

MoEP+: Enhancing Modular Expert Paths for Data-Efficient Language Modeling

Noor E. Mustafa

Lund University

no6827mu-s@student.lu.se

Valdemar Melkersson

Lund University

va7575me-s@student.lu.se

Pierre Nugues

Lund University

pierre.nugues@cs.lth.se

Abstract

We present MoEP+, an enhanced version of the Modular Expert Paths (MoEP) decoder. MoEP+ retains MoEP’s top- k routing over parallel expert paths, simplifies its dimensional mapping layers, and modernizes the backbone with RoPE, RMS normalization, and SwiGLU. It also introduces a projection-specialized routing scheme in which different experts apply nonlinear residual transformations to distinct attention projections. We evaluate MoEP+ on the BabyLM 2026 strict-small track, which is limited to 10 million words, and present it as a combined architectural recipe rather than attributing the observed gains to any single component.

Under a matched training budget, MoEP+ outperforms the unmodified MoEP baseline on all eight zero-shot benchmarks. On five of these benchmarks, the advantage persists even when the baseline is trained for five times as many epochs. Most notably, MoEP+ is substantially more data-efficient: after only two epochs, it surpasses the unmodified MoEP’s ten-epoch BLiMP score. Despite its smaller size, it also remains competitive with the official BabyLM 2026 baselines, with its advantages concentrated in the reading-based measures.

1 Introduction

Transformer-based large language models (LLMs), building on the architecture introduced by Vaswani et al. (2017); Radford et al. (2018), require large amounts of training data to approximate language distributions, resulting in substantial computational costs. Sparse Mixture-of-Experts (MoE) architectures, first introduced by Shazeer et al. (2017), can reduce these costs by decomposing a large network into parallel subnetworks that are activated sparsely. Modular Expert Paths (MoEP), introduced by Tapaninaho (2025) as a decoder-only model for the 2025 BabyLM Challenge, is a recent instance of this approach. In this work, we

adopt the MoEP implementation as our starting point and investigate how it can be improved.

We revisit MoEP with two observations in mind. First, its backbone uses earlier Transformer components, including learned positional embeddings, LayerNorm, standard MLP feed-forward blocks, and linear query, key, and value projections. Since then, RoPE (Su et al., 2024), RMSNorm (Zhang and Sennrich, 2019), and SwiGLU feed-forward blocks (Shazeer, 2020) have become common in modern decoder architectures. In parallel, Karbevski (2026) and Karbevski and Mijoski (2025) showed that replacing the linear query projection with a nonlinear residual function,

$$Q(X) = \frac{X + f_{\theta}(X)}{2},$$

can outperform a linear baseline with more nonembedding parameters. Second, MoEP does not explicitly encourage its parallel experts to assume distinct functional roles.

We address both observations by introducing **MoEP+**¹ an enhanced version of MoEP that

1. Simplifies the dimensional mapping layers,
2. Modernizes the backbone with RoPE, RMSNorm, and SwiGLU (see Table 1), and
3. Introduces *projection-specialized experts*.

Different routed paths apply nonlinear residual transformations to different attention projections: some specialize in queries, whereas others specialize in keys or values, as illustrated in Figure 1. Under top- k routing, the model can therefore combine, for example, the output of a nonlinear-query expert with that of a nonlinear-key expert without introducing symmetric residual transformations within

¹The source code and instructions for obtaining and pre-processing the training data are available at <https://github.com/vame-lth/MoEPPlus/tree/main>.

the same attention head. This design avoids the unweighted, parameter-free $XX^\top$ term that Karbevski (2026) seeks to prevent by restricting non-linearity to the query projection alone. This term would otherwise bias the attention matrix toward self-similarity.

Crucially, we introduce these modifications as a unified architectural package and evaluate the resulting model as a whole. Because several components are modified simultaneously, *we do not claim that any single component is responsible for the observed gains*; isolating their individual contributions is left to future work. Our results on the BabyLM strict-small track show that the combined design consistently outperforms the unmodified baseline and is substantially more data-efficient: MoEP+ matches or exceeds the baseline’s best BLiMP score after only a fraction of the training.

2 Related Work

2.1 Modern Decoder Components

Several components used in recent decoder-only language models differ from the original Transformer formulation (Vaswani et al., 2017). Rotary Position Embeddings (RoPE) replace learned absolute positional embeddings with a relative position mechanism applied directly to the query and key representations (Su et al., 2024). RMSNorm simplifies LayerNorm by normalizing only with the root mean square of the activations, reducing normalization overhead while preserving training stability (Zhang and Sennrich, 2019). Gated feed-forward networks such as SwiGLU extend the standard Transformer MLP by introducing a multiplicative gating mechanism, which has been shown to improve Transformer performance (Shazeer, 2020).

MoEP+ incorporates these components as part of a single architectural update to the original MoEP backbone. We therefore do not evaluate RoPE, RMSNorm, or SwiGLU as isolated modifications. Instead, they are treated as part of the complete MoEP+ configuration, together with projection-specialized routing and nonlinear attention projections.

2.2 BabyLM and Small-Scale Language Models

The BabyLM challenge is a shared-task, focused on small language models, providing several tracks: The small-track with a 100M word budget, and the strict-small track with a 10M word budget,

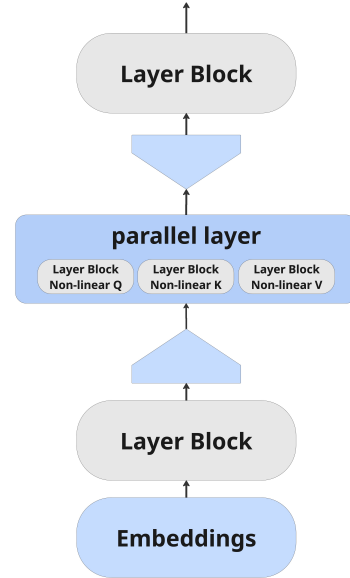


Figure 1: Overall architecture of MoEP+. The model follows a decoder-only Transformer structure with an embedding layer, an initial global Transformer block, a routed parallel layer, a final global Transformer block, and an output layer. The global blocks operate at the full model dimension $d_{\text{model}} = 384$, while the routed parallel layer operates at the reduced parallel dimension 192. The first global block applies nonlinearity to the query projection, the final global block applies nonlinearity to the key projection, and the parallel blocks alternate between query-, key-, and value-specialized nonlinear configurations.

among others. Findings from the BabyLM challenge (Warstadt et al., 2023; Choshen et al., 2026) have demonstrated that, when subject to the mentioned word-budget constraints, architectural innovations can potentially improve performance.

2.3 Modular Expert Paths (MoEP)

The Modular Expert Paths (MoEP) architecture, introduced by Tapaninaho (2025), is a novel language model presented in the third iteration of the BabyLM Challenge and an extension of the Papafarmer architecture proposed by Tapaninaho and Oussala (2025). Operating within the mixture-of-experts (MoE) paradigm, MoEP employs top- k routing across parallel computational paths.

While MoEP provides the foundation for our architecture, we introduce several modifications. First, we simplify the dimensional mapping layers. Whereas the original MoEP employs multiple routed linear layers to reduce and restore the dimensionality of representations entering and exiting the parallel blocks, our architecture uses a single

linear projection for each of the down- and up-projections. Furthermore, we introduce nonlinearity into the queries, keys, and values attention projections, SwiGLU feed-forward, RMSNorm and incorporate rotary position embeddings (RoPE) (Su et al., 2024).

2.4 Beyond Linearity

Our exploration of nonlinear attention pathways is motivated by Karbevski (2026), who propose replacing the redundant linear query projection matrix W_Q with a nonlinear residual function of the form

$$Q(X) = \frac{X + f_\theta(X)}{2},$$

where f_θ is a bottleneck MLP with $d^2 + O(d)$ learnable parameters, implemented as

$$f_\theta(X) = \text{LN}(\text{GELU}(\text{LN}(X)W_1)W_2),$$

with $W_1 \in \mathbb{R}^{d \times r}$, $W_2 \in \mathbb{R}^{r \times d}$, and $r = d/2$, and where GELU is implemented using the standard hyperbolic tangent approximation,

$$\text{GELU}(x) \approx \frac{x}{2} (1 + \tanh[\sqrt{\frac{2}{\pi}} (x + 0.044715x^3)]).$$

This gives $2dr = d^2$ matrix parameters, while the normalization layers add $O(d)$ additional parameters. Their experiments on GPT-3–small–style models showed that this nonlinear query projection outperforms a baseline that uses 12.5% more non-embedding parameters. However, to the best of our knowledge, the interaction between such nonlinear projections and MoE models, such as MoEP, has not previously been investigated.

Furthermore, Karbevski (2026) restricts nonlinearity exclusively to the query projection. The author argues that applying the same residual structure symmetrically to the key projection would introduce an unweighted, parameter-free $XX^\top$ term, thereby biasing the attention matrix toward self-similarity.

Rather than adopting this restriction, MoEP+ distributes nonlinearity across projection-specialized experts, as illustrated in Figures 2 and 1. By assigning query, key, and value nonlinearities to different routed blocks, the model gains access to nonlinear key and value processing without introducing the symmetric $XX^\top$ term within a single attention computation.

Note that our implementation of f_θ differs in that we use LayerNorm in place of the inner RMSNorm and the tanh approximation of GELU.

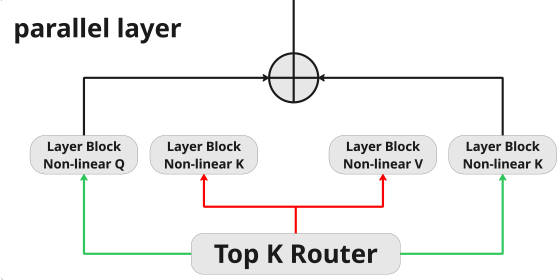


Figure 2: Top- k routing in the MoEP+ parallel layer. With $k = 2$, the router selects two projection-specialized expert blocks per token and combines their outputs using learned routing weights. This allows different nonlinear projection roles, such as query and key or query and value, to be combined without placing all nonlinear projections in the same attention block.

3 Methodology

The projection-specialized routing illustrated in Figures 2 and 1 distributes nonlinearity across the attention projections by assigning distinct projection roles to different blocks. Rather than applying the same nonlinear residual structure to queries, keys, and values within a single attention block, MoEP+ allows different routed components to specialize in different projections. The first large global layer block, see Figure 3, uses a nonlinear query projection while keeping keys and values linear. The final large global layer block uses a nonlinear key projection, while keeping queries and values linear. Inside the intermediate *Parallel-Layer*, see Figure 1, the parallel blocks cyclically alternate between query, key, and value nonlinear configurations:

For a selected projection $P \in \{Q, K, V\}$, the nonlinear residual projection is defined as:

$$P_{\text{nonlin}}(X) = \frac{1}{2} (X + f_\theta(X)), \quad (1)$$

while the remaining projections are kept linear. For example, in a key-specialized block,

$$K(X) = \frac{1}{2} (X + f_\theta(X)), \quad (2)$$

with

$$Q(X) = XW_Q, \quad V(X) = XW_V. \quad (3)$$

Top- k sparse routing has previously been used to distribute tokens across a limited subset of experts while maintaining conditional computation (Shazeer et al., 2017; Lepikhin et al., 2021).

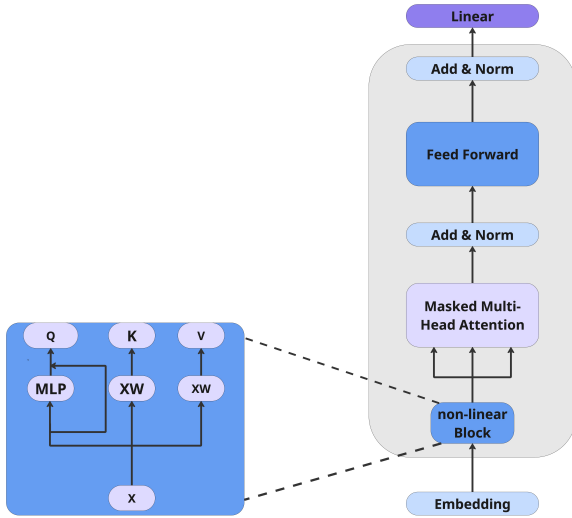


Figure 3: Projection-specialized attention block used in MoEP+. Each block keeps the standard query, key, and value structure, but applies the nonlinear residual projection to only one selected projection path. The shown example applies the nonlinear transformation to the query projection, while keys and values remain linear. Other blocks use the same structure but specialize the nonlinear projection to keys or values instead.

Our intuition is that top- k routing with $k = 2$ can combine outputs from experts with different projection specializations. One selected expert may contribute a nonlinear query representation, while another selected expert may contribute a nonlinear key or value representation; see Figure 2. Since the routed expert outputs are weighted and combined, the model can learn to use different nonlinear attention features without forcing both query and key to use the same residual structure inside a single attention computation. This gives the model access to nonlinear query and key processing while avoiding a direct symmetric residual construction that would explicitly introduce the problematic XX^T term.

4 Training and Experimental Setup

4.1 Data

Our models are trained on the BabyLM 2026 *Strict-small* (10M-word) track corpus (Choshen et al., 2026). The corpus contains roughly 10M words drawn from six developmentally plausible, predominantly transcribed-speech sources: CHILDES (child-directed speech), the spoken portion of the British National Corpus (BNC), Switchboard, OpenSubtitles, Simple English Wikipedia, and Project Gutenberg. No additional data is used for

training, and the same corpus is used for MoEP+ and Unmodified MoEP from Tapaninaho (2025).

4.2 Tokenization

We use the original MoEP article’s single shared subword tokenizer for all experiments: a **GPT-2** style byte-pair encoding (BPE) tokenizer (Sennrich et al., 2016) with a vocabulary of 16,384 tokens, trained at the 16K level on the 2025 BabyLM Strict-small track corpus. Sequences are packed to a fixed length of 512 tokens.

4.3 Training

Each variant (MoEP and MoEP+) is trained with the configuration in Table 3. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a peak learning rate of 3×10^{-4} and a cosine schedule with linear warmup over 800 steps. MoEP and MoEP+ are trained for 10 and 2 epochs over the 10M-word corpus. For the MoEP+ model, we stopped training after two epochs, as the model started to overfit. Training is performed in full 32-bit precision on a single Radeon RX 7900 XTX (24 GB) GPU and takes approximately 1.5 hours per model for 10 epochs. Tables 2 and 3 show additional hyperparameters.

To enable analysis of how performance evolves over the course of training, we save checkpoints at fixed word-count budgets following the BabyLM thresholds: every 1M words up to 10M, and then every 10M words up to 100M. Each configuration is trained for three independent runs, differing in their random initialization. The values reported in Tables 4, 5, and 6 are averages over the three runs for both MoEP+ and the unmodified MoEP baseline.

5 Results

The evaluation pipeline follows the BabyLM Challenge 2026 setup and includes both zero-shot and fine-tuned evaluations. Our models were evaluated on the following benchmarks: BLiMP (Warstadt et al., 2020), BLiMP Supplement, COMPS (Misra et al., 2023), Entity Tracking (Kim and Schuster, 2023), EWoK (Ivanova et al., 2025), Reading Eye Tracking, Reading Self-Paced, and Reading Overall. For fine-tuning, we evaluate on tasks drawn from GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a), including BoolQ, MNLI, MRPC, MultiRC, QQP, RTE, and WSC.

Variant	Pos. emb.	Norm	FFN	Params
Unmodified (baseline)	Learned	LN	MLP (multiplier 4)	28.4M
MoEP+	RoPE	RMS	SwiGLU (multiplier 8/3)	28M

Table 1: The baseline (Unmodified MoEP) uses learned positional embeddings, LayerNorm (LN), a standard MLP feed-forward block, and linear $Q/K/V$ projections. Our model replaces these with rotary positional embeddings (RoPE), RMSNorm, and a SwiGLU feed-forward block, and additionally alters the query, key, and value projections.

Hyperparameter	Value
Vocabulary size	16,384
Model dimension d_{model}	384
Parallel dimension	192
Layers	12
Attention heads	6
Head dimension	64
Parallel heads	3
Parallel blocks	4
Number of experts	4
Top- k routing	2
Max sequence length	512
Dropout (embd/resid/attn)	0.1

Table 2: Shared architectural hyperparameters used for MoEP+ and the unmodified MoEP baseline. These settings define the model size, attention configuration, routing setup, context length, and regularization parameters used in the matched-budget experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	3×10^{-4}
Adam β	(0.9, 0.999)
Adam ϵ	1×10^{-6}
Weight decay	0.1
Batch size	16
Gradient accumulation steps	1
Training epochs	MoEP: 10, MoEP+: 2
LR scheduler	Cosine
Warmup steps	800
Sequence length	512
Precision	FP32
GPU	RX 7900 XTX
VRAM	24 GB
Runs per configuration	3

Table 3: Training setup and optimization parameters used for MoEP+ and the unmodified MoEP baseline. The configuration includes the optimizer, learning rate, and others.

5.1 Evaluation Protocol

We report results under two comparison settings. First, we compare MoEP+ and the unmodified MoEP baseline under a matched training budget, where both models are evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. This comparison isolates the performance of the two architectures under the same amount of training data.

Second, we compare MoEP+ after two epochs with longer-trained checkpoints of the unmodified MoEP baseline. In this setting, the baseline is evaluated after two, three, six, and ten epochs, corresponding to 20M, 30M, 60M, and 100M words seen during training. This comparison is intended to measure data efficiency: if MoEP+ after two epochs matches or exceeds the unmodified baseline after more training, this suggests that the combined architectural modifications allow the model to reach similar performance with fewer training updates.

All reported MoEP+ and unmodified MoEP values are averages over three independent runs. The unmodified MoEP results reported in this paper are obtained from our own retraining and evaluation setup, using the same data, tokenizer, train-

ing schedule, and evaluation pipeline as MoEP+. Therefore, these values may differ from the official MoEP scores reported on the BabyLM 2025 leaderboard (BabyLM-community, 2025). We use our retrained unmodified MoEP model as the primary baseline because it provides a controlled comparison under matched experimental conditions.

For the BabyLM baseline comparisons, we use the scores reported for the BabyLM Baseline (Interaction) and BabyLM Baseline GPT-2 models on the BabyLM 2026 leaderboard as external reference points (BabyLM Community, 2026). These leaderboard scores are not obtained from re-training by us and are therefore used only to contextualize the performance of MoEP+ relative to the official BabyLM 2026 baselines. In contrast, comparisons between MoEP+ and unmodified MoEP are based on our own three-run averages.

5.2 MoEP+ vs. MoEP

As shown in Table 4, we compare MoEP+ with the baseline MoEP model, referred to here as unmodified MoEP. When both models are trained for two epochs (20M words), MoEP+ outperforms unmodified MoEP on all eight zero-shot benchmarks,

including BLiMP and BLiMP Supplement. The advantage largely persists when comparing MoEP+ with unmodified MoEP trained for longer. MoEP+ outperforms the three-epoch baseline (30M words) on seven of eight benchmarks, and it outperforms both the six-epoch (60M words) and ten-epoch (100M words) baselines on five of eight benchmarks. At 60M words, the unmodified baseline exceeds MoEP+ only on COMPS (by 0.27), Entity Tracking (by 0.56), and EWoK (by 0.02). At 100M words, it exceeds MoEP+ on the same three benchmarks, by 0.45, 1.42, and 0.46, respectively.

For the fine-tuning evaluation, both models were fine-tuned from their two-epoch checkpoints. As shown in Table 5, the two models achieve largely comparable results, although our model obtains an average improvement of 0.29 points.

Table 6 compares MoEP+ with two official BabyLM 2026 baselines trained on the same 10M-word strict-small corpus. We focus primarily on the GPT-2 baseline, which obtains the higher fine-tuned macro average of the two baselines. MoEP+ obtains higher scores than this baseline on BLiMP Supplement (+1.18) and all three reading-based measures: Reading Overall (+1.65), Eye Tracking (+0.54), and Self-Paced (+2.75). On the remaining benchmarks, MoEP+ matches the GPT-2 baseline on EWoK (50.63), remains within one point on Entity Tracking (−0.32) and COMPS (−0.19), and trails it by 2.23 points on BLiMP (63.00 vs. 65.23). On the fine-tuning tasks, MoEP+ achieves a macro average of 63.60, which is 0.21 points lower than the GPT-2 baseline’s score of 63.81.

Relative to the Interaction baseline, MoEP+ obtains higher scores on five of the eight zero-shot benchmarks, including BLiMP Supplement, COMPS, and all three reading-based measures. The two models also achieve nearly identical fine-tuned macro averages (63.60 vs. 63.62). Differences below roughly one point should be interpreted with caution, as we do not report variance estimates for the leaderboard baselines. This applies particularly to the EWoK, Entity Tracking, and COMPS margins against the GPT-2 baseline, as well as the BLiMP margin against the Interaction baseline (63.00 vs. 63.07).

Note: The MoEP+ scores reported on the BabyLM 2026 leaderboard may differ from those presented in our tables because the leaderboard reports the best-performing run, whereas the values in our tables are averaged over three independent runs.

6 Discussion

6.1 Data Efficiency

The clearest and most robust finding is improved data efficiency. After only two epochs, it already exceeds the BLiMP score of the unmodified baseline after 6 epochs, (63.00 vs. 62.12). Thus, MoEP+ achieves a higher BLiMP score using one-third of the training steps. Nevertheless, this faster convergence does not generalize uniformly across tasks: although MoEP+ leads on the reading measures early in training, reading performance declines during extended training, whereas the BLiMP advantage persists. Faster convergence should therefore be interpreted as a BLiMP-specific result rather than a general property across all benchmarks.

6.2 Comparison with the BabyLM 2026 Baselines

Despite using substantially fewer parameters (28M vs. 98.4M), **MoEP+** matches or outperforms the official BabyLM 2026 GPT-2 baseline on five of the eight reported zero-shot benchmarks, as shown in Table 6. MoEP+ outperforms the baseline on BLiMP Supplement and all three reading-based benchmarks, while matching it on EWoK. The largest advantages are observed on Reading Self-Paced and Reading Overall, with gains of 2.75 and 1.65 points, respectively.

On the remaining zero-shot benchmarks, MoEP+ trails the baseline by 0.19 points on COMPS, 0.32 points on Entity Tracking, and 2.23 points on BLiMP. On the fine-tuning tasks, the models achieve similar macro averages (63.60 vs. 63.81). MoEP+ scores higher on MRPC, QQP, and RTE, matches the baseline on WSC, and scores lower on BoolQ, MNLI, and MultiRC. These results suggest that the strengths of MoEP+ are concentrated in reading-based zero-shot measures, whereas its weaknesses are more visible on grammatical acceptability.

The comparison with the Interaction baseline follows a similar pattern, which suggests that this profile reflects a property of MoEP+ rather than a peculiarity of one particular baseline. A fuller investigation of this pattern is left for future work.

6.3 Projection-specialized Routing

Our architecture is motivated by the intuition that top- k routing can combine experts with different nonlinear projection specializations, such as query-, key-, and value-specialized blocks. How-

Average metric	MoEP+ @20M	MoEP @20M Δ 20M	MoEP @30M Δ 30M	MoEP @60M Δ 60M	MoEP @100M Δ 100M
BLiMP Average	63.00	59.14 +3.86	60.40 +2.60	62.12 +0.88	62.04 +0.96
BLiMP Supplement Average	58.43	55.94 +2.49	56.49 +1.94	57.86 +0.57	58.07 +0.36
COMPS Average	51.62	51.02 +0.60	51.16 +0.46	51.89 -0.27	52.07 -0.45
Entity Tracking Average	18.78	17.74 +1.04	18.80 -0.02	19.33 -0.56	20.20 -1.42
EWoK Average	50.63	49.15 +1.48	49.81 +0.82	50.65 -0.02	51.09 -0.46
Reading Eye Tracking Average	10.17	9.50 +0.67	9.74 +0.43	9.58 +0.59	9.38 +0.79
Reading Self-Paced Average	4.39	4.05 +0.34	4.14 +0.25	3.84 +0.55	3.66 +0.73
Reading Overall Average	7.28	6.77 +0.51	6.94 +0.34	6.71 +0.57	6.52 +0.76
Wins	-	8/8	7/8	5/8	5/8

Table 4: Zero-shot comparison between MoEP+ and the unmodified MoEP baseline. MoEP+ is evaluated at a single training point (20M words, i.e., two epochs), while the baseline is evaluated at four checkpoints (20M, 30M, 60M, and 100M words) to measure data efficiency. Bold Δ values indicate that MoEP+ outperforms the corresponding baseline checkpoint. All scores are averages over three independent runs.

Model	Zero-shot Tasks									Fine-tuned Tasks							Macro
	BLiMP	Supp.	EWoK	Entity	COMPS	Reading	Eye Track.	Self-Paced	AoA	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg
MoEP Unmodified	59.14	55.94	49.15	17.74	51.02	6.77	9.50	4.05	0	65.67	47.72	82.97	60.16	62.14	60.43	64.10	28.15 63.31
MoEP+	63.00	58.43	50.63	18.78	51.62	7.28	10.17	4.39	0	65.71	48.82	82.28	64.95	61.93	58.03	63.46	29.37 63.60

Table 5: Comparison between MoEP+ and the unmodified MoEP baseline on zero-shot and fine-tuned tasks under a matched training budget. Both models are evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. In the Macro Avg column, the upper value is the zero-shot average and the lower value is the fine-tuning average. Reported values are averages over three independent runs.

ever, we do not directly verify this behavior in the present work. In particular, we do not analyze whether the router consistently selects complementary projection-specialized experts, or whether specific combinations such as nonlinear queries with nonlinear keys or values emerge during training. A more detailed analysis of routing patterns, expert utilization, and projection-specialization behavior is therefore left for future work.

6.4 Architectural Takeaways

First, modernizing the MoEP backbone with contemporary components such as RoPE, RMSNorm, and SwiGLU, together with projection-specialized experts, yields consistent gains over the original MoEP, particularly during the early stages of training. Second, projection specialization provides a practical approach to nonlinear attention in a routed model: By assigning query, key, or value nonlinearity to different experts, the model gains access to nonlinear key and value processing without introducing the symmetric, parameter-free $XX^\top$ term that in part motivated prior work to restrict nonlinearity to the query projection. Third, parallel models can achieve performance comparable to, and in some cases better than, the official BabyLM 2026 baselines.

7 Future Work

Several directions remain open for extending this work:

- Scale the model by testing deeper architectures with more parallel experts. Another direction is to set top- k to three, which could allow the router to select three nonlinear projection paths corresponding to queries, keys, and values.
- Test whether the same data-efficiency effect holds when the model is trained on a larger dataset, such as the 100M-word setting.
- Further analyze the routing mechanism to determine whether the router consistently combines specific nonlinear projection types, such as keys and values or queries and keys.
- Add a skip-connection mechanism in which less important tokens skip heavy attention computation, while larger or more important tokens receive more attention.
- Test projection-specialized routing with and without nonlinear projections in order to determine whether the observed data-efficiency gains are caused by the nonlinear projections themselves or by the routing specialization.

Model	Zero-shot Tasks									Fine-tuned Tasks								Macro
	BLiMP	Supp.	EWoK	Entity	COMPS	Reading	Eye Track.	Self-Paced	AoA	BoolQ	MNLI	MRPC	MultiRC	QQP	RTE	WSC	Avg	
MoEP+	63.00	58.43	50.63	18.78	51.62	7.28	10.17	4.39	0	65.71	48.82	82.28	64.95	61.93	58.03	63.46	29.37	63.60
BabyLM Baseline (Interaction)	63.07	58.13	51.45	19.62	51.29	5.10	8.50	1.69	0	68.01	51.20	81.0	65.31	64.31	53.96	61.54	28.76	63.62
BabyLM Baseline GPT-2	65.23	57.25	50.63	19.1	51.81	5.63	9.63	1.64	-12.15	67.71	49.84	81.37	65.76	61.67	56.83	63.46	27.64	63.81

Table 6: Comparison between MoEP+ and BabyLM baseline models on zero-shot and fine-tuned tasks. MoEP+ is evaluated after two epochs over the 10M-word strict-small corpus, corresponding to 20M words seen during training. BabyLM baseline scores are taken from the BabyLM 2026 leaderboard and constitute an evaluation after 10 epochs; table names abbreviate the leaderboard entries BabyLM-2026-Baseline-Strict-Small-Interaction, BabyLM-2026-Baseline-GPT2-Strict-Small. In the Macro Avg column, the upper value is the zero-shot average and the lower value is the fine-tuning average. Reported MoEP+ values are averages over three independent runs.

8 Conclusion

This paper introduced MoEP+, a decoder-only Transformer architecture that extends the unmodified MoEP baseline with a set of architectural modifications: nonlinear attention projections, SwiGLU activations, RMSNorm, and Rotary Position Embeddings (RoPE). The central idea is to combine token-adaptive parallel routing with projection-specialized experts, so that different routed components can be assigned different query, key, or value projection roles.

The empirical results show that this combined architecture improves data efficiency in the BabyLM 2026 strict-small setting. Under a matched training budget, MoEP+ outperforms the unmodified MoEP baseline on all eight zero-shot benchmarks after two epochs of training. More importantly, MoEP+ surpasses the baseline’s 6-epoch BLiMP score after only two epochs, showing that the full set of modifications allows the model to reach stronger grammatical performance with substantially less training.

MoEP+ also performs competitively against the official BabyLM 2026 baselines, despite using substantially fewer parameters. Relative to the GPT-2 baseline, it outperforms the baseline on four of the eight zero-shot benchmarks and matches it on EWoK. Relative to the Interaction baseline, it obtains higher scores on five of the eight zero-shot benchmarks. At the same time, MoEP+ does not uniformly outperform either baseline, and its gains are concentrated in specific benchmarks rather than across the full evaluation suite.

Overall, the results suggest that the MoEP+ modification package is a promising direction for improving sample efficiency in small language models. However, the present experiments do not isolate which individual component is responsible for

the observed improvements. In particular, we cannot conclude whether the gains are caused by the nonlinear projections, the routing specialization, the updated Transformer components, or the interaction between them. Future work should therefore focus on larger-scale experiments, controlled ablation studies, and detailed routing analysis.

Limitations

Unverified expert specialization. The projection-specialized routing mechanism is motivated by the assumption that different experts learn to represent distinct projection nonlinearities and that the router combines them appropriately. However, we do not directly verify whether such specialization emerges. In particular, we report no routing statistics, expert-utilization metrics, or analyses of which experts are selected for different input types.

Limited scale and scope. All experiments are conducted at a single model scale of approximately 28M parameters, using one training corpus from the 10M-word strict-small track and a single tokenizer. It therefore remains unclear whether the observed improvements, particularly the gains in data efficiency, generalize to larger model sizes or training budgets.

Acknowledgments

This work was partially supported by *Vetenskapsrådet*, the Swedish Research Council, registration number 2021-04533.

References

BabyLM-community. 2025. BabyLM Leaderboard 2025: All Tasks. <https://huggingface.co/spaces/BabyLM-community/>

- [babylm-leaderboard-2025-all-tasks](#). Hugging Face Space, accessed 8 July 2026.
- BabyLM Community. 2026. BabyLM Leaderboard 2026. <https://huggingface.co/spaces/BabyLM-community/BabyLM-Leaderboard-2026>. Hugging Face Space, accessed 26 June 2026.
- Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. [Babylm turns 4 and goes multilingual: Call for papers for the 2026 babylm workshop](#). *Preprint*, arXiv:2602.20092.
- Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. [Elements of world knowledge \(EWoK \): A cognition-inspired framework for evaluating basic world knowledge in language models](#). *Transactions of the Association for Computational Linguistics*, 13:1245–1270.
- Marko Karbevski. 2026. [Beyond linearity in attention projections: The case for nonlinear queries](#). In *ICLR 2026 Workshop GRaM*.
- Marko Karbevski and Antonij Mijoski. 2025. [Key and value weights are probably all you need: On the necessity of the query, key, value weight triplet in decoder-only transformers](#). *CoRR*, abs/2510.23912.
- Najoung Kim and Sebastian Schuster. 2023. [Entity tracking in language models](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855, Toronto, Canada. Association for Computational Linguistics.
- Dmitry Lepikhin, HyukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. [GShard: Scaling giant models with conditional computation and automatic sharding](#). In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.
- Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. [COMPS: conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2920–2941. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. [Improving language understanding by generative pre-training](#). Preprint.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Noam Shazeer. 2020. [Glu variants improve transformer](#). *Preprint*, arXiv:2002.05202.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. [Outrageously large neural networks: The sparsely-gated mixture-of-experts layer](#). In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*. OpenReview.net.
- Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. [Roformer: Enhanced transformer with rotary position embedding](#). *Neurocomputing*, 568:127063.
- Joonas Tapaninaho. 2025. [MoEP: Modular expert paths for sample-efficient language modeling](#). In *Proceedings of the First BabyLM Workshop*, pages 540–547, Suzhou, China. Association for Computational Linguistics.
- Joonas Tapaninaho and Mourad Oussala. 2025. [Papaformer: Language model from pre-trained parallel paths](#). *Preprint*, arXiv:2508.00544.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. [Superglue: A stickier benchmark for general-purpose language understanding systems](#). In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. [BLiMP: The benchmark of linguistic minimal pairs for English](#). *Trans. Assoc. Comput. Linguistics*, 8:377–392.

Biao Zhang and Rico Sennrich. 2019. [Root mean square layer normalization](#). In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12360–12371.