Choice of Optimizer

This experiment compares two optimizers for pretraining BLaLM: AdamW, the default choice for Transformer training, and Muon, a recently proposed optimizer designed to improve convergence speed and numerical conditioning for matrix-valued parameters.

Setup AdamW is applied to all parameters, while Muon is used in a hybrid scheme as described

in Section 5. Specifically, Muon updates matrix-shaped parameters such as projections and MLP weights, while scalar-valued parameters (e.g., biases, embeddings, normalization layers) are handled by AdamW.

Experiments are conducted in the STRICT track using a fixed learning rate of 4e-4. Each optimizer is evaluated across three independent runs to account for variability in training and initialization. Performance is measured both in terms of validation perplexity and average zero-shot score.

OPTIMIZER	PPL	AVG.
AdamW	11.21 ±0.11	35.75 ±1.74
Muon	7.95 ±0.15	36.24 ±1.16

Table 4: Validation perplexity and average zero-shot scores across three runs comparing AdamW and Muon optimizers for xLSTM training.

Results Table 4 summarizes the results. Muon achieves a lower average validation perplexity (7.95 ± 0.15) compared to AdamW (11.21 ± 0.11), suggesting more stable and efficient optimization.

Zero-shot performance is slightly higher for Muon (36.24 ± 1.16) than for AdamW (35.75 ± 1.74), although the gap is modest. Notably, Muon exhibits more consistent results across runs, indicating improved training stability.

Overall, these findings suggest that Muon improves convergence and may lead to marginal downstream gains under strict resource constraints. Based on these observations, we use Muon for all subsequent experiments.

6.4 Experiment 4: Learning Rate Sweep

This experiment aims to identify the optimal learning rate for pretraining BLaLM under the BabyLM constraints for both the STRICT-SMALL and STRICT tracks.

Setup We conduct a sweep over learning rates in the range from 2e-4 to 7e-4. Each configuration is trained using the same setup described in Section 5, with Muon as the optimizer and a training budget of 10 epochs.

After the initial sweep, we include one additional intermediate learning rate for each track, selected based on observed trends in the initial results. All models are evaluated based on validation perplexity and average zero-shot score. Full results are provided in Appendix F.

	STRICT-SMALL		STRICT	
LEARNING RATE	PPL.	AVG.	PPL.	AVG.
2e-4	16.41	34.80	9.83	34.28
3e-4	16.41	35.61	8.46	35.82
4e-4	20.01	37.27	8.06	35.08
5e-4	16.41	35.03	7.74	35.82
6e-4	16.61	34.17	7.76	35.06
7e-4	**15.73**	**37.53**	7.64	36.10
Additional Learning Rates				
7.5e-4	14.84	37.04	-	-
5.5e-4	-	-	7.70	**37.49**

Table 5: Results from a learning rate sweep for BLaLM on both tracks. Additional intermediate rates were selected based on observed trends.

Results Table 5 reports evaluation results for all tested learning rates. In the STRICT-SMALL track, the highest average score is achieved at 7e-4 (37.53), while 4e-4 and 5e-4 also perform competitively. A follow-up experiment with 7.5e-4 yields slightly lower performance (37.04), suggesting diminishing returns beyond 7e-4.

In the STRICT track, performance peaks at 5.5e-4 with an average score of 37.49. This outperforms 5e-4 and 7e-4, suggesting 5.5e-4 offers the best trade-off.

Overall, the results highlight that optimal learning rates differ by data scale. In low-resource regimes, higher learning rates such as 7e-4 are beneficial, while in higher-resource settings, more moderate values around 5.5e-4 provide the best trade-off between stability and generalization.

6.5 Experiment 5: Evaluating Lightweight Architectural Enhancements

In this experiment, we augment the base BLaLM architecture with a range of lightweight mechanisms that have shown promise in recent work on efficient sequence modeling. These additions are designed to improve local processing, inductive bias, and compositional mixing.

Setup All experiments are conducted in both the STRICT-SMALL and STRICT tracks using the same training setup as in previous sections. The learning rate is fixed at 4e-4, and the Muon optimizer is used for all runs.

Each enhancement is introduced independently to isolate its effect on performance. In addition, a subset of combinations is also evaluated to test potential synergies between modules. Results are reported in terms of validation perplexity and average zero-shot score across the BabyLM benchmark

Mechanism	strict-small PPL.	strict-small Avg.	strict PPL.	strict Avg.
BLaLM	20.01	**37.27**	7.95	35.08
- *ShortConv*	12.37	36.41	6.48	34.57
- *SWA*	12.08	36.16	7.38	35.86
- *SWA with Memory*	10.08	34.96	6.67	37.21
- *SWA DynMod*	9.44	36.15	7.76	**38.82**
- *SWA DynMod Bounded*	8.58	34.41	6.84	36.21
- *Hedgehog*	6.18	33.58	6.68	36.65
- *Hedgehog + SWA*	7.27	36.25	6.63	34.20

Table 6: Evaluation of lightweight architectural enhancements added to BLaLM. Each mechanism is tested independently on both BabyLM tracks. Results include validation perplexity and average zero-shot performance.

suite.

Results Table 6 presents the results. In the STRICT-SMALL track, most mechanisms improve over the base model, with ShortConv and SWA variants performing particularly well. Hedgehog yields the lowest perplexity (6.18), suggesting improved optimization efficiency, although this does not translate directly into the highest downstream score.

In the STRICT track, the most effective mechanism is SWA combined with dynamic modulation, which reaches the highest average score of 38.82. Hedgehog and bounded DynMod also improve performance relative to the base configuration.

We additionally tracked the learned weights α for SWA in the dynamic modulation setups. As shown in Appendix G, these weights vary across layers and increase over training time, suggesting that deeper layers rely more heavily on local context mixing.

Overall, these results indicate that augmenting mLSTM with lightweight attention or modulation mechanisms can improve both perplexity and downstream performance, particularly when local structure and compositional control are emphasized.

6.6 Final Submission Models

For our final BabyLM 2025 submissions, we select configurations that balance strong downstream performance with stable optimization, as identified in our preceding experiments.

STRICT-SMALL Track (10M words): We use BLaLM with mLSTM token mixing, augmented with short convolutions. The learning rate is set to 7e-4, and optimization uses Muon for matrix-shaped parameters and AdamW for scalars. This configuration yields robust zero-shot accuracy across linguistic and educational benchmarks while maintaining low perplexity.

STRICT Track (100M words): We adopt the same architecture, but with a learning rate of 5.5e-4, which in our sweep showed superior generalization in higher-data regimes. We include SWA with bounded dynamic modulation, avoiding further additions to preserve architectural simplicity.

We denote the models BLaLM-STRICT-SMALL and BLaLM-STRICT respectively.

In both tracks, models are trained for 10 epochs using the curated dataset described in Section 3. Final submissions are fine-tuned on GLUE for hidden test set evaluation, as per shared task protocol.

The results are shown in Table 7.

Model	Zero-Shot Avg.	Fine-Tune Avg.
BLaLM-STRICT-SMALL	29.54	57.35
BLaLM-STRICT	36.49	56.70

Table 7: Final performance of our submitted models (BLaLM-STRICT-SMALL and BLaLM-STRICT) on the full BabyLM and (Super)GLUE benchmark suites. Results are averaged across all tasks.

7 Conclusion

We introduced **BLaLM**, a sample-efficient language model built with linear attention and lightweight enhancements. Across both strict and strict-small tracks, BLaLM outperforms Transformer baselines in low-resource settings and remains competitive at larger scales. Our results highlight two actionable insights: (1) combining mLSTM with sliding window attention and dynamic modulation consistently improves downstream generalization, and (2) the Muon optimizer stabilizes training and reduces perplexity, outperforming AdamW for matrix-valued parameters. These findings offer concrete guidance for efficient model design in data-constrained environments.

References

Essential AI, :, Ishaan Shah, Anthony M. Polloreno, Karl Stratos, Philip Monk, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, Anil Thomas, Ashish Tanwer, Darsh J Shah, Khoi Nguyen, Kurt Smith, Michael Callahan, Michael Pust, Mohit Parmar, Peter Rushton, Platon Mazarakis, Ritvik Kapila, Saurabh Srivastava, Somanshu Singla, Tim Romanski, Yash

Vanjani, and Ashish Vaswani. 2025. Practical efficiency of muon for pretraining.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report.

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.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Loubna Ben Allal, Anton Lozhkov, Guilherme Penedo, Thomas Wolf, and Leandro von Werra. 2024. Cosmopedia.

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.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. 2023. Symbolic discovery of optimization algorithms.

Tri Dao and Albert Gu. 2024. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality.

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

Martin Gerlach and Francesc Font-Clos. 2020. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1).

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces.

Vineet Gupta, Tomer Koren, and Yoram Singer. 2018. Shampoo: Preconditioned stochastic tensor optimization.

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.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention.

Jordan Keller. 2024. Muon: A drop-in optimizer for faster convergence. `https://kellerjordan.github.io/posts/muon/`.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Disen Lan, Weigao Sun, Jiaxi Hu, Jusen Du, and Yu Cheng. 2025. Liger: Linearizing large language models to gated recurrent structures.

Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. 2024. Sophia: A scalable stochastic second-order optimizer for language model pretraining.

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, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. 2025. Muon is scalable for llm training.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. 2024. Fineweb-edu: the finest collection of educational content.

Brian MacWhinney. 2000. *The CHILDES project: Tools for analyzing talk: Transcription format and programs, Vol. 1, 3rd ed.* Lawrence Erlbaum Associates Publishers, Mahwah, NJ, US. This book is the first of two volumes (for volume 2, see record 2000-03631-000) documenting the three components of the CHILDES Project: CHAT transcription manual and CLAN analysis manual. Useful for novice and experienced users, as well as instructors and students studying child language transcripts.

Chien Van Nguyen, Ruiyi Zhang, Hanieh Deilamsalehy, Puneet Mathur, Viet Dac Lai, Haoliang Wang, Jayakumar Subramanian, Ryan A. Rossi, Trung Bui, Nikos Vlassis, Franck Dernoncourt, and Thien Huu Nguyen. 2025. Lizard: An efficient linearization framework for large language models.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von

Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all – adapting pre-training data processing to every language.

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.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding.

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.

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*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization.

Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Ré. 2024. The hedgehog the porcupine: Expressive linear attentions with softmax mimicry.

A Dataset Curation: Prompt

```
Below is an extract from a web page. Evaluate whether the page has
    ↪ a high educational value and could be useful in an
    ↪ educational setting for teaching from primary school to
    ↪ grade school levels using the additive 5-point scoring
    ↪ system described below. Points are accumulated based on the
    ↪ satisfaction of each criterion:
Scoring Criteria:
- +1 Educational Relevance: The extract contains factual or
    ↪ instructional content related to general knowledge,
    ↪ science, math, language, or other academic domains, even if
    ↪ mixed with irrelevant content like ads or unrelated
    ↪ commentary.

- +1 Coherence and Structure: The extract has a recognizable
    ↪ structure (e.g. paragraphs, bullet points, logical flow)
    ↪ and is written in a mostly coherent and syntactically
    ↪ correct way, even if it includes some tangents or
    ↪ inconsistencies.

- +1 Readability and Simplicity: The language is accessible to
    ↪ grade school students, avoiding technical jargon or overly
    ↪ complex sentence constructions. Sentences are clear,
    ↪ concise, and vocabulary is age-appropriate.

- +1 Explainability and Pedagogical Quality: Concepts are
    ↪ explained, not just stated. The text may include analogies,
    ↪ definitions, or examples that make it easier to understand.
    ↪ It supports comprehension and learning.

- +1 Learnability by Small Models: The extract is particularly
    ↪ suitable for training smaller language models: it avoids
    ↪ long-range dependencies, sticks to one or two topics, and
    ↪ has low noise and high signal. Ideal examples follow a
    ↪ pattern, use repetition to reinforce structure, and do not
    ↪ rely heavily on context outside the extract.

The extract:
{0}

After examining the extract:
- Briefly justify your total score, up to 100 words.
- Conclude with the score using the format: "Educational score:
    ↪ <total points>"
```

Figure 2: LLM-based prompt used to assign a **custom** educational scores to FineWeb-Edu samples. The prompt includes a 5-point additive scoring rubric focusing on pedagogical value, readability, and coherence.

B Model Configurations

Hyperparameter	Value
Hidden Size	1024
Intermediate Size	1536
Num Attention Heads	16
Num Hidden Layers	
- Transformer	26
- BLaLM	24
Vocab Size	15K
Parameter Count	
- Transformer	250M
- BLaLM	270M

Table 8: Model configurations for Transformer and BLaLM. Hidden layer count is adjusted to ensure comparable parameter counts across architectures.

C Experiment 1: Full Results

DATASET	BLIMP acc.	B. SUPPL. acc.	ENTITY acc.	EWOK acc.	EYE ΔR^2	READING ΔR^2	AVG.
Baseline-10M (7)	64.96	66.8	40.01	51.55	0.98	0.45	37.45
Baseline-10M (8)	65.49	60.4	35.22	52.27	0.76	0.4	35.75
Baseline-100M (9)	75.43	63.6	20.47	53.36	0.59	0.26	35.61
Baseline-100M (6)	75.68	65.2	34.82	51.82	0.82	0.33	38.11
Our-10M (8)	67.99	63.6	39.46	52.27	1.09	0.65	37.51
Our-10M (9)	66.81	59.2	41.97	52.73	0.85	0.53	37.01
Our-100M (10)	74.51	57.6	14.65	55.73	1.18	0.9	34.09
Ours-100M (10)	74.6	57.2	16.56	53.64	1.21	0.57	33.96

Table 9: Detailed results comparing our curated dataset to the official BabyLM baseline. The number in parentheses indicates the best-performing training epoch.

D Experiment 2: Full Results

Track	Model	LR	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
Strict-Small	Transformer (10)	3e-4	64.47	58.8	15.71	49.45	1.06	0.56	31.67
	Transformer (9)	4e-4	64.95	57.2	18.07	51.36	1.13	0.93	32.27
	Transformer (10)	5e-4	65.37	54.4	14.5	52.18	0.83	0.44	31.28
	BLaLM (9)	3e-4	63.07	59.6	38.52	51.27	0.82	0.5	35.63
	BLaLM (9)	4e-4	66.93	55.6	28.74	52.09	0.91	0.5	34.12
	BLaLM (9)	5e-4	66.72	55.6	40.93	51.0	0.91	0.6	35.96
Strict	Transformer (10)	3e-4	72.81	60.8	18.51	55.09	0.82	0.58	34.76
	Transformer (9)	4e-4	72.44	62.0	20.63	53.36	1.02	0.74	35.03
	Transformer (10)	5e-4	72.47	63.2	18.21	51.55	1.11	0.71	34.54
	BLaLM (10)	3e-4	73.40	61.2	14.70	54.55	0.93	0.51	34.21
	BLaLM (10)	4e-4	74.60	57.2	16.56	53.64	1.21	0.57	33.96
	BLaLM (10)	5e-4	74.49	60.4	21.99	53.91	1.03	0.71	35.42

Table 10: Detailed zero-shot results for Transformer and BLaLM across BabyLM benchmarks. Parentheses indicate best-performing epoch.

E Experiment 3: Full Results

Model	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
AdamW							
Run1 (10)	74.96	60.8	13.25	55.55	0.85	0.61	34.33
Run2 (10)	74.37	65.6	32.8	54.82	0.9	0.68	38.19
Run3 (10)	76.54	61.6	14.72	53.82	0.92	0.57	34.69
Muon							
Run1 (10)	76.4	70.4	22.27	55.91	1.17	0.79	37.82
Run2 (10)	75.82	66.4	15.39	55.73	1.1	0.88	35.88
Run3 (10)	75.27	64.0	15.98	53.18	1.07	0.76	35.03

Table 11: Zero-shot results for xLSTM models trained with AdamW and Muon optimizers (3 runs). Parentheses indicate best-performing epoch.

F Experiment 4: Full Results

Learning Rate	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
STRICT-SMALL							
1e-4 (6)	57.16	58.4	39.95	52.73	0.28	0.3	36.04
2e-4 (6)	61.37	59.2	41.81	52.55	0.74	0.59	34.80
3e-4 (7)	67.19	60.8	33.44	50.55	1.14	0.58	35.61
4e-4 (8)	69.55	58.0	41.91	52.36	1.17	0.68	37.27
5e-4 (6)	69.12	59.6	26.8	52.91	1.1	0.69	35.03
6e-4 (9)	69.98	61.6	18.89	52.91	1.03	0.65	34.17
7e-4 (7)	70.68	60.4	38.87	53.82	0.96	0.47	37.53
STRICT							
2e-4 (9)	73.81	60.4	17.42	52.64	0.87	0.54	34.28
3e-4 (10)	75.91	67.6	15.06	55.09	0.81	0.5	35.82
4e-4 (8)	66.82	54.8	33.54	54.0	0.92	0.43	35.08
5e-4 (10)	76.25	66.4	15.08	55.55	1.03	0.62	35.82
5.5e-4 (9)	76.1	63.6	28.04	55.64	0.87	0.73	37.49
6e-4 (9)	76.42	59.2	18.64	54.36	0.94	0.83	35.06
7e-4 (9)	75.85	66.0	19.46	53.73	1.03	0.55	36.10

Table 12: Full results for Experiment 4. The number in brackets after each learning rate denotes the best performing epoch.

G Experiment 5: Full Results

Mechanism	BLiMP acc.	B. Suppl. acc.	Entity acc.	EWoK acc.	Eye ΔR^2	Reading ΔR^2	Avg.
strict-small							
ShortConv (6)	67.13	57.6	38.82	52.82	1.42	0.71	36.41
SWA (9)	64.86	54.4	43.17	52.91	1.1	0.52	36.16
SWA With Memory (7)	65.83	52.8	35.97	52.18	1.99	1.04	34.96
SWA DynMod (6)	67.36	54.4	41.28	51.36	1.69	0.83	36.15
SWA DynMod Bounded (7)	65.59	52.4	33.86	52.36	1.35	0.93	34.41
Hedgehog (10)	68.3	53.2	24.5	53.0	1.52	0.99	33.58
Hedgehog + SWA (6)	65.43	54.4	42.49	52.73	1.55	0.94	36.25
strict							
ShortConv (8)	74.31	61.2	15.63	54.18	1.05	1.08	34.57
SWA (8)	74.29	60.8	21.42	56.45	1.2	1.02	35.86
SWA With Memory (10)	71.52	65.2	31.04	54.18	0.85	0.49	37.21
SWA DynMod (9)	76.39	68.0	31.32	55.64	0.83	0.76	38.82
SWA DynMod Bounded (10)	73.64	66.0	22.36	53.55	0.97	0.75	36.21
Hedgehog (8)	74.64	62.0	24.69	56.73	1.32	0.55	36.65
Hedgehog + SWA (7)	72.94	58.8	16.28	54.64	1.69	0.9	34.20

Table 13: Full results for Experiment 5. Parentheses indicate the best-performing epoch per configuration.

G.1 Alpha Value Development for DynMod Runs

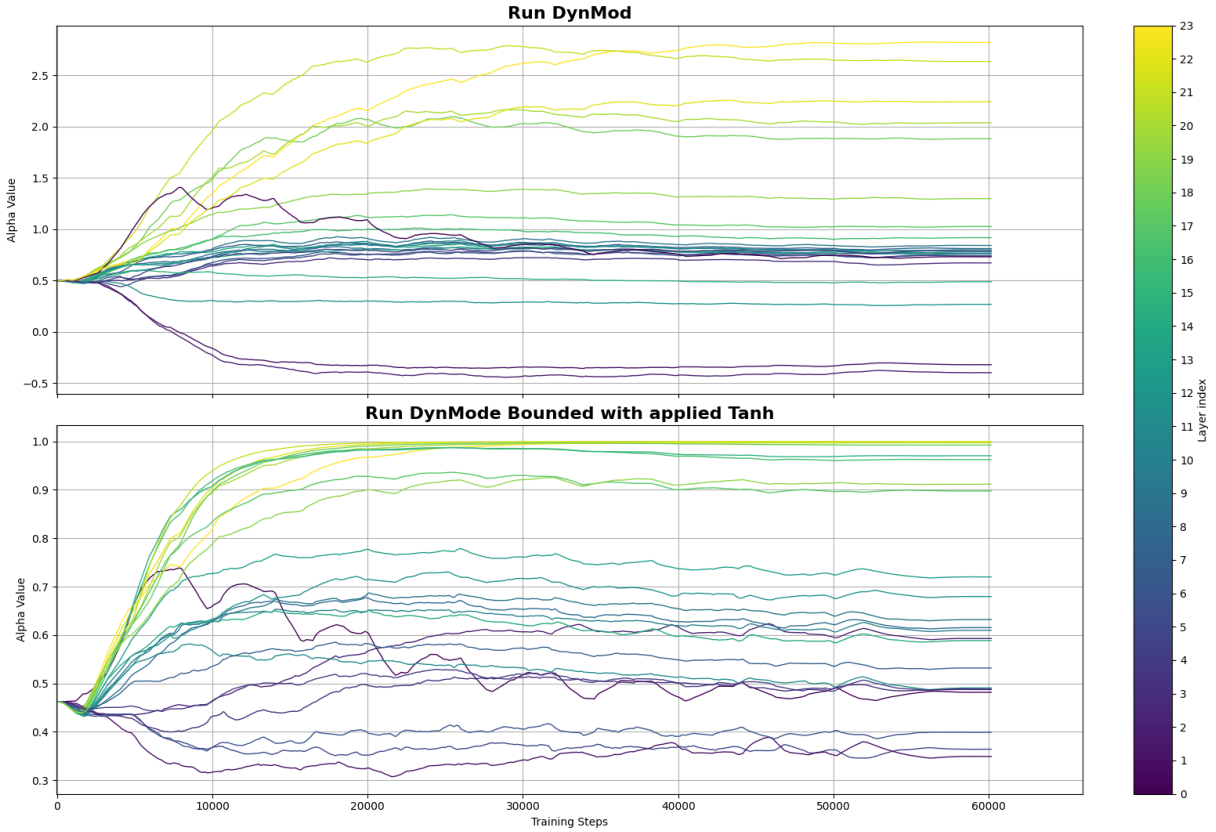

Figure 3: Layer-wise development of dynamic modulation weights () during training for the bounded DynMod variant. We apply the tanh function to stabilize values. Later layers show increased reliance on local mixing.

H Final Submission: Full Results

BENCHMARK	BLaLM STRICT-SMALL	BLaLM STRICT
BLIMP	67.0	74.7
B. SUPPL.	53.3	61.0
ENTITY TRACKING	33.7	22.2
EWOK	50.6	53.6
EYE TRACKING	1.1	1.1
SELF PACED READING	1.0	0.6
WUG ADJ. NORM.	50.3	47.5
WUG. PAST TENSE	-20.7	37.5
COMPS	50.5	58.3
AOA	8.6	8.6
AVERAGE	29.54	36.49

Table 14: Final BabyLM benchmark results for BLaLM-STRICT-SMALLand BLaLM-STRICTmodels. Includes hidden tasks.

MODEL	BOOLQ acc.	MNLI acc.	MRPC acc.	QQP acc.	MULTIRC acc.	RTE acc.	WSC acc.	AVG.
BLaLM-STRICT-SMALL	64.03	34.18	69.60	59.92	57.54	54.67	61.54	57.35
BLaLM-STRICT	64.03	34.27	69.10	58.90	57.54	51.79	61.53	56.70

Table 15: Performance of BLaLM-STRICT-SMALLand BLaLM-STRICTon (Super)GLUE tasks after fine-tuning.