SLlama: Parameter-Efficient Language Model Architecture for Enhanced Linguistic Competence Under Strict Data Constraints

Victor Adelakun Omolaoye
HPI / University of Potsdam
victor.omolaoye@guest.hpi.de

Babajide Alamu Owoyele
HPI / University of Potsdam
babajide.owoyele@hpi.de

Gerard de Melo
HPI / University of Potsdam
gerard.demelo@hpi.de

Abstract

Scaling data and model size has driven recent advances in language modeling, but this strategy falters under scenarios with strict data constraints, as in the *BabyLM Challenge*. However, insights from training compute-optimal large language models highlight that smaller models trained on more data outperform larger counterparts trained inadequately, emphasizing the need for compact architectures. Furthermore, while embedding weight tying is a common parameter-reduction technique, we find that it significantly diminishes linguistic competence in compact models. In response, we explore alternative architectural strategies that preserve the parameter-efficiency of tied models without sacrificing the representational benefits of untied embeddings. Consequently, we introduce **SLlama**, a Llama-3 architecture variant that incorporates targeted modifications—*Repeated Reduced Hidden Size and Projection (RRHP)*, *Permutated Weight Attention (PWA)*, *Shared Projection Multi-Layer Perceptron (SPMLP)*, and *Layer Weight Sharing*—to compress Transformer components. Without relying on distillation, SLlama achieves a **31.72% improvement** in linguistic knowledge acquisition over the Baby Llama baseline, with a comparable GLUE score and significantly lower parameter count. These results demonstrate that well-designed, compact models can rival larger ones under strict data constraints.

1 Introduction

Large-scale language models (LLMs) have shown remarkable performance across a wide array of natural language understanding tasks. This success is often attributed to the trend of scaling both model size and training data, a strategy epitomized by recent architectures such as GPT-3 and LLaMA. But reliance on massive datasets and billions of parameters poses challenges when data availability is limited—a scenario increasingly relevant in controlled research settings like the *BabyLM Challenge*.

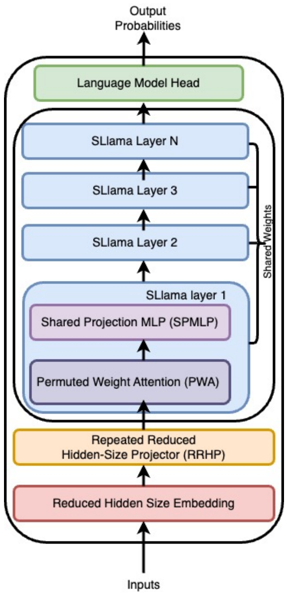

Figure 1: SLlama – Llama Architecture with Reduced Embedding, Repeated Projection, Permuted Weight Attention, Shared Projection MLP and Weight Sharing

Hoffmann et al. (2022) offers a pivotal insight into this problem by demonstrating that, under fixed compute or data budgets, models with fewer parameters but trained on more data tend to outperform larger counterparts trained on less data. In contexts where data resources are limited to barely 10M tokens, it is imperative to design architecturally compact models that can learn efficiently from limited data. This often spurs the adoption of embedding weight tying as a parameter-saving technique. Yet, we find that embedding weight tying impairs the linguistic competence of small models by collapsing distinct representational roles—input encoding and output prediction—into a single shared space. To address this, we investigate architectural strategies that circumvent the need for weight tying while retaining the parameter efficiency of tied

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 23480–23495

models and the representational flexibility of untied models. Our goal is to develop compact yet competent language models optimized for training on just 10 million tokens—the core constraint of the *BabyLM Challenge*.

Hence, we introduce SLlama, a parameter-efficient variant of the Llama-3 architecture designed to balance representational capacity with parameter efficiency. SLlama leverages four key architectural innovations to reduce a model's parameter count and maximize learning from limited data: (1) Repeated Reduced Hidden Size and Projection (RRHP), (2) Permutated Weight Attention (PWA), (3) Shared Projection Multi-Layer Perceptron (SPMLP), and (4) Hidden Layer Weight Sharing.

Crucially, SLlama (with 2.6M model parameters) is trained without distillation of teacher models. Despite this, it achieves a 31.72% improvement in linguistic knowledge acquisition over the Baby Llama[1] baseline (58M), maintains comparable performance on GLUE, and does so with significantly fewer parameters. These results suggest that with thoughtful architectural design, smaller models can not only survive but thrive in data-scarce environments.

1.1 Contributions

Our key contributions are:

1. We demonstrate that embedding weight tying, while widely used for model compression, has a detrimental effect on the linguistic competence of small models.
2. We propose and evaluate architectural strategies that eliminate the need for weight tying while preserving both compactness of weight tying and representational flexibility of untied weights, achieving a 31.72% improvement over the Baby Llama baseline.
3. We introduce SLlama—a novel variant of the Llama-3 architecture tailored for data-constrained settings, which combines several transformer compression techniques to optimize performance under a 10M token constraint.

To ensure transparency and reproducibility, we release code, trained models, and evaluation scripts on GitHub[2] and Hugging Face[3].

2 Preliminaries

The BabyLM Challenge. Choshen et al. (2024) hosted a second round of a shared task where the volume of training data was restricted to 10M tokens. The training and evaluation data contain words that children under the age of 5 years are likely to have heard. This was to motivate small-scale pretraining, which can be a sandbox for developing novel techniques for improving data efficiency. The resulting models would be evaluated on linguistic competence (BLiMP), conceptual understanding (GLUE), and general world knowledge (Ewok).[4]

Among these assessments, BLiMP is of particular interest to us, as we believe that a language model, true to its name, should exhibit meaningful linguistic competence. Moreover, if such competence can be acquired from as few as 10 million tokens, we believe that collecting comparable volumes of data for low-density languages is a feasible goal. This would open the door to training pure language models—those untainted by data from other languages and thus less susceptible to cross-linguistic or cultural bias—for linguistically faithful modeling in low-density settings.

BLiMP Evaluation Unlike HELM (Liang et al., 2022), MMLU (Hendrycks et al., 2020), and FLASK (Cheng et al., 2023), which emphasize high-level task performance and alignment with user intent, BLiMP provides a fine-grained evaluation of core linguistic competence. Although older, BLiMP offers detailed probes into phenomena such as anaphor agreement, argument structure, island effects, irregular forms, and ellipsis—structures fundamental to syntactic and semantic understanding across languages. While recent frameworks reflect the evolving capabilities of large language models, they often obscure fine-grained linguistic diagnostics by focusing on derived abilities like reasoning and discourse. BLiMP, by contrast, foregrounds the grammatical structures that underlie these abilities, offering a clearer lens into a model's linguistic fluency.

Initial Experiments. Motivated by our interest in acquiring linguistic competence from just 10

[1]A student Llama Model distilled from two teacher models Llama (360M) and GPT-2 (705M)

[2]https://github.com/aiintelligentsystems/sllama

[3]https://huggingface.co/aiintelligentsystems/sllama

[4]Since models thrive on experience, the evaluation sets were filtered by the organizers.

million tokens, we adopted a standard approach of sweeping over a range of model configurations—varying hidden sizes from 64 to 1,024 and the number of decoder layers from 2 to 10—while tying the embedding layers and language model heads. While we initially expected the largest model to demonstrate the strongest linguistic competence, we were surprised to find that the best-performing model had a hidden size of 1,024 and only 4 layers. Across the 24 configurations, we observed only weak correlations between model size and performance. Further details on this are given in Appendix A.

Following Hoffmann et al. (2022), which recommends doubling training tokens with each doubling of model size (approximate ratio 1:2), a 10M token budget implies an ideal model size of 5M parameters. In practice, this ratio is often higher. Chinchilla itself has 70B parameters trained on 1.4T tokens (1:20). Based on this, we trained two models with a hidden size of 64 and 6 decoder layers: one with 4.4M parameters and untied weights, closely matching the theoretical target, and another with 2.4M parameters and tied weights, which is closer to the practical design ratio. We show the performance of the two models in Table 1. The experimental results show that the untied model far surpasses the tied model, earlier larger models, as well as the baseline model.

Model Name	**Size**	**BLiMP**	**IoB(%)**
Small Tied	2.4M	56.0%	-19.76
Small Untied	4.4M	**91.9%**	31.72
Big Tied	120M	64.5%	-7.60
Baby Llama	58M	69.8%	0.00
SLlama	2.6M	91.9%	31.72

Table 1: BLiMP scores for models of different sizes under a 10M token training budget and the baseline model. The model in italics is the baseline model. **IoB** means Improvement over Baseline.

This early finding suggests that weight tying negatively affects the linguistic competence of small language models. While we defer a detailed explanation of this phenomenon to a later section, it is important to acknowledge its impact. Despite this drawback, the parameter savings from weight tying are appealing—achieving comparable performance with a 2.4M-parameter model relative to a 4.4M-parameter model offers clear advantages at scale. To mitigate the adverse effects of embedding weight tying while preserving its parameter efficiency, we introduce several parameter reduction techniques into different Transformer components: Linear Hidden-Size Reduction and Projection (LHRP), Attention Hidden-Size Reduction and Projection (AHRP), Repeated Reduced Hidden-Size Projection (RRHP), Shared Key Query Attention (SKQA), Repeat-Reduced-Attention (RRA), Permutated Weight Attention (PWA) and Shared Projection Multi-Layer Perceptron (SPMLP). We adopted existing techniques like Hidden Layer Weight Sharing and intermediate weight reuse. In view of empirical evidence, we streamlined these reduction techniques. The techniques we adopted are collectively named SLlama.

3 Model Reduction Techniques

Recent studies have focused on minimizing the memory footprint of models by reducing parameters within the embedding layer, language model head, and MLP units (Tang et al., 2024; Liu et al., 2024; Zhang et al., 2024b). Our investigation of parameter reduction schemes, detailed below, focuses on the embedding layer, Feed Forward Network, and the self-attention blocks of a Transformer model.

$$\text{Linear}(x, A) = xA^T + b \tag{1}$$

where:

$x \in \mathbb{R}^{m \times h_r}$

$A \in \mathbb{R}^{h_r \times h}$

3.1 Embedding Parameter Reduction

Inspired by the Mixed Dimension Embeddings (MDE) approach proposed by Pansare et al. (2022) and Ginart et al. (2021), we explored alternatives to embedding weight sharing by reducing the dimensionality of the embedding layer. Specifically, we reduced the hidden size (h) of the embedding layer by a factor of four (h_r) . Given that the hidden layers of the decoder are initialized with h, a projection scheme is required to map the reduced embedding dimension to the original hidden size h. We investigated three projection methods: Linear Hidden-Size Reduction and Projection (LHRP), Attention Hidden-Size Reduction and Projection (AHRP), and Repeated Reduced Hidden-Size and Projection (RRHP).

LHRP employs a linear layer as described in Equation 1, reducing the parameters from vh to vh_r. It projects the embedding into a larger dimensional space, assuming the relationship between the small and large representations is linear. AHRP leverages the conventional attention mechanism described in Equation 2. Attention becomes a projector when $Q, K \in \mathbb{R}^{a \times a}$ and $V \in \mathbb{R}^{a \times b}$ where $a \neq b$. AHRP utilises $vh_r + 2h_r + h^2/r$ parameters instead of vh. Conceptually, AHRP magnifies the cogent dimensions of the smaller representations. Finally, RRHP repeats the reduced embedding r times before feeding it to the decoder layers, effectively duplicating the information encoded in the smaller representation r times, hence, reducing the parameter count by $3vh_r$.

3.2 Self-Attention Parameter Reduction

Optimized attention mechanisms with reduced complexity have shown performance comparable to standard multi-head attention (MHA) (Zhang et al., 2024a; Kitaev et al., 2020). While prior work addresses inference-time KV cache memory, our focus is on reducing the parameter count of self-attention in compact language models. Building on earlier embedding reduction strategies, we propose three lightweight attention variants: Shared Key Query Attention (SKQA), Repeat-Reduced Attention (RRA), and Permutated Weight Attention (PWA).

The design of SKQA stems from the interpretation of the attention mechanism as a similarity selection process, which is particularly relevant in language modeling. The attention weights are computed according to Equation (2), and the attention output is derived using Equation (3). Equation (2) can be viewed as computing a probability distribution of inter-token similarity when K and Q are equivalent. We investigated the feasibility of this similarity-based attention by equating the weights of K and Q; effectively reducing parameter count by h^2.

$$\text{Attn\_weight}(Q, K) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (2)$$

$$\text{Attn}(Q, K, V) = \text{Attn\_weight}(Q, K)V \quad (3)$$

where:

$Q \in \mathbb{R}^{h_r \times h_r}$

$K \in \mathbb{R}^{h_r \times h_r}$

$V \in \mathbb{R}^{h_r \times h}$

RRA, in contrast, was inspired by the Repeated Reduced Hidden-Size and Projection reduction technique described earlier, where $Q, K, V \in \mathbb{R}^{h \times h_r}$ also making the hidden representation $h^{(l)} \in \mathbb{R}^{h \times h_r}$, which we subsequently repeat by h_r along the last dimension. Finally, PWA was motivated by the embedding layer reduction strategy presented by Li et al. (2017); Algorithm 1 illustrates its implementation. PWA effectively reduces memory demand from $4h^2$ to $6h$.

Algorithm 1 Permutated Weight Attention

Require: $h, n, m > 0$
Ensure: `permutation`$(n, m) > 3h$
  `permutes` $\leftarrow$ list of `permutation`(n, m)
  $\theta \leftarrow \text{Embedding}(n, h)$
  `q_idx` $\leftarrow$ `permutes[0:h]`
  `k_idx` $\leftarrow$ `permutes[h:2h]`
  `v_idx` $\leftarrow$ `permutes[2h:3h]`
  $Q = \text{Linear}(x, \theta[q\_idx])$
  $K = \text{Linear}(x, \theta[k\_idx])$
  $V = \text{Linear}(x, \theta[v\_idx])$
  `attn` $= \text{Attn}(Q, K, V)$

3.3 MLP Block Parameter Reduction

The Feed-Forward Network (FFN) in Transformers accounts for a large share of parameters, typically using two linear layers: one expanding the hidden size h to nh (with $n = 3$) and another projecting back to h, totaling $6h^2$ parameters. Llama adds a gated projection layer, increasing this to $7h^2$. To reduce this overhead, we propose **Shared Projection MLP (SPMLP)**, which ties the weights of the expansion and reduction layers. We set the weights of the latter to the transpose of the weights of the former thereby saving $3h^2$ parameters.

3.4 Inter-Layer Weight Reduction Strategies

To further reduce model size, we explored two common inter-layer weight reduction techniques: layer reuse and weight sharing. Layer reuse (Liu et al., 2024) passes the hidden state through a layer multiple times (in our case, twice). Thus, if layer reuse $r = 2$, the model is initialized with n/r layers where n is the number of layers, effectively reducing model size by $11nh^2/2$ parameters provided no reduction scheme was introduced. In contrast, weight sharing (Lan et al., 2020) ties the weights of multiple layers, significantly reducing the number of parameters to $11n_gh^2$, where n_g is the number of groups into which the layers are divided. We implemented both techniques, sharing weights across

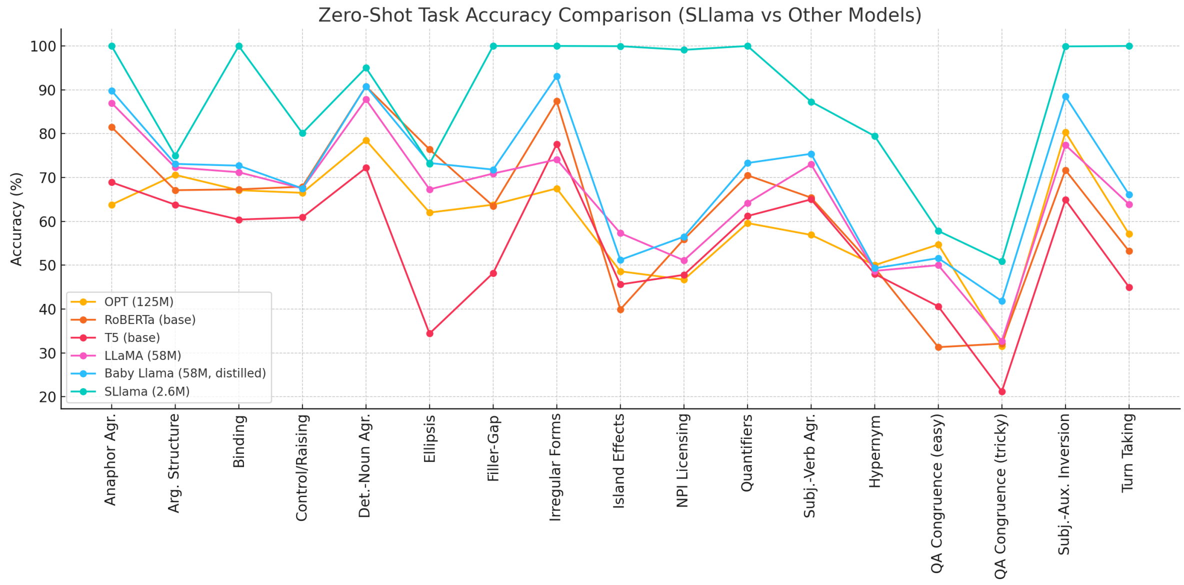

Figure 2: SLlama performance in zero-shot BLiMP tasks relative to Baby Llama and other larger models.

all layers in the model for the weight-sharing approach.

4 Main Experiments

Training Setup. Our experiments (both those described in Section 2 and this section) utilized the BabyLM challenge dataset Choshen et al. (2024), with a complete data description available in Warstadt et al. (2023a). After initial hyperparameter search, all pretraining employed cosine learning rate decay with minimum and maximum rates of 4×10^{-5} and 4×10^{-4}, respectively. We set the gradient accumulation to 2, batch size to 128, and sequence length to 256. Training runs were conducted for 3,000 iterations. We used a single NVIDIA RTX A6000 to train every model in this study. We trained each model multiple times to ascertain consistent results.

Baseline Model and Evaluation Tasks The Baby Llama model (Timiryasov and Tastet, 2023), which was among the leading solutions in the original BabyLM challenge and serves as the state-of-the-art baseline for the second BabyLM challenge[5], was trained using knowledge distillation from two larger teacher models (Llama and GPT2), with the student model reportedly outperforming the teachers.

Evaluations was performed using the pipeline provided by Choshen et al. (2024); Gao et al. (2023), encompassing four tasks: BLiMP, BLiMP supplement (Warstadt et al., 2023c), GLUE (Wang et al., 2019), and Ewok (Ivanova et al., 2024). These tasks assess linguistic competence (BLiMP), conceptual understanding (GLUE), and general world knowledge (Ewok).

Successive Evaluations of the reduction techniques We evaluated the impact of the reduction techniques in each model block and report the results in Table 2. Linear Hidden Reduction and Projection (LHRP), Attention Hidden Reduction and Projection (AHRP), Repeated Reduced Hidden-Size and Projection (RRHP) are schemes to reduce parameter count at the embedding layer. Shared Key Query Attention (SKQA), Repeat-Reduced-Attention (RRA), and Permutated Weight Attention (PWA) were applied to the self-attention implementation. Shared Projection MLP (SPMLP) was applied to the MLP of each decoder layer. Lastly, intermediate layer reuse and inter-layer weight sharing were applied to the decoder layers.

5 Results

We begin by examining the extent to which individual reduction techniques balance parameter efficiency and model performance. Following this, we turn our attention to the combined application of the techniques which use least parameters. We refer to the combination of those techniques as SLlama. We analyse the results with the BLiMP (Warstadt et al., 2023c) framework.

[5] https://github.com/babylm/evaluation-pipeline-2024?tab=readme-ov-file

Model Block	Reduction Techniques	Model Size(M)	BLiMP (Sup.) (%)	Ewok (%)	GLUE (%)	Avg. (%)
Embedding Layer	LHRP	2.8814	60.47 (49.22)	57.58	63.26	57.63
	AHRP	2.8820	59.02 (52.80)	56.58	62.41	57.70
	RRHP	2.8803	91.94 (77.61)	57.91	63.57	72.76 ↑
Self Attention	PWA	4.3200	91.94 (77.61)	57.52	63.47	72.64
	PWA^R	2.7800	91.94 (77.61)	57.76	63.02	72.58 ↑
	RRA	4.3400	59.20 (52.51)	57.88	63.33	58.23
	RRA^R	2.8100	62.28 (51.65)	57.87	62.83	58.66
	SKQA	4.4200	91.94 (77.61)	58.25	63.18	72.75
	SKQA^R	2.8800	91.94 (77.61)	57.71	63.75	72.75
Decoder Layer	Reuse	2.6700	91.94 (77.61)	57.84	63.83	72.81
	Reuse^S	2.6700	91.94 (77.61)	57.63	62.40	72.41
	Share	2.6300	91.94 (77.61)	57.76	63.14	72.62
	Share^S	2.6100	91.94 (77.61)	57.22	62.33	72.28 ↑

Table 2: Performance of LlaMA-based models with different reduction techniques applied at various model blocks. Upward arrows (↑) mark the final choices within each block. Gray shading indicates that a technique uses the selected technique from the previous block. Technique^R denotes the use of Repeated Reduced Hidden-size Projection (RRHP), while Technique^S denotes Shared Projection MLP (SPMLP).

5.1 Comparison of Reduction Techniques

Of the three reduction techniques applied to the embedding layer, RRHP has the optimal balance of reduction and performance as demonstrated in Table 2. Recall that, for RRHP, we divide the hidden dimension by four then repeat for further processing. This implies that the model learns salient representations of tokens, which when repeated, are sufficient to undertake down-stream tasks. In further experiments, we disregarded LHRP and AHRP.

At the self attention block, PWA uses the smallest number of parameters while maintaining a competitive overall performance, closely followed by SKQA, as shown in Table 2. Relative to SKQA, PWA reduces parameter count by a larger factor but suffers a performance drop. Comparing RRPH to PWA^R and SKQA^R, the performance of the latter only dropped by 0.01 while that of PWA^R dropped by 0.18. We consider this drop as a weakness of PWA. However, its gain in parameter reduction compensates for its weakness. We disregarded RRA and SKQA from subsequent experiments.

For the MLPs, although we observe a minor decline in overall performance when SPMLP is included in the architecture, the parameter reduction remains compelling. Hence, we include SPMLP in the SLlama architecture. Furthermore, Table 2 includes the performance of models that employ intermediate layer weight reuse and layer weight sharing in conjunction with SPMLP . The macro-average scores across all models show minimal variation. Thus, the parameter reduction achieved through weight-sharing presents a compelling advantage. Note that the discrepancies introduced by PWA and SPMLP in the overall performance of RRPH variants emerge from the GLUE scores and not the BLiMP scores. This signifies that our model reduction techniques are optimised for linguistic competence with a potential slight degradation of conceptual competence.

SLlama Architecture The SLlama architecture integrates reduction techniques with least parameter count while preserving competitive[6] performance. Specifically, SLlama combines Repeated Reduced Hidden Size and Projection (RRHP), Permutated Weight Attention (PWA), Shared Projection Multi-Layer Perceptron (SPMLP), and Layer Weight Sharing to achieve architectural compactness. Compared to a similar configuration of Llama architecture, SLlama achieves a 40% reduction in parameter count without compromising linguistic competence.

5.2 Comparison with Baselines and other Models

We compared SLlama with Baby Llama (58M, distilled), OPT (125M), RoBERTa (base), T5 (base), Llama2 (58M), GPT-2 (705M) in Figures 2 and 3. All models are trained on the same BabyLM

[6]By competitive, we mean the drop in performance is less than 1.0

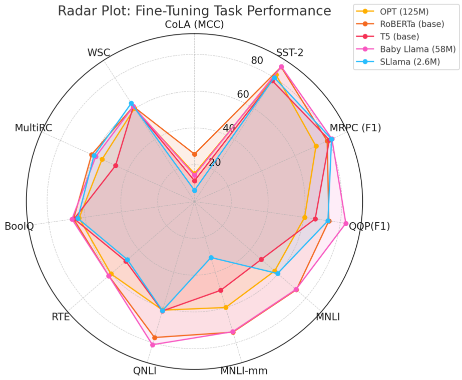

Figure 3: Comparing SLlama with other models on SuperGLUE. All models are trained on the 10M corpus.

challenge dataset. Note the superiority of SLlama architecture over other models in BLiMP tasks maintaining the prowess of the base Llama architecture despite size reduction. Compared to Baby Llama (58M) (Timiryasov and Tastet, 2023), SLlama (2.6M) has around 20× fewer parameters and improves linguistic competence by 31.72% without any knowledge distillation. It also maintains a comparable GLUE score without hyperparameter tuning.

We further evaluated Llama-3.2-3B-Instruct (Grattafiori et al., 2024) on the BLiMP benchmark (Warstadt et al., 2023c) in order to provide a strong baseline for comparison. Interestingly, despite its substantially larger parameter count and extensive pretraining, Llama-3.2-3B-Instruct achieved only 81.79% on BLiMP and 78.82% on the BLiMP supplement. By contrast, our compact 2.6M parameter model reached 91.94% and 77.61%, respectively. This result is striking: it shows that under small-data conditions, carefully designed lightweight models can not only remain competitive with, but in some cases even outperform, much larger instruction-tuned models. Rather than contradicting our central claim, these findings accentuate it: given limited supervision, aligning model capacity to dataset size is more effective than scaling parameters indiscriminately.

SLlama's Strong Generalization Across Core Grammar The BLiMP tasks span syntactic, morphological, semantic, and pragmatic domains. SLlama achieves near-perfect accuracy on core grammatical phenomena such as anaphor agreement, filler-gap dependencies, irregular forms, and quantifier interpretation. It also excels in subject–auxiliary inversion (99.9%) and binding (99.98%). Following the observations of Warstadt et al. (2023c), this performance suggests that SLlama effectively encodes core syntactic dependencies and morphological regularities despite its small size. Such strong generalization indicates that, with targeted architectural reductions, even highly compact models can acquire grammatical competence typically associated with much larger models.

Furthermore, we assess out-of-domain generalization, which provides a more stringent test of model robustness. To this end, we evaluate SLlama, Baby Llama, and other baseline architectures trained on the BabyLM dataset against the MMLU benchmark (Hendrycks et al., 2020). MMLU consists of high school–level examination questions, a domain that lies far outside the scope of the BabyLM training corpus. As shown in Table 3, SLlama demonstrates a notably stronger generalization capacity compared to larger-parameter baselines, reinforcing the view that compact models can achieve competitive performance even under extreme domain transfer.

Improvements Over Comparable Models Compared to Baby LLaMA (58M, distilled) and even LLaMA-360M, SLlama frequently outperforms across categories: 1. Filler-gap: SLlama (100%) > Baby LLaMA (71.8%) > LLaMA-360M (70.6%) 2. NPI licensing: SLlama (99.11%) > LLaMA-360M (57.3%) 3. Island effects: SLlama (99.95%) > LLaMA-360M (50.4%) These suggest that scaling down parameters does not necessarily reduce linguistic competence, and may even improve it when guided by effective architectural design.

6 Discussion

We provide an explanation for the degradation in linguistic performance caused by weight tying and discuss how the employed reduction techniques shed light on language processing dynamics in parameter-efficient architectures.

6.1 Degraded Linguistic Competence with Weight Tying

By weight tying, we refer to the practice of sharing parameters between the input embedding matrix and the language model output head. As demonstrated in Table 1, this technique degrades linguistic competence in small models—a phenomenon warranting further investigation. Notably, the findings

Group	SLlama (2.6M) (%)	Baby Llama (58M)(%)	Baby Llama-2 (345M)(%)	SmolLM135 (135M)(%)
Humanities	0.2339	0.2462	**0.2527**	0.2472
Social Sciences	**0.3063**	0.2213	0.2199	0.2222
STEM	**0.2833**	0.2200	0.2249	0.2191
Other	**0.2559**	0.2420	0.2389	0.2402
MMLU Overall	**0.2698**	0.2324	0.2341	0.2322

Table 3: Performance of SLlama and baseline models on the MMLU benchmark. Scores are reported as accuracy across subject groups and overall. SLlama (2.6M parameters) achieves competitive or superior performance relative to models more than 50× larger, underscoring the efficiency of compact architectures in low-resource regimes. All models are trained on the BabyLM dataset.

of Eldan and Li (2023); Press and Wolf (2017); Mnih and Teh (2012) offer insights that may justify this degradation.

Mnih and Teh (2012) hypothesized that when tying the embedding weights, rows corresponding to semantically similar words should exhibit near-identical representations—such that the input embedding encodes synonyms in a comparable manner, while the output embedding assigns similar score distributions to interchangeable words. Expanding on this, Press and Wolf (2017) empirically demonstrated that tying input and output embeddings produces a joint representation more closely aligned to the output embedding of an untied model.

However, their findings also suggest that untied embeddings evolve into distinct representations. By compressing these distinct roles into a shared space, weight tying limits the model's ability to retain rich input representations essential to linguistic competence.

Furthermore, Eldan and Li (2023) confirmed that the embedding and shallow layers of a model host most linguistic information. Given that the poor performance of tied Llama are pronounced on linguistic evaluation, we conclude that the drop in performance is due to the observation of Press and Wolf (2017); that is, the embedding aligns more to the output layer and has lost salient linguistic information. Thus, empirically, untying embeddings improves performance on linguistic tasks for small language models.

This raises the question: would linguistic performance improve without reducing the hidden size? In practice, no—LLaMA models with a 64×6 configuration and those with larger hidden sizes but tied weights perform similarly, as shown in Table 4.

6.2 Implications of the Reduction Techniques

At the embedding layer, LHRP reveals that linguistic information encoded in the embedding layer cannot be linearly projected into a higher-dimensional space without incurring a loss of critical content. Similarly, even the more expressive attention mechanism fails to reliably upscale linguistic representations without degradation. In contrast, the effectiveness of RRHP suggests that simple repetition, rather than projection, offers a more viable path for preserving and extending learned linguistic representations. Shared Key-Query Attention (SKQA) reframes self-attention as a linguistic operation based on token similarity. It enforces symmetry by sharing the key and query weight matrices. While future work may explore omitting one matrix entirely, such simplifications require careful evaluation. SKQA may also be less effective in asymmetric tasks like machine translation, where source–target distinctions are crucial.

Additionally, while repetition of learned embeddings (as seen in RRHP) has proven effective, our experiments with Reduced Repeated Attention (RRA) demonstrate that modifying the attention-defining neurons—particularly by altering or compressing them—can significantly impair model performance. This highlights a key asymmetry: embedding representations tolerate structural repetition, whereas the attention mechanism is more sensitive to architectural perturbations during language processing.

6.3 The Vicious Cycle

While RRHP in the embedding layer yields the clearest gains in our setting, the effect of other reduction techniques may only emerge at larger depths and widths. Scaling, however, is limited by both data and compute—data balance: as shown by Hoffmann et al. (2022), smaller models trained on more data outperform larger ones trained on less. With only 10M tokens, enlarging the model would lead to under-training, while enlarging the dataset would break comparability with BabyLM. This dilemma motivates our focus on evaluating reduction techniques strictly under BabyLM's data-scarce conditions, where RRHP delivers meaningful improvements. Future work with larger data regimes will be needed to fully assess the other methods.

7 Related Work

As large models like PaLM (Chowdhery et al., 2022) and GPT-3 (Brown et al., 2020) push performance boundaries, their computational demands have prompted interest in data-efficient and compact alternatives. Data efficiency efforts include dataset reduction via k-means clustering (Kaddour, 2023), deduplication (Lee et al., 2022), and high-quality data curation (Mueller and Linzen, 2023; Eldan and Li, 2023; Gunasekar et al., 2023; Huebner et al., 2021), with studies emphasizing the role of data diversity (Lu et al., 2024; Mekala et al., 2024). We build on this by training SLlama under the 10M-token constraint of the BabyLM Challenge (Warstadt et al., 2023b,a; Choshen et al., 2024), highlighting performance under limited data.

Compression techniques such as ROBE (Desai et al., 2022), MEmCom (Pansare et al., 2022), Mixed Dimension Embeddings (Ginart et al., 2021), and Slim Embeddings (Li et al., 2017) have reduced large embedding table sizes. For Transformer models, inter-layer weight sharing and factorized embeddings (Lan et al., 2020) helped reduce BERT's footprint (Devlin et al., 2019). Concurrently, smaller models like OPT (Zhang et al., 2022), Phi (Gunasekar et al., 2023), and PanGu-π (Tang et al., 2024) show that careful architectural design—often overlooked under fixed-compute assumptions (Kaplan et al., 2020)—can yield competitive performance. SLlama continues this trend, introducing novel reductions that preserve linguistic competence.

Weight sharing, though common (Tang et al., 2024; Lan et al., 2020; Ainslie et al., 2023), has uneven effects. While normalized shared embeddings can mitigate performance loss (Liu et al., 2020), we find that tying input-output embeddings degrades linguistic quality. In contrast, sharing attention weights (e.g., key–query) retains expressivity, suggesting that selective weight sharing is key to balancing efficiency and capability.

8 Conclusion

We introduced SLlama, a parameter-efficient adaptation of the LLaMA architecture designed for data- and scale-constrained settings like the BabyLM Challenge. Combining reduction strategies—Repeated Reduced Hidden Size and Projection (RRHP), Permutated Weight Attention (PWA), Shared Projection MLP (SPMLP), and Layer Weight Sharing—we show that small models can achieve strong linguistic performance without relying on embedding weight tying, which we find degrades linguistic competence.

Our findings suggest that repetition-based projections offer a more robust path for preserving linguistic representations than linear expansion or tied embeddings. Moreover, our analysis of SLlama's components offers a deeper understanding of how architectural efficiency and linguistic expressivity interact, revealing design principles that extend beyond scaling.

SLlama contributes both a performant architecture and a conceptual framework for future exploration of efficient language models—particularly in low-resource or edge deployment scenarios.

Limitations

While this study demonstrates promising results, several limitations must be considered. Our findings are primarily based on the LLama architecture, and while certain trends may generalize, further research is needed to assess the applicability of our techniques across diverse model architectures. Additionally, the BabyLM dataset, while useful for studying small-data training, lacks linguistic diversity, limiting the evaluation of our models to English. Future work should explore performance on more diverse datasets, including low-resource languages, and assess the models' ability to acquire commonsense and factual knowledge.

Moreover, real-world deployment challenges remain, particularly regarding performance on edge devices, where quantization-related degradation has yet to be fully examined. The scalability of our compression techniques to larger models and datasets also requires further investigation. Ultimately, striking an optimal balance between model efficiency and linguistic richness is an ongoing challenge, and future research should focus on refining model reduction strategies to ensure robust language representation while maintaining computational efficiency.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *Preprint*, arXiv:2305.13245.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Eric Zelikman Cheng, Eric Zhao, Swaroop Mishra, John Canny, Tianyi Zhang, and James Zou. 2023. Flask: Fine-grained language model evaluation based on alignment skill sets. *arXiv preprint arXiv:2307.10928*.

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *Preprint*, arXiv:2204.02311.

Aditya Desai, Li Chou, and Anshumali Shrivastava. 2022. Random offset block embedding array (robe) for criteotb benchmark mlperf dlrm model : $1000\times$ compression and $3.1\times$ faster inference. *Preprint*, arXiv:2108.02191.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

A.A. Ginart, Maxim Naumov, Dheevatsa Mudigere, Jiyan Yang, and James Zou. 2021. Mixed dimension embeddings with application to memory-efficient recommendation systems. In *2021 IEEE International Symposium on Information Theory (ISIT)*, page 2786–2791. IEEE Press.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal

Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad

Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *Preprint*, arXiv:2306.11644.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

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. *Preprint*, arXiv:2203.15556.

Philip A. Huebner, Elior Sulem, Fisher Cynthia, and Dan Roth. 2021. BabyBERTa: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 624–646, Online. Association for Computational Linguistics.

Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian Paulun, Maria Ryskina, Ekin Akyürek, Ethan Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2024. Elements of world knowledge (ewok): A cognition-inspired framework for evaluating basic world knowledge in language models. *Preprint*, arXiv:2405.09605.

Jean Kaddour. 2023. The minipile challenge for data-efficient language models. *Preprint*, arXiv:2304.08442.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *Preprint*, arXiv:2001.04451.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *Preprint*, arXiv:1909.11942.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445, Dublin, Ireland. Association for Computational Linguistics.

Zhongliang Li, Raymond Kulhanek, Shaojun Wang, Yunxin Zhao, and Shuang Wu. 2017. Slim embedding layers for recurrent neural language models. *Preprint*, arXiv:1711.09873.

Percy Liang, Rishi Bommasani, Tony Lee, et al. 2022. Holistic evaluation of language models. In *Proceedings of NeurIPS 2022 Track on Datasets and Benchmarks*.

Jinyang Liu, Yujia Zhai, and Zizhong Chen. 2020. Normalization of input-output shared embeddings in text generation models. *Preprint*, arXiv:2001.07885.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, Liangzhen Lai, and Vikas Chandra. 2024. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *Preprint*, arXiv:2402.14905.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D. Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *Preprint*, arXiv:2409.15790.

Dheeraj Mekala, Alex Nguyen, and Jingbo Shang. 2024. Smaller language models are capable of selecting instruction-tuning training data for larger language models. *Preprint*, arXiv:2402.10430.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. *Preprint*, arXiv:1206.6426.

Aaron Mueller and Tal Linzen. 2023. How to plant trees in language models: Data and architectural effects on the emergence of syntactic inductive biases. *Preprint*, arXiv:2305.19905.

Niketan Pansare, Jay Katukuri, Aditya Arora, Frank Cipollone, Riyaaz Shaik, Noyan Tokgozoglu, and Chandru Venkataraman. 2022. Learning compressed embeddings for on-device inference. *Preprint*, arXiv:2203.10135.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. *Preprint*, arXiv:1608.05859.

Yehui Tang, Fangcheng Liu, Yunsheng Ni, Yuchuan Tian, Zheyuan Bai, Yi-Qi Hu, Sichao Liu, Shangling Jui, Kai Han, and Yunhe Wang. 2024. Rethinking optimization and architecture for tiny language models. *Preprint*, arXiv:2402.02791.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 279–289, Singapore. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. *Preprint*, arXiv:1804.07461.

Alex Warstadt, Leshem Choshen, Aaron Mueller, Adina Williams, Ethan Wilcox, and Chengxu Zhuang. 2023a. Call for papers – the babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus. *Preprint*, arXiv:2301.11796.

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

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2023c. Blimp: The benchmark of linguistic minimal pairs for english. *Preprint*, arXiv:1912.00582.

Jiale Zhang, Yulun Zhang, Jinjin Gu, Jiahua Dong, Linghe Kong, and Xiaokang Yang. 2024a. Xformer: Hybrid x-shaped transformer for image denoising. *Preprint*, arXiv:2303.06440.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024b. Tinyllama: An open-source small language model. *Preprint*, arXiv:2401.02385.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

A Initial Experiments of Model Sweep

Characterizing the Llama Architecture To isolate the effect of distillation, we conducted experiments to characterize the inherent capabilities of the Llama architecture and to establish the relationship between its key configuration parameters (hidden size, intermediate size, and number of layers) and performance on the aforementioned evaluation tasks. Following the recommendations of Tang et al. (2024), we tie the embedding layer and language model head, a widely used strategy to improve parameter efficiency in small-scale language models. Starting with a hidden size of 64 (to minimize resource consumption), we varied the number of layers from 2 to 12.

We observed that the macro-average scores for models with six and eight layers were similar, as were those for models with ten and twelve layers. Based on this, we focused subsequent experiments on layer counts of 2, 4, 6, and 10, while logarithmically increasing the hidden size from 64 to 1,024. The model with a hidden size of 512 and 2 layers achieved the best average macro score. However, the model with hidden size 64 and 6 layers obtains a competitive macro-averaged score while requiring less time to train and evaluate. In order to minimize computational cost, memory usage, and experimental time, subsequent experiments were based on the latter configuration (hidden size 64 and 6 layers). Finally, to ascertain the plausibility of weight tying, we trained a 64 by 6 model with untied weights.

Characterizing the Llama Architecture We present the results of the experiment to characterize the inherent ability of Llama architecture without distillation in Table 1. We observed that the relationship between macro-averaged scores and model size is not direct. Further analysis presented in Figure 4, shows the correlation between model size parameters (hidden size and number of layers) and the model's performance across the different evaluation dimensions (linguistic competence, world knowledge, and conceptual understanding). While statistical significance was generally weak, several trends emerged: 1) a weak but consistent positive correlation between hidden size and BLiMP score (linguistic knowledge); 2) an inconsistent positive relationship between hidden size and GLUE score; 3) a strong and consistent negative correlation between hidden size and world knowledge; 4) an inconsistent positive trend between the number of layers and linguistic competence; 5) a weak positive trend between the number of layers and conceptual understanding; and 6) a noticeable weak negative trend between the number of layers and linguistic competence. While these observations suggest the need to carefully balance horizontal (hidden size) and vertical (number of layers) scaling, particularly while training on limited data, more data is needed to fully concretize these claims. However, the positive impact of increasing layer count for smaller hidden sizes was evident, supporting previous findings of Liu et al. (2024).

The results in Table 4 influenced our hyperparameter selection.

B Architectural Comparison

We compare different language model architectures and present them in Table 5. All models are trained on the same dataset but for different epochs.

C SuperGLUE scores

We also include the performance of other architecture reported in other studies in Table 6. While SuperGLUE is not our focus in this work, it is noteworthy to demonstrate that the architecture maintains a reasonable degree of conceptual competence relative to the larger models.

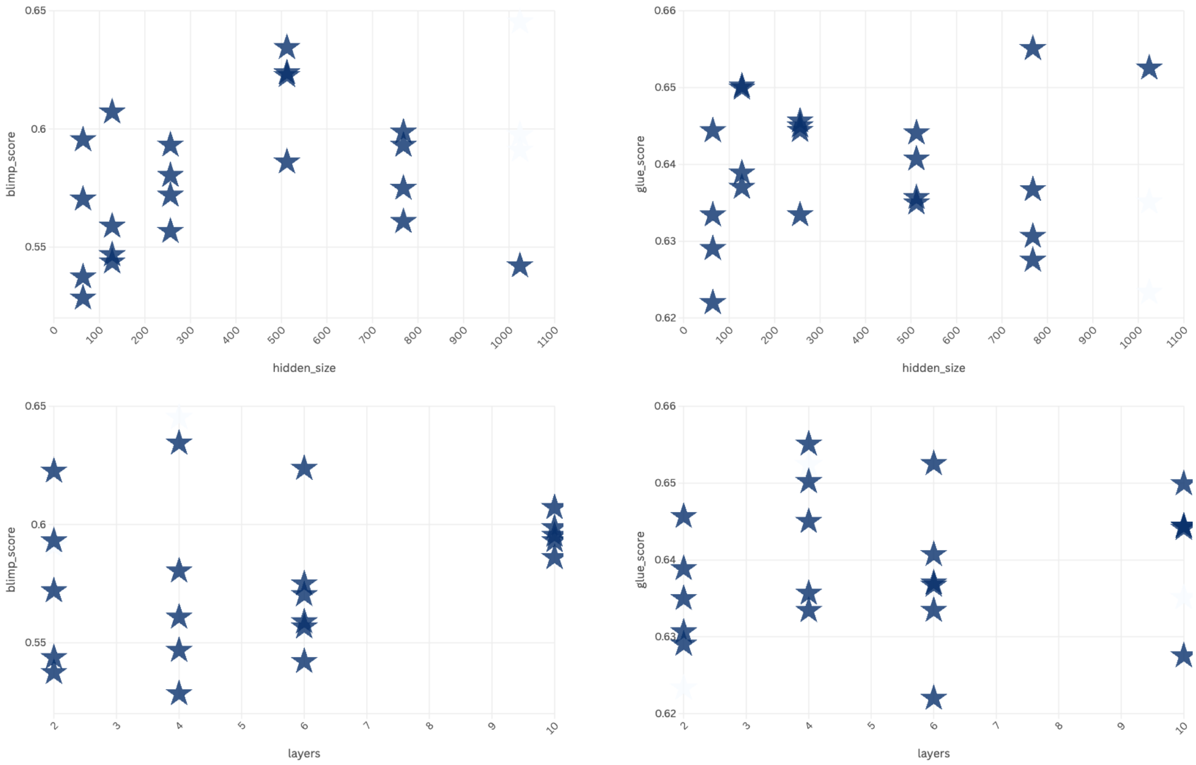

Figure 4: Correlation of hidden size and number of layers to BLiMP and GLUE scores with Spearman correlations of 0.38 and 0.88, respectively.

Model	Hidden Size	Layers	BLiMP	GLUE	EWoK	BLiMP-Sup	Macro Avg.
1	1,024	10	59.75%	63.51%	54.09%	49.45%	56.70%
2	1,024	2	59.10%	62.33%	53.86%	52.31%	56.90%
3	1024	4	**64.53%**	65.24%	53.30%	48.47%	57.89%
4	1,024	6	54.21%	**65.25%**	53.94%	50.87%	56.07%
5	128	10	60.72%	64.99%	56.14%	48.48%	57.58%
6	128	2	54.37%	63.89%	55.96%	48.51%	55.68%
7	128	4	54.68%	65.02%	56.10%	**53.65%**	57.36%
8	128	6	55.89%	63.70%	56.96%	48.48%	56.26%
9	256	10	59.32%	64.44%	55.65%	51.32%	57.68%
10	256	2	57.20%	64.57%	55.13%	**56.06%**	**58.24%**
11	256	4	58.03%	64.50%	55.72%	50.43%	57.17%
12	256	6	55.67%	63.34%	55.96%	49.66%	56.16%
13	512	10	58.60%	64.41%	55.13%	46.85%	56.25%
14	512	2	62.26%	63.50%	55.36%	53.28%	58.60%
15	512	4	63.44%	63.57%	55.73%	48.21%	57.74%
16	512	6	62.37%	64.07%	55.59%	51.58%	58.40%
17	64	10	59.54%	64.44%	**58.01%**	49.77%	57.94%
18	64	2	53.74%	62.90%	57.99%	55.07%	57.42%
19	64	4	52.85%	63.34%	57.91%	48.00%	55.52%
20	64	6	57.03%	62.20%	57.02%	48.71%	56.24%
21	768	10	59.87%	62.75%	54.85%	49.57%	56.76%
22	768	2	59.31%	63.06%	54.48%	54.54%	57.85%
23	768	4	56.08%	**65.51%**	54.07%	53.49%	57.29%
24	768	6	57.49%	63.67%	53.74%	53.18%	57.02%

Table 4: Evaluation scores across models with varying hidden sizes and number of layers. Best values per metric are in bold.

Task	OPT (125M)	RoBERTa (base)	T5 (base)	LLaMA2 (58M)	LLaMA2 (360M)	GPT-2 (705M)	BabyLlama (58M)	SLlama (2.6M)
Anaphor Agr.	63.80	81.50	68.90	87.00	87.60	89.60	89.80	**100.00**
Arg. Structure	70.60	67.10	63.80	72.30	73.50	73.50	73.10	**74.98**
Binding	67.10	67.30	60.40	71.20	72.10	71.50	72.70	**99.98**
Control/Raising	66.50	67.90	60.90	67.50	67.40	68.40	67.50	**80.11**
Det.-Noun Agr.	78.50	90.80	72.20	87.80	89.60	87.40	90.80	**95.03**
Ellipsis	62.00	76.40	34.40	67.30	68.50	69.90	73.30	**73.13**
Filler-Gap	63.80	63.50	48.20	70.90	70.60	70.20	71.80	**100.00**
Irregular Forms	67.50	87.40	77.60	74.10	68.90	83.10	93.10	**100.00**
Island Effects	48.60	39.90	45.60	57.30	50.40	51.60	51.20	**99.95**
NPI Licensing	46.70	55.90	47.80	51.10	57.30	50.50	56.50	**99.11**
Quantifiers	59.60	70.50	61.20	64.20	59.00	69.80	73.30	**100.00**
Subj.-Verb Agr.	56.90	65.40	65.00	73.00	69.70	67.50	75.40	**87.29**
Hypernym	50.00	49.40	48.00	48.70	49.40	49.20	49.30	**79.45**
QA Congr. (easy)	54.70	31.30	40.60	50.00	53.10	56.20	51.60	**57.81**
QA Congr. (tricky)	31.50	32.10	21.20	32.70	41.80	45.50	41.80	**50.91**
Subj.-Aux. Inversion	80.30	71.70	64.90	77.40	84.30	81.70	88.50	**99.90**
Turn Taking	57.10	53.20	45.00	63.90	68.60	65.70	66.10	**100.00**

Table 5: Comparative performance of SLlama and larger models on BLiMP tasks. Results for baseline models (OPT, RoBERTa, T5, LLaMA, GPT-2, and Baby LLaMA) are taken from the original baseline paper. All models, including SLlama, are trained on the same 10M-token dataset.

Task	OPT (125M)	RoBERTa (base)	T5 (base)	Baby Llama (58M)	SLlama (2.6M)
CoLA (MCC)	15.2	**25.8**	11.3	14.3	6.1
SST-2	81.9	87.0	78.1	**87.2**	80.1
MRPC (F1)	72.5	79.2	80.5	**82.0**	81.8
QQP(F1)	60.4	73.7	66.2	**83.0**	73.2
MNLI	57.6	**73.2**	48.0	72.9	59.7
MNLI-mm	60.0	**74.0**	50.3	73.7	31.7
QNLI	61.5	77.0	62.0	**81.1**	61.8
RTE	60.0	**61.6**	49.4	**61.6**	48.2
BoolQ	63.3	66.3	66.0	**67.2**	64.0
MultiRC	55.2	**61.4**	47.1	58.9	60.0
WSC	60.2	61.4	61.4	61.4	**63.5**

Table 6: Evaluation results on SuperGLUE. The reported scores are accuracy values except when specified otherwise. All models are pretrained on the training dataset.

Parameter	**Value**
gradient_accumulation_steps	2
batch_size	128
block_size	block_size
dropout	0.1
learning_rate	4e-4
max_iters	3000
weight_decay	0.0
warmup_iters	200
lr_decay_iters	5000
min_lr	4e-5
train_or_dev	train

Table 7: Training configuration for SLlama experiments