BABYHGRN: Exploring RNNs for Sample-Efficient Training of Language Models

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

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

Abstract

This paper explores the potential of recurrent neural networks (RNNs) and other subquadratic architectures as competitive alternatives to transformer-based models in low-resource language modeling scenarios. We utilize HGRN2 (Qin et al., 2024), a recently proposed RNN-based architecture, and comparatively evaluate its effectiveness against transformer-based baselines and other subquadratic architectures (LSTM, xLSTM, Mamba). Our experimental results show that BABYHGRN, our HGRN2 language model, outperforms transformer-based models in both the 10M and 100M word tracks of the challenge, as measured by their performance on the BLiMP, EWoK, GLUE and BEAR benchmarks. Further, we show the positive impact of knowledge distillation. Our findings challenge the prevailing focus on transformer architectures and indicate the viability of RNN-based models, particularly in resource-constrained environments.

1 Introduction

In recent years, natural language processing (NLP) has been revolutionized by transformer-based language models (LMs), like BERT (Devlin et al., 2019) or GPT (Brown et al., 2020) and their derivatives, achieving state-of-the-art results (Touvron et al., 2023; Abdin et al., 2024) across a wide range of tasks such as machine translation, question answering, and text generation. However, despite their dominance, transformers come with notable limitations: they require extensive training data (Hoffmann et al., 2022) and enormous computational resources, which pose challenges for their use in resource-constrained environments.

These limitations led to an increasing interest in more sample-efficient alternatives and approaches with lower computational requirements (Wang et al., 2020b). The shared task of the BabyLM Challenge (Warstadt et al., 2023a) systematically explores this trend by training LMs on datasets of limited size (10M words in the "strict-small" and 100M words in the "strict" setup). The resulting models are then evaluated on linguistic and general language understanding tasks.

While most participants in BabyLM Challenge focus on adapting transformers to low-resource settings, we propose revisiting recurrent neural networks (RNNs). Once foundational to sequence modeling tasks (Lample et al., 2016; Howard and Ruder, 2018), RNNs have been largely overshadowed by transformers due to their sequential nature which does not easily allow for parallelization.

Potential of RNN-architectures. In this paper, we investigate whether the inductive biases of RNN architectures, such as their sequential processing and memory states, provide advantages in data-constrained settings. This question is especially relevant given that state-of-the-art transformer models depend on quadratic self-attention, which requires calculating the inner product between all tokens. In particular, we investigate the potential of the HGRN2 (Qin et al., 2024), a novel subquadratic RNN-based architecture based on hierarchical gating. We train our model using knowledge distillation (Hinton et al., 2015) and evaluate our approach, BABYHGRN, against state-of-the-art transformer models and other efficient RNN architectures (e.g. xLSTM (Beck et al., 2024) or Mamba (Gu and Dao, 2024)). Our experiments demonstrate that our resulting model yields better performance compared to both transformer-based and other RNN-based architectures.

We summarize our contributions as follows:

1. We conduct an exploratory evaluation of transformer-based and other RNN-based architectures (HGRN2, LSTM, xLSTM, Mamba), contributing to the ongoing research on sample-efficient language modeling.

2. We present a comprehensive evaluation of our

Dataset	Count	Ratio (%)	Dataset	Count	Ratio (%)
Pile-CC	4,900,155	49.00	Pile-CC	49,214,555	49.21
OpenWebText2	3,078,791	30.79	OpenWebText2	30,344,790	30.34
FreeLaw	946,382	9.46	FreeLaw	9,471,436	9.47
USPTO Backgrounds	261,159	2.61	USPTO Backgrounds	2,519,390	2.52
Wikipedia (en)	187,094	1.87	Wikipedia (en)	1,855,709	1.86
PubMed Central	142,698	1.43	PubMed Central	1,449,273	1.45
PubMed Abstracts	118,427	1.18	PubMed Abstracts	1,175,838	1.18
Others	365,188	3.65	Others	3,968,870	3.97
Total	9,999,894		Total	99,999,861	

Table 1: Composition of the **10M** (left table) and **100M** (right table) word datasets (word counts and ratio per domain) we created from the PILE to train BABYHGRN.

proposed HGRN2 language model BABYH-GRN. We show the impact of knowledge distillation and the choice of dataset.

3. We release all code, datasets, and experimental setups to the research community to facilitate reproducibility and further research[1].

Our results show that BABYHGRN outperforms transformer-based baselines on both tracks of the BabyLM challenge.

2 BABYHGRN

We utilize HGRN2 as our backbone architecture with a hidden size of 2048 and 18 layers, resulting in a total parameter count of 330M. We train our model either with (*1*) the default dataset of the BabyLM Challenge or (*2*) a sub-sampled split of ThePile (Gao et al., 2020). Further, we employ knowledge distillation training using a teacher-student setup. In the following, we will discuss the details of our design choices.

2.1 Training Dataset

We curate our own training datasets for the `strict` and `strict-small` tracks by sub-sampling the Pile dataset (see Table 1). The Pile consists of 22 smaller datasets that cover a variety of domains, including books, web pages, scientific literature, and programming code. The main motivation behind choosing the Pile dataset is its diverse composition, which may offer several advantages for language model training. Approximately 14% of the original BabyLM dataset consists of child-related text (e.g., the Children's Book Test (Hill et al., 2016),

Children's Stories Text Corpus[2], and CHILDES project (Macwhinney, 2000)), which may limit its generalizability across diverse domains. In contrast, the broader scope of the Pile dataset could improve resilience in zero-shot tasks and potentially enhance adaptability for fine-tuning on specific areas of interest.

We create the splits by randomly sampling from each chosen subset until we reached the pre-defined thresholds. We depict details on our selected subsets and corresponding word counts in Table 1.

To minimize computational overhead, we concatenate all samples and segment them into uniform chunks of 512 tokens. Subsequently, each input sample is tokenized using Byte-Pair Encoding (BPE), employing a vocabulary size of 16,000 tokens. We chose the *BabyLlama* tokenizer provided with the baseline models by the organizers[3].

2.2 Training Objectives

We use standard next-token prediction as the language modeling task and employ token-level cross-entropy loss for training our models. For a sequence of tokens $x = (x_1, ..., x_N)$, the loss is calculated as:

$$\mathcal{L}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \log P(x_i | x_1, ..., x_{i-1}; \theta)$$

where θ represents the model parameters and $P(x_i \mid x_1, ..., x_{i-1}; \theta)$ is the probability the model assigns to the i-th token given all previous tokens.

We further improve our model through knowledge distillation (Bucila et al., 2006; Hinton et al.,

---

[1]https://github.com/HallerPatrick/BabyLM-2024

[2]https://www.kaggle.com/datasets/edenbd/children-stories-text-corpus

[3]https://huggingface.co/babylm/babyllama-100m-2024

2015), where we train a second HGRN2 model (student) using predictions from our initially trained model (teacher). While knowledge distillation traditionally transfers knowledge from larger to smaller models, using same-sized teacher and student models has proven effective in recent work - notably in the previous BabyLM Challenge where an ensemble of teachers was used for knowledge transfer (Timiryasov and Tastet, 2023).

The training process for the student model incorporates an additional loss term based on soft labels produced by the teacher model. The total loss function for the student model can be expressed as:

$$\mathcal{L}_{\text{total}} = (1 - \alpha)\mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{KD}}$$

where $\mathcal{L}_{\text{CE}}$ is the standard cross-entropy loss for the student model, $\mathcal{L}_{\text{KD}}$ is the knowledge distillation loss, and α is a hyperparameter that balances the two loss terms.

In our implementation, the knowledge distillation loss $\mathcal{L}_{\text{KD}}$ is calculated using the Kullback-Leibler divergence between the probability distributions of the teacher and student models:

$$\mathcal{L}_{\text{KD}} = \text{KL}(\sigma(z_t)||\sigma(z_s))$$

where z_t and z_s are the output logits of the teacher and student model respectively. And $\sigma(z)$ is the softmax function applied to the logits z.

2.3 Training Details

For fine-tuning on the (Super)Glue tasks, we follow the provided hyperparameters by the shared task organizer (see Appendix A). Except for the *WSC* tasks, which had unusually low scores. We used a maximum of 20 epochs, a patience of 6 epochs and a learning rate of 1×10^{-5} for our final submission models.

Software. For training our model we use the `Pytorch` (Ansel et al., 2024) library. Relevant metrics are logged with `Weights and Biases` (Biewald, 2020). We use Hugging-Face `datasets` (Lhoest et al., 2021) library for dataset loading and subsampling. All relevant models were either directly imported with the `transformers` (Wolf et al., 2020) library or implemented as a custom model. For the HGRN2 model we used the FLA (Yang and Zhang, 2024) library.

Hardware. All models were trained with the `torch.distributed` package in data-parallel mode. Models were trained on 4 RTX A6000 49GB graphics cards on one node.

3 Empirical Evaluation

In Section 3.1, we shortly present the evaluation benchmarks of the BabyLM Challenge and the BEAR knowledge probe. In Sections 3.2 to 3.4, we evaluate BABYHGRN compared with other efficient RNN architectures, its training dynamics, and the influence of different datasets. Finally, in Section 3.5, we evaluate BABYHGRN using knowledge distillation.

3.1 Evaluation Datasets

The BabyLM challenge covers three benchmarks: BLiMP (Warstadt et al., 2023b), EWoK (Ivanova et al., 2024), and parts of GLUE (Wang et al., 2019) and SuperGLUE (Wang et al., 2020a), respectively. These benchmarks are designed to assess language model performance such as grammatical knowledge or complex reasoning tasks. Additionally, we include the BEAR probe (Wiland et al., 2024) to evaluate factual knowledge capabilities.

BLiMP (Benchmark of Linguistic Minimal Pairs) is an English zero-shot benchmark evaluating the grammatical knowledge of language models. It has 67 sub-tasks, each focusing on a specific syntactic or semantic phenomenon. Specifically, the dataset contains pairs of sentences and the model is tasked to differentiate which of the sentences is grammatically correct. Further, we consider the hidden task *"BLiMP Supplement"* of the 2023 BabyLM Challenge (Warstadt et al., 2023a).

EWoK (Elements of World Knowledge) evaluates basic world knowledge in language models. This cognition-inspired approach tests whether language models can identify plausible contexts given different fillers. EWoK was introduced as the hidden task for the 2024 BabyLM Challenge.

GLUE (General Language Understanding Evaluation) is a multi-task benchmark evaluating natural language understanding systems. It contains nine tasks such as sentiment analysis, question answering, or textual entailment. As models began to surpass human performance on several GLUE tasks, SuperGLUE was introduced as an extension, including more challenging tasks.

BEAR (Wiland et al., 2024) tests relational knowledge in language models using 7,731 instances over 60 relations. BEAR compares the models' log-likelihood for different factual statements of which only one is true. We leverage the implementation by Ploner et al. (2024) to conduct the BEAR probing experiments.

Model	#Params	Epoch	BLiMP	BLiMP-Supp.	EWoK	Macro-Avg.
Transformer	360M	4	62.64	54.86	50.48	55.99
LSTM	300M	5	62.27	51.63	50.48	54.79
xLSTM	340M	3	51.20	48.77	49.89	50.02
Mamba	350M	2	64.44	55.39	50.39	56.74
HGRN2	360M	4	67.05	55.69	49.88	**57.54**

Table 2: Results from training on the 10M word corpus, comparing various RNN architectures to a Transformer-based model (LLaMA architecture). Each model was trained for 5 epochs, with evaluations after each epoch, and the best-performing model was selected.

Hyperparameter	Value
Epochs	3
Batch Size	64
Learning Rates	{1e-3, 1e-4, 1e-5, 1e-6}
Optimizer	Adam
Sequence Length	512
Max Grad Norm	1.0
LR Scheduler	Linear

Table 3: Pretraining hyperparameters used for all models and experiments.

3.2 Experiment 1: RNN Architecture Selection

Our first experiment compares the HGRN architecture with other RNN-based and transformer architectures. Specifically, we compare HGRN2, the vanilla LSTM, xLSTM, Mamba, and a Transformer baseline.

Experimental setup. We select configurations such that all architectures have a similar parameter count of 300 to 360 million. We use the configurations as as originally proposed for xLSTM, Mamba, and HGRN2. For the decoder-only transformer, we use the LLaMA (Touvron et al., 2023) model and follow the Pythia (Biderman et al., 2023) 410M model configuration with 22 hidden layers. For the vanilla LSTM, we set the hidden size to 4096 with two layers to match the parameter count of the other architectures. We refer to Appendix B for a detailed overview of all configurations.

For each architecture, we perform learning rate selection for all considered architectures by executing a grid search over commonly used learning rates ({1e-3, 1e-4, 1e-5, 1e-6}). We train each model for 5 epochs on the `strict-small` dataset of the BabyLM challenge. Further, we do not employ any knowledge distillation and train

all LMs using the next-token prediction objective. We report results on the zero-shot benchmarks of BabyLM, namely BLiMP and EWoK, together with their best hyperparameter configuration.

Results. Table 2 shows the number of parameters of each considered architecture and the results achieved during the exploration phase on the zero-shot benchmarks[4]. We find that the HGRN2 exhibits the best performance, closely followed by Mamba. Both outperform the transformer model, suggesting that these architectures offer advantages in low-resource scenarios. The standard LSTM, serving as a baseline for classical RNN architectures, performs worse than the transformer but better than the xLSTM[5]. Further, we observe that all architectures perform best using a learning rate of $1e^{-3}$.

3.3 Experiment 2: Learning Dynamics of HGRN2

To better understand the learning dynamics of the selected HGRN2 architecture, we investigated how its zero-shot performance on the BabyLM benchmark changes over the epochs during training.

Experimental setup. We re-use the best performing hyperparameters from Section 3.2. After each epoch, we evaluate on BLiMP, BLiMP Supp. and EWoK.

Results. The results of this experiment are illustrated in Figure 1. Our analysis reveals early peaks in performance on BLiMP and EWoK and a later peak on BLiMP Supplement. This finding indicates that HRGN2 initially captures certain linguistic patterns from the limited training data, although the

---

[4]We report the complete results of the parameter sweep in Appendix C.

[5]We note considerable training instabilities during pretraining of the xLSTM model, likely due to discrepancies in model architecture or training setup from the original implementation, which may have impacted performance in our low-resource setting.
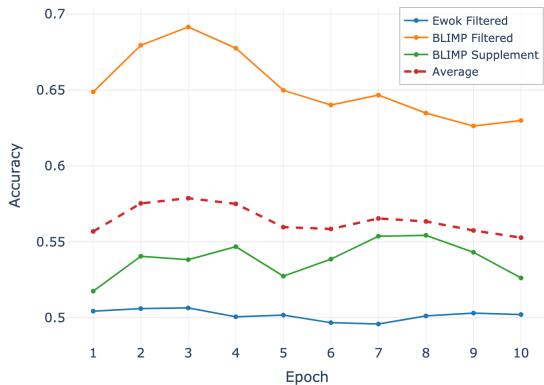
Figure 1: Performance evaluation of epochs of pretraining, with the macro average at epoch 3 being the highest.

Dataset	BLiMP	BLiMP-Supp.	EWoK	BEAR
BabyLM - 10M	67.05	55.69	49.88	5.29
Ours - 10M	67.49	52.22	50.62	5.36
BabyLM - 100M	69.44	55.56	50.31	6.17
Ours - 100M	**72.89**	**57.43**	50.61	**7.38**

Table 4: Zero-shot evaluation results comparing HGRN2 models trained on the BabyLM dataset versus our proposed Pile subset. Both models were trained with a learning rate of 1×10^{-3}. All metrics are reported as percentages.

gains over random baseline are modest. Further iterations yield only incremental improvements, which may point to constraints in the model's ability to leverage the available data fully.

3.4 Experiment 3: Impact of Training Dataset

In this experiment, we evaluate the impact of the choice of training data. We compare models trained over the default BabyLM dataset to models trained using our custom dataset derived from the Pile (see section 2.1).

Experimental Setup. We re-use our chosen hyperparameter configuration for the HGRN2 architecture from Section 3.3 and train two models on (*1*) our derived Pile subset and (*2*) on the default BabyLM dataset. We train models for 5 epochs, evaluate after each epoch, and report results of the best performing model. In this experiment we include both the 10M and 100M word datasets for a full comparison.

Results. Table 4 summarizes the performance across all benchmarks. For the 10M word track, the HGRN2 model trained on our derived dataset shows modest gains on BLiMP, EWoK, and BEAR, but underperforms on the BLiMP-Supplemental subset (↓3.47 pp). This suggests that at smaller data scales, our dataset may lack certain syntactic structures present in the original BabyLM dataset. Furthermore, given the limited dataset size in the 10M word track, these numbers may lack statistical significance.

In contrast, the 100M word track demonstrates consistently stronger performance across all metrics, with particularly notable improvements on

BLiMP (↑3.45 pp) and BEAR (↑1.21 pp). Indicating that our dataset selection strategy enhances the model's ability to acquire both syntactic and factual knowledge when given sufficient training data.

3.5 Experiment 4: BABYHGRN With Knowledge Distillation

Based on the exploratory experiments of the previous subsections, we selected the HGRN2 model trained on our proposed dataset for the BabyLM challenge. We furthermore apply knowledge distillation as outlined in Section 2.2 to our final model. We refer to this model as BABYHGRN.

In this section, we evaluate BABYHGRN using knowledge distillation learning and compare it with two baselines (BabyLlama and LTG-BERT) and a BABYHGRN version using only the *next-token prediction* objective. We denote this ablation model as BabyHGRN$_{ntp}$.

Hyperparameters. We increase the model size in accordance with scaling laws for language models (Kaplan et al., 2020) from 360M to 1.0B. We reduce the learning rate from 1×10^{-3} to 4×10^{-4} accordingly, following the configuration found in Sections 3.2 and 3.3. Empirical work (Kaplan et al., 2020; Hoffmann et al., 2022) suggests that lower learning rates in larger models help mitigate instabilities during training, promoting smoother convergence and more efficient use of computational resources.

3.5.1 Results

Table 5 and Table 6 summarize our experimental results for the 10M and 100M word tracks, respectively.

HGRN2 outperforms baselines. Most importantly, we find that our HGRN2 models show competitive performance across both the 10M and 100M word tracks of the BabyLM challenge. On the 10M words track, BabyHGRN achieves an overall macro average of 63.3% (↑2.5 pp vs. BabyL-

	BLiMP	BLiMP-Supp.	EWoK	SuperGLUE	Average	BEAR
BabyLlama	69.8	59.5	50.7	63.3	60.8	5.4
LTG-BERT	60.6	60.8	48.9	60.3	57.7	5.7
BabyHGRN$_{ntp}$ (ours)	69.4	55.6	50.7	63.0	59.7	5.6
BabyHGRN (ours)	72.1	58.6	51.3	65.8	**63.3**	7.5

Table 5: Evaluation results for the **10M words** track ("strict-small"). The BabyLM score is computed as a macro average over four datasets (BLiMP, BLiMP Supp., EWoK and SuperGLUE) but note that the macro average may not be a representative overall score for each model, since the datasets are of widely varying size (e.g. the BLiMP supplements is only 7% in size compared to the BLiMP). We additionally include the BEAR score for comparison and evaluation of factual knowledge.

	BLiMP	BLiMP-Supp.	EWoK	SuperGLUE	Average	BEAR
BabyLlama	73.1	60.6	52.1	69.0	63.7	8.5
LTG-BERT	69.2	66.5	51.9	68.4	64.0	8.2
BabyHGRN$_{ntp}$ (ours)	74.5	59.1	52.88	69.1	63.9	13.5
BabyHGRN (ours)	77.5	58.5	51.6	70.7	**64.9**	13.6

Table 6: Evaluation results for the **100M words** track ("strict"). The BabyLM score is computed as a macro average over four datasets (BLiMP, BLiMP Supp., EWoK and SuperGLUE). We additionally include the BEAR score for comparison and evaluation of factual knowledge.

lama). As Table 5 shows, BabyHGRN particularly outperforms the baselines on the BLiMP (↑2.4 pp vs. BabyLlama) and SuperGLUE (↑2.5 pp vs. BabyLlama) tasks, and significantly improves the BEAR score (↑1.8 pp vs. LTG-BERT).

On the 100M words track (refer to Table 6), BabyHGRN outperforms the baselines with a marco average of 64.9% (↑0.9 pp vs. LTG-BERT), though the improvement is not as pronounced as in the more data-constrained 10M scenario. Here, BabyHGRN improves in particular the BLiMP (↑4.4 pp vs. LTG-BERT) and SuperGLUE (↑1.7 pp vs. BabyLlama) tasks, but falls short on BLiMP-Supplement (↓7.4 pp vs. LTG-BERT)[6].

Knowledge distillation is helpful. We also note that our knowledge distillation approach significantly improves performance of BABYH-GRN, compared to the distillation-free approach BabyHGRN$_{ntp}$. As Tables 5 and 6 show, Baby-HGRN outperforms both, BabyLlama and LGT-BERT, baselines. Further, we observe BABYH-GRN outperforms BabyHGRN$_{ntp}$ by 5.3 pp on average in the data-constrained 10M setting, confirming the usefulness of distillation losses in such settings.

BABYHGRN is better at learning factual knowledge. While the accuracy on BEAR is relatively low across all settings (compared to state-of-the-art models such as LLaMA-3 with 68.6), we observe that BABYHGRN strongly outperforms transformer-based baselines in data-restricted settings. For instance, BEAR shows a pronounced difference between BabyHGRN and BabyHGRN$_{ntp}$ on the 10M track, and a large difference between the HGRN models and the baselines on the 100M track. We primarily attribute this improvement to the use of our custom dataset.

4 Related Work

In recent years, there has been a resurgence of interest in recurrent neural network (RNN) architectures for sequence modeling, particularly in the context of large language models (LLMs). This renewed focus has led to the development of several RNN-based architectures that aim to combine the efficiency of recurrent models with the expressiveness of more complex architectures like transformers.

HGRN and HGRN2 The Hierarchically Gated Recurrent Neural Network (HGRN) (Qin et al., 2023) introduces a novel gating mechanism that allows for more effective modeling of long-term dependencies. The key innovation of HGRN is

---

[6]Detailed results for BLiMP, BLiMP-Supplement, EWoK and (Super)Glue are provided in Appendix D.

its hierarchical structure, in which forget gates have monotonically increasing lower bound values from bottom layers to upper layers. This design enables lower layers to model short-term dependencies while upper layers capture long-term relationships in the data. HGRN achieves efficient training by reformulating its recurrent computation as a parallel scan operation to enable parallelization across sequence length while maintaining linear time complexity.

Building upon HGRN, Qin et al. (2024) introduced HGRN2 which further enhances the capabilities of gated linear RNNs. HGRN2 addresses some limitations of its predecessor by incorporating a state expansion mechanism. This innovation significantly increases the recurrent state size without introducing additional parameters, leading to improved expressiveness.

xLSTM Another recently proposed RNN-based architecture is the Extended Long Short-Term Memory (xLSTM) (Beck et al., 2024). xLSTM builds upon the classical LSTM (Hochreiter and Schmidhuber, 1997) by introducing two key modifications: exponential gating and modified memory structures. The exponential gating mechanism allows the model to revise storage decisions more effectively, addressing a key limitation of traditional LSTMs. xLSTM introduces two variants: sLSTM with a scalar memory and new memory mixing technique, and mLSTM with a matrix memory and covariance update rule, which is fully parallelizable. The xLSTM approach demonstrates strong performance across various modalities, including language, vision (Alkin et al., 2024; Chen et al., 2024), and audio (Yadav et al., 2024), while maintaining linear scaling in sequence length and efficient inference.

The **Mamba** architecture (Gu and Dao, 2024) improves on state space models (SSMs) by introducing selective state spaces. Building on structured SSMs (Gu et al., 2022), Mamba achieves linear-time sequence processing through input-dependent SSM parameters, enabling selective information propagation across sequences. This mechanism is conceptually similar to gating in classical RNNs (Hochreiter and Schmidhuber, 1997) while maintaining modern computational benefits. The architecture consists of repeated blocks that combine selective SSMs with feed-forward components, in contrast to more complex predecessors

like H3 (Fu et al., 2023) and Hyena (Poli et al., 2023). Though attention-free, Mamba matches or exceeds Transformer performance (Vaswani et al., 2023) across various domains. Its recurrent computation pattern eliminates the need for attention caches during inference, leading to 5× faster inference compared to similar-sized Transformers. This combination of linear scaling and efficiency, without sacrificing model quality, makes Mamba a significant development in sequence modeling.

The development of HGRN2, xLSTM, and Mamba is part of a broader trend in revisiting and improving RNN architectures (Peng et al., 2023; Sun et al., 2023).

5 Conclusion

We presented BabyHGRN, an RNN-based language model that utilizes the HGRN2 architecture. Our experimental results on the evaluation datasets of the BabyLM Challenge and the BEAR probe indicate that BabyHRGN is competitive. Indeed, despite relatively little hyperparameter optimization, our approach significantly outperforms strong transformer-based baselines on the evaluation datasets.

Revisiting our research question posed in Section 1, we conclude that RNN-based language models are indeed competitive in low-resource language modeling scenarios. Based on these results, we believe that advanced RNN-based architectures such as HGRN and Mamba may hold promise for research in sample-efficient language modeling. Accordingly, future work could explore further optimizations of the underlying RNN architectures, investigate their performance on a broader range of tasks, and examine their scalability to larger datasets and model sizes.

Limitations

Our experiments with HGRN2 in the BabyLM Challenge demonstrate the competitiveness of RNN-based models with transformers in low-resource scenarios. However, while we find our results to be promising, it's important to acknowledge that there are several avenues for optimization that we have yet to explore:

Dataset sampling The dataset we used to train BabyHGRN was produced using a naive random sampling of the PILE dataset. More sophisticated approaches, such as importance sampling specialized for downstream tasks, would likely yield better results, especially if optimized for the tasks BabyLM evaluates on. In our work, we refrained from such "dataset engineering" and focused solely on a comparison of different RNN architectures.

Model configurations We utilized the configurations provided by the authors of HGRN2 and xLSTM. Further experimentation with different architectures and hyperparameters for the low-resource scenario could well lead to improved performance of these models.

Context length Optimizing the context length for our specific tasks and data could potentially enhance the model's capabilities. Work from previous years challenge (Edman and Bylinina, 2023; Cheng et al., 2023) suggests that a smaller context size improves performance on all benchmarks.

Knowledge distillation As previously discussed, we only implemented a basic knowledge distillation approach to train BabyHGRN. More sophisticated techniques, such as those employed by Timiryasov and Tastet (2023) could further boost performance.

Our work thus serves as a proof of concept, demonstrating that RNNs can be competitive with transformers in this domain, while leaving room for further advancements.

Acknowledgements

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Benedikt Alkin, Maximilian Beck, Korbinian Pöppel, Sepp Hochreiter, and Johannes Brandstetter. 2024. Vision-lstm: xlstm as generic vision backbone. *arXiv preprint arXiv:2406.04303*.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian

Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. 2024. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ASPLOS '24, page 929–947, New York, NY, USA. Association for Computing Machinery.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2024. xlstm: Extended long short-term memory. *Preprint*, arXiv:2405.04517.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. *Preprint*, arXiv:2304.01373.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

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.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. volume 2006, pages 535–541.

Tianrun Chen, Chaotao Ding, Lanyun Zhu, Tao Xu, Deyi Ji, Yan Wang, Ying Zang, and Zejian Li. 2024. xlstm-unet can be an effective 2d 3d medical image segmentation backbone with vision-lstm (vil) better than its mamba counterpart. *Preprint*, arXiv:2407.01530.

Ziling Cheng, Rahul Aralikatte, Ian Porada, Cesare Spinoso-Di Piano, and Jackie CK Cheung. 2023. McGill BabyLM shared task submission: The effects of data formatting and structural biases. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*,

pages 207–220, Singapore. Association for Computational Linguistics.

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.

Lukas Edman and Lisa Bylinina. 2023. Too much information: Keeping training simple for BabyLMs. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 89–97, Singapore. Association for Computational Linguistics.

Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. 2023. Hungry hungry hippos: Towards language modeling with state space models. *Preprint*, arXiv:2212.14052.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. *Preprint*, arXiv:2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. *Preprint*, arXiv:2111.00396.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. *Preprint*, arXiv:1511.02301.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. 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.

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.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brian Macwhinney. 2000. The childes project: tools for analyzing talk. *Child Language Teaching and Therapy*, 8.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023. Rwkv: Reinventing rnns for the transformer era. *Preprint*, arXiv:2305.13048.

Max Ploner, Jacek Wiland, Sebastian Pohl, and Alan Akbik. 2024. Lm-pub-quiz: A comprehensive framework for zero-shot evaluation of relational knowledge in language models. *Preprint*, arXiv:2408.15729.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. 2023. Hyena hierarchy: Towards larger convolutional language models. *Preprint*, arXiv:2302.10866.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. 2024. Hgrn2: Gated linear rnns with state expansion. *Preprint*, arXiv:2404.07904.

Zhen Qin, Songlin Yang, and Yiran Zhong. 2023. Hierarchically gated recurrent neural network for sequence modeling. *Preprint*, arXiv:2311.04823.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. 2023. Retentive network: A successor to transformer for large language models. *Preprint*, arXiv:2307.08621.

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.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2020a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Preprint*, arXiv:1905.00537.

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.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020b. Linformer: Self-attention with linear complexity. *Preprint*, arXiv:2006.04768.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell, editors. 2023a. *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Singapore.

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

Jacek Wiland, Max Ploner, and Alan Akbik. 2024. BEAR: A unified framework for evaluating relational knowledge in causal and masked language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2393–2411, Mexico City, Mexico. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sarthak Yadav, Sergios Theodoridis, and Zheng-Hua Tan. 2024. Audio xlstms: Learning self-supervised audio representations with xlstms. *Preprint*, arXiv:2408.16568.

Songlin Yang and Yu Zhang. 2024. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism.

A   Finetune Hyperparameters

Hyperparameter	Value
Initial learning rate	5e-5
Batch size	64
Maximum epochs	10
Evaluate every (epochs)	1
Patience	3

Figure 2: Default hyperparameters for fine-tuning on the (Super)Glue tasks.

B   Model Configurations

Transformer	Value
Hidden Size	1024
Intermediate Size	4096
Hidden Layers	22
Attention Heads	32
LSTM	**Value**
Hidden Size	9120
Embedding Size	512
LSTM Layers	2
Dropout	0.1
xLSTM	**Value**
Embedding Size	1024
Num Blocks	48
mLSTM Heads	4
sLSTM Heads	4
sLSTM BLocks at	[3, 5, 7, 40, 42, 44]
Mamba	**Value**
Hidden Size	1024
Intermediate Size	2048
Hidden Layers	48
State Size	8
HGRN2 - 360M	**Value**
Hidden Size	1024
Layers	26
Hidden Ratio	4
Expand Ratio	128
HGRN2 - 1.2B	**Value**
Hidden Size	2048
Layers	18
Hidden Ratio	4
Expand Ratio	128

Table 7: Complete list of model configurations.

C Learning Rate Parameter Sweep
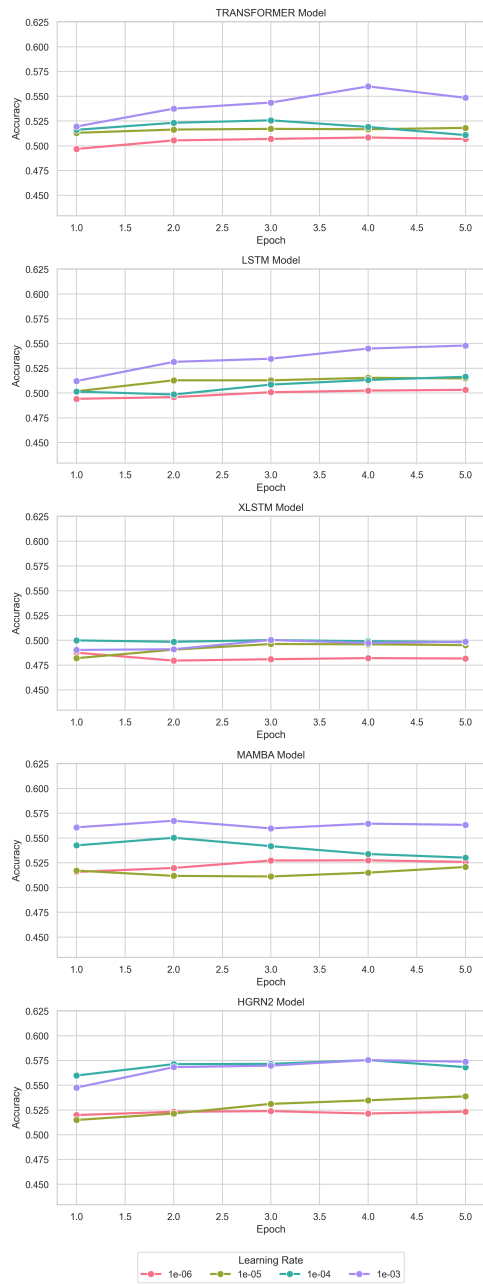
Figure 3: Evaluation results of learning rate sweep over different architectures. Scores are reported as the macro average over the three zero-shot benchmarks BLiMP, BLiMP-Supplement and EWoK.

D Final BabyLM Evaluation Scores

We provide detailed scores of all SuperGLUE, BLiMP-Supplement and EWoK tasks in Tables 8, 9 and 10. Due to the large number of subtasks in BLiMP, we will make the scores accessible though our Github repository: `https://github.com/HallerPatrick/BabyLM-2024`.

						SuperGLUE						
Model (variant)	BoolQ	CoLA (MCC)	MNLI	MNLI-MM	MRPC (F1)	MulitRC	QNLI	QQP (F1)	RTE	SST-2	WSC	Average
Strict-small Track (10M Words)												
BabyLlama$_{baseline}$	65.0	2.2	72.4	74.2	82.0	60.1	82.8	83.6	49.6	86.2	38.5	63.3
LTG-BERT$_{baseline}$	68.8	0.0	68.9	68.9	82.2	58.5	76.5	34.2	58.3	85.1	61.5	60.3
HGRN2	63.8	19.1	**68.7**	**68.7**	82.5	63.4	64.7	79.9	58.9	85.5	38.5	63.0
HGRN2$_{distilled}$	65.4	33.1	69.3	69.5	81.0	59.7	72.3	81.9	54.0	89.4	48.1	**65.8**
Strict-small Track (100M Words)												
BabyLlama$_{baseline}$	66.1	37.3	75.6	76.2	86.8	62.1	83.1	84.5	60.4	88.3	38.5	69.0
LTG-BERT$_{baseline}$	61.7	34.6	77.7	78.1	83.1	52.6	78.2	86.7	46.8	91.5	61.5	68.4
HGRN2	64.4	39.9	**74.3**	**74.3**	82.8	61.4	79.9	83.1	58.9	89.6	51.6	69.1
HGRN2$_{distilled}$	64.8	40.3	74.8	75.9	81.5	61.4	81.5	84.1	58.3	90.1	65.4	**70.7**
Majority Labels$_{val}$	64.0	69.9	35.7	-	68.1	57.7	50.9	62.7	53.9	51.8	61.5	57.6

Table 8: Detailed results for every task in die (Super)GLUE benchmark for the `strict` and `strict-small` track.

Model	Hypernym	QA congruence (easy)	QA congruence (tricky)	Subj.-Aux. Inversion	Turn Taking	Average
Strict-small Track (10M Words)						
BabyLlama	49.6	54.7	41.2	86.0	66.1	59.5
LTG-BERT	54.2	62.5	49.1	79.9	58.2	**60.8**
HGRN2$_{distill}$	49.8	56.2	37.6	89.6	59.6	58.6
Strict Track (100M Words)						
BabyLlama	45.6	56.2	44.8	83.9	72.5	60.6
LTG-BERT	55.0	75.0	53.3	87.5	61.4	**66.5**
HRN2$_{distill}$	48.6	64.1	35.8	84.9	59.3	58.5

Table 9: Detailed results for the BLiMP-Supplement benchmark for the `strict` and `strict-small` track.

Model	Agent Properties	Material Dynamics	Material Properties	Physical Dynamics	Physical Interactions	Physical Relations	Quantitative Properties	Social Interactions	Social Properties	Social Relations	Spatial Relations	Macroaverage
Strict-small Track (10M Words)												
BabyLlama	50.5	51.7	49.4	54.2	50.4	50.6	53.5	50.7	50.3	49.8	46.7	50.7
LTG-BERT	50.2	51.0	45.3	42.5	49.1	51.0	48.1	51.7	53.4	50.6	45.3	48.9
HGRN2$_{distll}$	50.1	50.9	50.6	55.0	50.7	50.4	51.3	54.1	51.2	50.3	49.8	**51.3**
Strict Track (100M Words)												
BabyLlama	50.1	55.5	50.0	57.5	51.4	50.5	56.7	52.7	49.7	50.0	49.0	**52.1**
LTG-BERT	50.1	55.8	50.6	58.3	48.9	50.9	53.8	51.4	50.8	53.8	49.2	51.9
HGRN2$_{distll}$	50.2	52.5	51.8	49.2	51.4	50.6	54.5	51.4	57.0	49.7	49.6	51.6

Table 10: Detailed results for the EWoK benchmark for the `strict` and `strict-small` track.