Instability of LLM Pre-Pretraining: It Doesn't Always Help. An Investigation on Multiple Languages

Anonymous ACL submission

Abstract

Pretraining LLMs on artificial languages ("pre-pretraining") is a technique that could reportedly increase token efficiency by 33%, i.e., save up to 33% of training tokens needed to reach a certain performance. We validate this prior result for English on a larger set of natural languages across four language families, using two different tokenizers and varying model sizes. We also relate the observed gains (or losses) in token efficiency to quantified linguistic properties of the languages, such as sentence length, morphological richness, and features of dependency syntactic trees (tree depth, number of children, number of crossing dependencies). Our empirical results indicate that the reported gains depend heavily on the experiment setup and the choice of random seed, although we can confirm the trend of stable gains with 128-Dyck pretraining of small models with the Llama tokenizer for most of the examined languages. On a general note, we argue that multiple training runs should be carried out at least for a subset of experiments to avoid the community adopting unstable approaches.

1 Introduction

Language models require more training data as their size grows, which poses a challenge for low-resource languages and specific domains. Transfer learning, i.e., benefiting from the knowledge in a different dataset, is one of the established techniques for this situation. Previous work in the machine translation area (Kocmi and Bojar, 2018; Lin et al., 2019) have shown that initial training on a related or even fully unrelated language pairs improves performance on the language pair of interest. Later, gains were also observed when pretraining large language models (LLMs) on artificial languages (or "pre-pretraining") (Ri and Tsuruoka, 2022; Papadimitriou and Jurafsky, 2023; Hu et al., 2025). However, these recent studies on LLMs were solely focused on English.

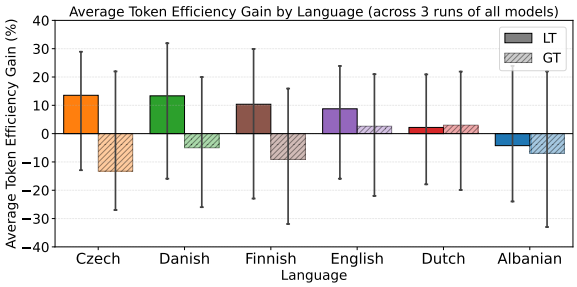


Figure 1: Average token efficiency gains of all models with two different tokenizers, where `LT` stands for Llama, and `GT` stands for Gemma tokenizer.

To our knowledge, it remains unclear whether the method consistently improves token efficiency—i.e., reduces the amount of data needs to achieve a given performance—across other natural languages, model sizes, and architectures.

In this work, we go beyond the previous studies of LLM transfer learning from artificial languages in two main directions: we extend (1) the set of formal languages used, and (2) the set of natural languages of different families, seeking a relation between the properties of the artificial and natural languages affecting the effectiveness of the method.

Experimental results suggest that the pretraining method may help across different natural languages, although the observed gains are highly dependent on the particular training run and are also sensitive to the setup details (Figure 1).

2 Related Work

Ri and Tsuruoka (2022) show that training LSTM- and Transformer-based language models on artificial languages can provide knowledge transfer to natural ones. They conclude `Nesting Dependency` outperforms `Flat Dependency` (see Section 3.1) in importance for language modeling. However, in contrast to the findings of Ri and Tsuruoka (2022), Kumar et al. (2025) observe that pretraining on

CROSS outperforms pretraining on NEST.[1]

Subsequent studies in Papadimitriou and Jurafsky (2023) and the ACL outstanding paper by Hu et al. (2025) claim that applying the same method to a decoder-only LLM can improve its language acquisition, reducing perplexity for the Pythia 160M model, and also resulting in a 33% token efficiency gain for the Pythia 1B variant (albeit with one run).

Budnikov and Yamshchikov (2025) suggest that some artificial languages used in pretraining lead to structurally richer token embeddings, but they fail to see any direct transfer of this to English.

3 Dataset

3.1 Formal Languages for Pretraining

Among formal languages, we consider k-Dyck (demonstrating Nesting Dependency) and k-Shuffle Dyck (Flat Dependency) languages for $k = 64$ and $k = 128$, which are defined as follows:

k-Dyck contains k different types of structurally-nested dependencies, each of which is denoted by matching pair of opening and closing brackets, e.g., ([{ }]) or ([ ]) { }.

k-Shuffle Dyck allows breaking the well-nestedness (e.g., ([ ] {) }) and including cross-serial dependencies (e.g., ([{)] }).

We follow the approach of Hu et al. (2025) and represent the bracket symbols with non-negative integers. In other words, if an opening symbol is denoted as the number n, its corresponding closing symbol is represented as $n + k$, where k is the number of distinct dependencies, and $0 \leq n < k$. An example pair of opening and closing symbols for $k = 128$ is ("81", "209"). We refer readers to Hu et al. (2025) for the exact implementation.

3.2 Natural Languages

Aside from English, we analyze the effects of pretraining for five other typologically diverse European languages from four language families, namely Albanian (Albanoid), Czech (Slavic), Danish and Dutch (Germanic), and Finnish (Uralic).

We construct the training dataset for each language as a monolingual, multi-domain dataset by mixing either FineWeb (for English; Penedo et al. 2024) or FineWeb 2 (for the other five languages; Penedo et al. 2025) with OpenSubtitles (Lison and Tiedemann, 2016) and EUBookshop (Tiedemann, 2012), in proportions of 8:1:1, respectively.

[1] NEST is equivalent to Nesting Dependency; CROSS is equivalent to Flat Dependency

Importantly, we attempt to isolate the effect of pretraining corpus size observed by Kocmi and Bojar (2018): a larger pretraining corpus always delivers bigger gains. Across all experiments, our pretraining corpus consists of 16,000 Dyck sequences of 2,048 tokens (i.e., 500 training steps with the batch size of 32) and our natural language corpus used afterwards is 320,000 sequences[2] of 2,048 tokens (i.e., 10,000 steps with the batch size of 32), subject to minor variations due to tokenization.

For evaluation, we use the first 2,000 rows of the validation split of the Multilingual Colossal Clean Crawled Corpus (mC4) dataset.[3]

4 Experiments and Results

4.1 Experiment Setups

We experiment with three model sizes, based on Llama 3 architecture (Grattafiori et al., 2024). Each model is accompanied by either Llama 3 (LT) or Gemma 3 tokenizer (GT) (Team et al., 2025). This results in six different models, ranging from 253M to 884M. These models, along with the shorthand aliases used hereafter, are outlined in Table 1.

Config	# of parameters			Name
	Effective	Embedding	Total	
0	154M	99M	253M	154M+LT
		201M	355M	154M+GT
1	308M	131M	440M	308M+LT
		268M	577M	308M+GT
2	481M	197M	678M	481M+LT
		403M	884M	481M+GT

Table 1: Overview of models. LT and GT stand for models with Llama and Gemma tokenizer, respectively.

With our fixed training budget of natural language tokens, we aim for our models to be as comparable as those of Hu et al. (2025) in terms of token-to-parameter ratio (Table 2). Importantly, we note that all ratio values in both our and Hu et al. (2025)'s experiments cannot be considered Chinchilla-optimal (Hoffmann et al., 2022), which proposes the optimal value of roughly 20-25 tokens (or even more) per parameter found in recent open-weight LLMs (Grattafiori et al., 2024; Team et al., 2025; DeepSeek-AI, 2026). Nonetheless, we see that our training and evaluation loss curves are stable in the last 3,000 training steps (e.g., in Figure 2 and Appendix A.4), which suggests that our models are sufficiently trained and comparable to

[2] In line with the practice of training LLMs but different from machine translation area, our sequences span sentence boundaries.

[3] allenai/c4

those of Hu et al. (2025). Crucially, we see little to no chance that the setups (baseline vs. pretrained) would change order with more training.

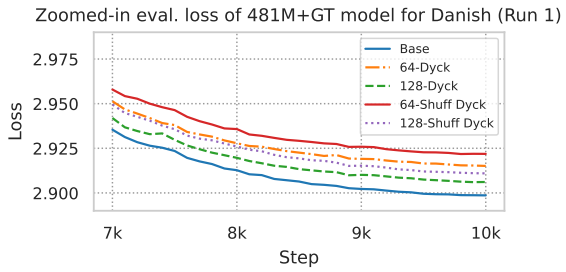


Figure 2: Evaluation loss of the 481M+GT model for Danish, in the last 3,000 training steps. Lower is better.

	Model	Size	Tokens	Ratio
Hu et al. (2025)	Pythia	160M	655M	$\approx 4:1$
	Pythia	1B	1.63B	$1.63:1$
Ours (Llama-3-based models)	154M+LT	253M	655M	$\approx 2.6:1$
	154M+GT	355M	655M	$\approx 1.8:1$
	308M+LT	440M	655M	$\approx 1.5:1$
	308M+GT	577M	655M	$\approx 1.1:1$
	481M+LT	678M	655M	$\approx 0.97:1$
	481M+GT	884M	655M	$\approx 0.74:1$

Table 2: Comparison of token-to-parameter ratios

An important difference for our artificial language presentation is that GT splits each number into separate digits,[4] different from LT.[5]

Each model is trained for one epoch on the full training dataset (see Section 3.2) with three approaches: (1) without pretraining (Base); (2) after being pretrained with k-Dyck; and (3) after being pretrained with k-Shuffle Dyck. For (2) and (3), $k = 64$ or 128. For each setup, we collect three training runs with three different seed choices.

Following Hu et al. (2025), we quantify the improvement by token efficiency gain, which reflects how many training tokens sooner the model pretrained on the artificial language has reached the same validation loss as the corresponding Base model. If the pretrained model performs worse, we negate the value. Our exact calculation is provided in Appendix A.3. Details of hyper-parameters used in our training are described in Appendix A.

4.2 Results

The results across three runs reveal unexpectedly high variance within each identical setup, observed both in the smallest (Figure 3) and the bigger models (see Figures 11 and 12 in Appendix B).

[4]e.g., "81 209" → ['8', '1', '_', '2', '0', '9']
[5]e.g., "81 209" → ['81', 'Ġ', '209']

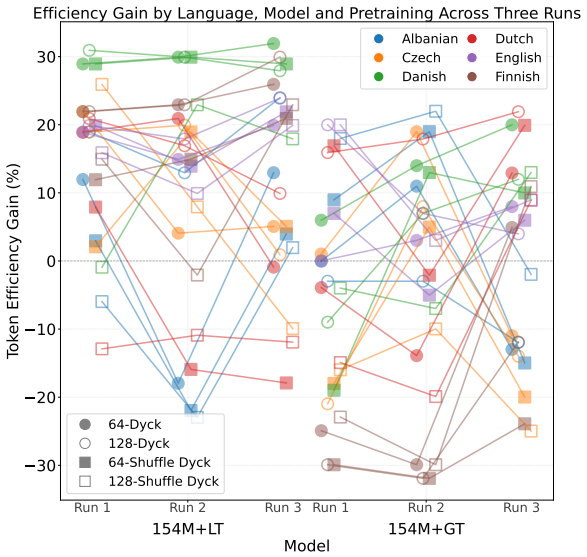


Figure 3: Average token efficiency gains/losses by language of 154M models. The lines connect the identical setups across the three runs; horizontal lines thus indicate stable behavior, while lines crossing the x-axis indicate highly unstable cases where some training runs lead to a gain and some to a loss.

However, in one specific setup, namely, 154M+LT pretrained on 128-Dyck, pretraining does bring consistent improvement (Figure 4 and Table 3). In addition, our experimental results with English shown in Table 3 suggests that pretraining the smallest 154M models with most artificial languages (barring only 64-Shuff Dyck) also leads to constant gain; therefore, we can safely corroborate the findings of previous studies for English (see Section 2).

Notably, models with LT on average have higher gains than those with GT (Figure 1). For the former, k-Dyck usually outperforms k-Shuffle Dyck in terms of gains, while the pattern is not that clear for the latter models (see Figure 11 in Appendix B).

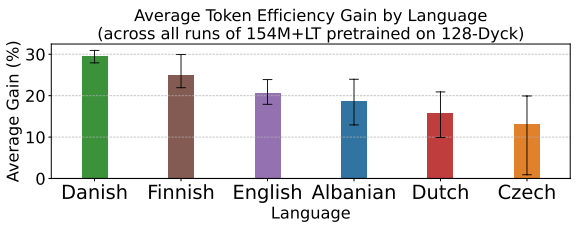


Figure 4: Average token efficiency gains by language (for 154M+LT pretrained on 128-Dyck)

We also conduct an additional experiment with one of the least stable setups—the 154M+GT model pretrained on 64-Dyck and subsequently trained on Czech—which we pretrain with 500, 1,000, and

Natural Language	Pretraining Language								Summary		
	64-Dyck		128-Dyck		64-Shuff Dyck		128-Shuff Dyck		LT	GT	Overall
Albanian	×	×	+	−	×	×	×	×	1+ 3× 0−	0+ 3× 1−	1+ 6× 1−
Czech	+	×	+	×	+	×	×	−	3+ 1× 0−	0+ 3× 1−	3+ 4× 1−
Danish	×	+	+	×	+	×	×	×	2+ 2× 0−	1+ 3× 0−	3+ 5× 0−
Dutch	+	×	+	+	×	×	−	−	2+ 1× 1−	1+ 2× 1−	3+ 3× 2−
English	+	+	+	+	+	×	+	+	4+ 0× 0−	3+ 1× 0−	7+ 1× 0−
Finnish	+	×	+	−	+	−	×	×	3+ 1× 0−	0+ 2× 2−	3+ 3× 2−

Table 3: Consistency of the results of pretraining with the 154M models. We only indicate if the setup lead to a consistent gain (+) over the Base model in all three runs, to a consistent loss (−), or to a mixed result (×) where some of the runs benefited and some lost. In each pretraining language group, the left and right columns represent the result with LT and GT, respectively. The summaries report the number of consistent gains, mixed outcomes and losses across the respective setting.

2,000 steps, three runs per value. Figure 5 suggests that pretraining with 1,000 steps could be optimal.

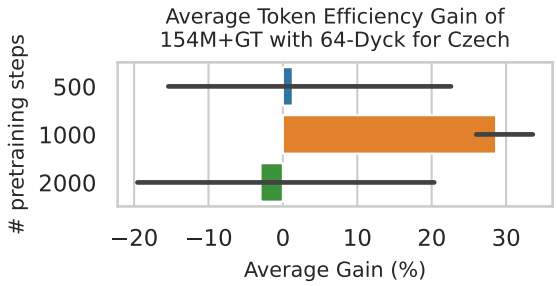


Figure 5: Average token efficiency gains of 154M+GT pretrained on 64-Dyck for Czech, with three different pretraining checkpoints. Right is better.

5 Language Specific Analysis

Might there be any correlation between the gain and the linguistic properties? Due to observed inconsistencies, we limit the analysis to 154M models and conduct significance testing. All definitions of the linguistic metrics used, along with all details of our language-specific analysis are provided in Appendices C and D, respectively.

We observe that only the 154M+GT models show statistically significant correlations stable across runs and pretraining languages. For these setups, positive correlation can generally be seen with *Mean Children*, *Average Perplexity*, and *Average Sentence Length*. Whereas, the level of *Morphological Richness*, *Vocabulary Size*, and *Max. Number of Children* corresponds to lower efficiency gains or losses (see Figures 17 and 18 in Appendix D).

Crossing Dependencies. Our primary hypothesis was that models pretrained on k-Shuffle Dyck would benefit from data with high *Percentage of Crossing Dependencies*. We expected the gains to correspond to the distribution of crossing dependencies (Figure 6). However, we do not detect neither stable across runs nor average positive statistically significant correlations between gains of models pretrained on k-Shuffle Dyck and *Percentage of Crossing Dependencies* in either data split. (see Figures 13 and 14 in Appendix D).

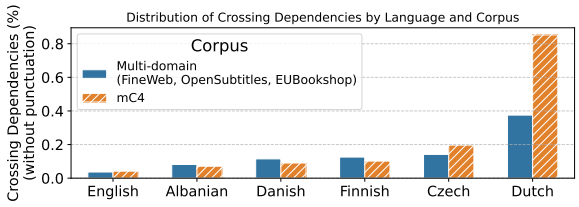


Figure 6: Distribution of crossing dependencies by language and corpus, sorted by the percentage value

6 Conclusion

In this work, we investigate the benefit of pretraining LLMs with artificial languages by extending the experiments to five other natural languages aside from English. Our results suggest that the approach may work for other languages than English too. However, it is highly sensitive to the exact setup, i.e., artificial language, tokenizer choice, and the amount of pretraining. Often, training runs differing only in the random seed lead to different gains or losses. We thus stress the need to validate experimental results by multiple runs. With the pretraining approach, we observe the most stability with small models using the Llama tokenizer and pretrained on 128-Dyck; while all other setups produce mixed results. Finally, we find almost no relation of the gains to over 20 linguistic properties.

Limitations

The work presented in this paper faces several limitations that restricted us from having more comprehensive results.

First, one noticeable issue concerns the easily understandable lack of a large-scale, multi-parallel dataset in multiple languages, both in training and evaluation. Having this would allow us to have better and fairer comparisons between models. The currently available datasets are mostly from the Internet, which might not reflect the highest quality in the language modeling task.

Second, due to the sheer scale of the experiments, we can only train each model with 10,000 training steps (which corresponds to approximately 655M tokens). In addition, on the technical side, we fail to conduct more experiments with a wider range and combination of training hyper-parameters. Even though the resulting models successfully converged during training, we cannot claim that our chosen set of training hyper-parameters is the best one.

Third, we acknowledge that with just 655M training tokens, our models could not be considered Chinchilla-optimal (Hoffmann et al., 2022), and should be viewed as "undertrained" compared to any production-grade LLMs. We nevertheless believe that the presented learning curves are stable enough to convincingly identify if the pre-pretraining has helped or harmed in that particular run.

Fourth, we only conduct experiments with six languages from the European region, which share the same Latin script. As a result, we can only claim the analyses for the examined languages, and this study should not be viewed as a proof that the method works for most, if not all, languages. We leave additional languages and writing scripts for future research.

References

Ondřej Bojar. 2015. Machine Translation. In *The Oxford Handbook of Inflection*. Oxford University Press.

Mikhail Budnikov and Ivan Yamshchikov. 2025. Transfer of Structural Knowledge from Synthetic Languages. In *Proceedings of the 1st Joint Workshop on Large Language Models and Structure Modeling (XLLM 2025)*, pages 242–251, Vienna, Austria. Association for Computational Linguistics.

Christos Christodouloupoulos and Mark Steedman. 2015. A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, 49(2):375–395.

DeepSeek-AI. 2026. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence.

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, and 542 others. 2024. The Llama 3 Herd of Models. *Preprint*, arXiv:2407.21783.

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, and 3 others. 2022. Training Compute-Optimal Large Language Models. *Preprint*, arXiv:2203.15556.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. 2025. Between Circuits and Chomsky: Pre-pretraining on Formal Languages Imparts Linguistic Biases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9691–9709, Vienna, Austria. Association for Computational Linguistics.

Tom Kocmi and Ondřej Bojar. 2018. Trivial Transfer Learning for Low-Resource Neural Machine Translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 244–252, Brussels, Belgium. Association for Computational Linguistics.

Nalin Kumar, Mateusz Lango, and Ondrej Dusek. 2025. Pretraining Language Models with LoRA and Artificial Languages. In *Proceedings of the First BabyLM Workshop*, pages 525–530, Suzhou, China. Association for Computational Linguistics.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing Transfer Languages for Cross-Lingual Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8402–8413, Singapore. Association for Computational Linguistics.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. FineWeb2: One Pipeline to Scale Them All – Adapting Pre-Training Data Processing to Every Language. *Preprint*, arXiv:2506.20920.

Ryokan Ri and Yoshimasa Tsuruoka. 2022. Pretraining with Artificial Language: Studying Transferable Knowledge in Language Models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7302–7315, Dublin, Ireland. Association for Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 Technical Report. *Preprint*, arXiv:2503.19786.

Jörg Tiedemann. 2012. Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

A Details of Experiment Setups

A.1 Model hyper-parameters

We take the model configuration of the published `meta-llama/Llama-3.2-1B` checkpoint, then modify four hyper-parameters as described in Table 4. For each setup, we collect three training runs with three different seed choices of 17, 42, and 3407.

A.2 Training hyper-parameters

Table 5 details chosen training hyper-parameters.

A.3 Algorithm to Calculate Token Efficiency Gain

The algorithm to calculate the token efficiency gain is described in Algorithm 1.

Config	0	1	2
`num_hidden_layers`	8	12	16
`num_attn_heads`	8	8	16
`head_dim`	16	16	32
`hidden_size`	768	1,024	1,536

Table 4: Details of models' hyper-parameters.

Training hyper-parameters	**Value**
Effective batch size	32
Learning rate	5e-4
Optimizer	AdamW
LR Scheduler	cosine
Min. learning rate	0.1
Warmup steps	1,000
Weight decay	0.1
Max. grad. norm.	1.0

Table 5: Training hyper-parameters

A.4 Training and Evaluation Losses

We provide some figures describing the loss curves within our experiments. Figures 7 and 8 show the training and evaluation loss curves of the `154M+{LT,GT}` models for Czech. Figures 9 and 10 describe the training and evaluation loss curves of the `154M+{LT,GT}` models for Albanian.

B Details of Token Efficiency Gain

Table 3 describes the consistency of evaluation results for the same model (`154M`) with different tokenizers (Llama - `LT` or Gemma - `GT`) across all pretraining and natural languages. Symbols '+', '−', and '×' denote consistent gain, consistent loss, and inconsistency across three runs (i.e., some of the runs resulted in efficiency gain, while others in loss or negligible (near-zero) gain), respectively.

Figures 11 and 12 describe how the token efficiency gain differs between all model setups, runs, pretraining, and natural languages.

Table 6 provides performance gain (or loss) results for all experiments.

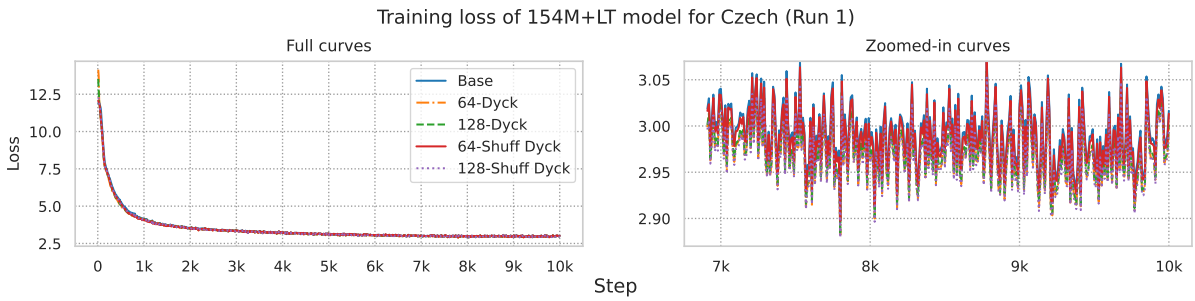
Algorithm 1: Calculate Token Efficiency Gain

Input: Sequence of the baseline model's validation losses B, Sequence of the pretrained model's validation losses P

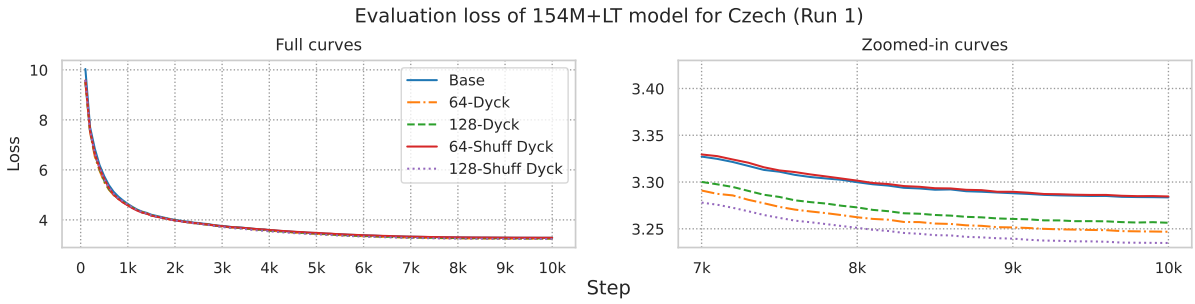
Output: Token Efficiency Gain (%)

$PrePretrainedTokens \leftarrow$ Total number of tokens in the pre-pretraining step
$BaselineTokens \leftarrow$ Total number of tokens used by the baseline model
$TargetLoss \leftarrow$ Final value in B

if $i \geq TargetLoss \; \forall i \in P$ **then**
  $Gain \leftarrow$ Calculate Token Efficiency Gain(P, B)
  return $-Gain$
end
else
  $n \leftarrow length(P)$
  $k \leftarrow \min\{j \in \{0, \ldots, n-1\}$ such that $\forall i, j \leq i < n : P[i] \leq TargetLoss\}$
  $PretrainedTokens \leftarrow$ Number of tokens used by the pretrained model at index k
  $PretrainedTokens \leftarrow PretrainedTokens + PrePretrainedTokens$
  $Gain \leftarrow (1 - \frac{PretrainedTokens}{BaselineTokens}) \times 100$
  return $Gain$
end

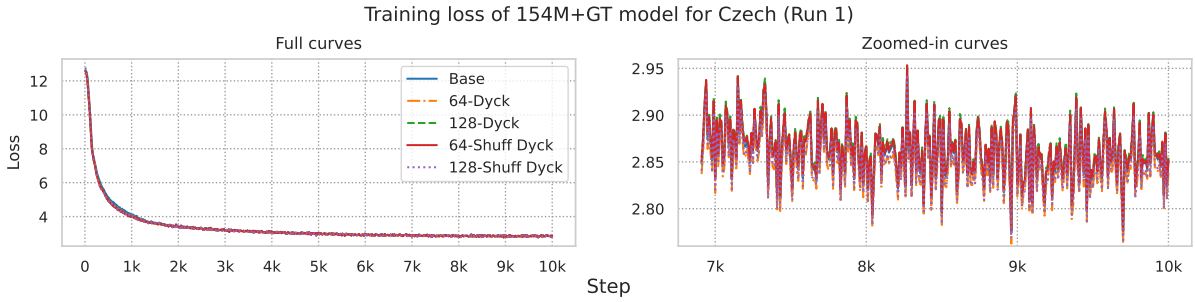


(a) Training Loss

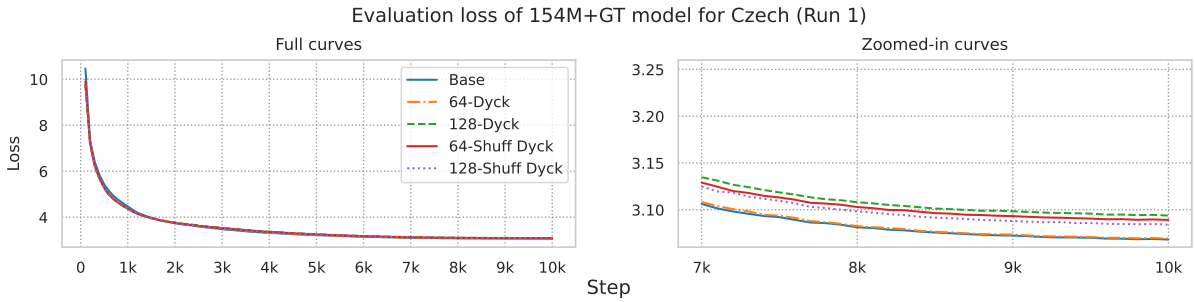


(b) Evaluation Loss

Figure 7: Loss curves of the 154M+LT model for Czech. The right-side figure is the zoomed-in version of the left one, from step 7,000 to 10,000.

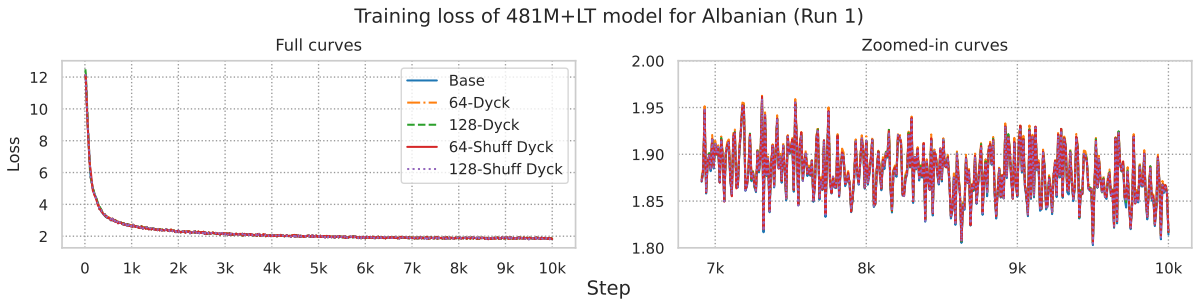


(a) Training Loss

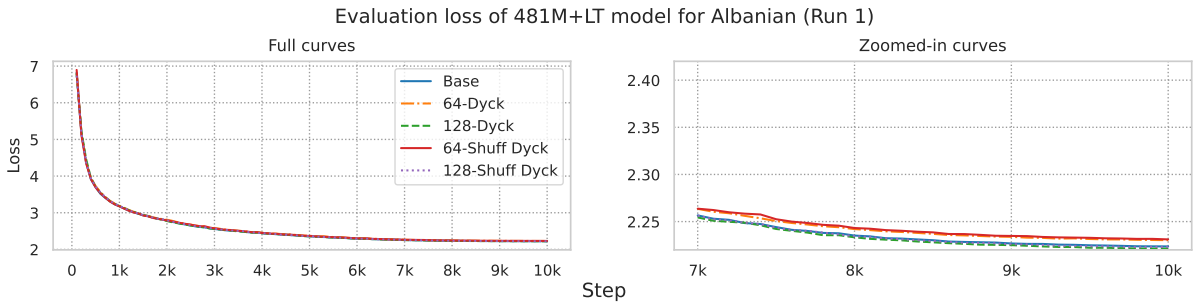


(b) Evaluation Loss

Figure 8: Loss curves of the 154M+GT model for Czech. The right-side figure is the zoomed-in version of the left one, from step 7,000 to 10,000.

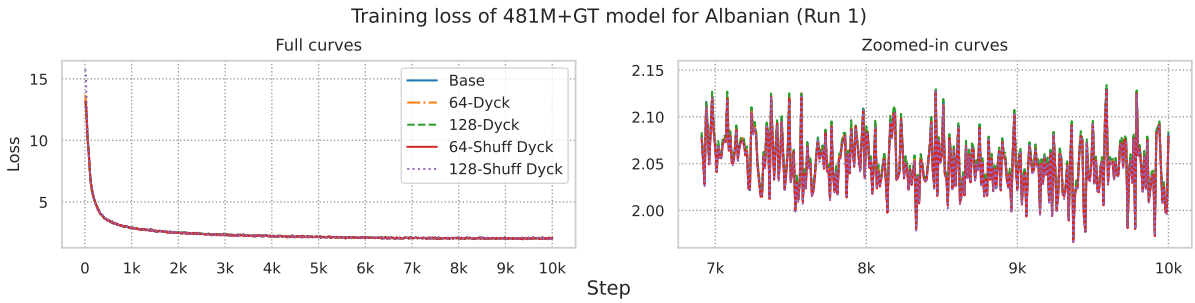


(a) Training Loss

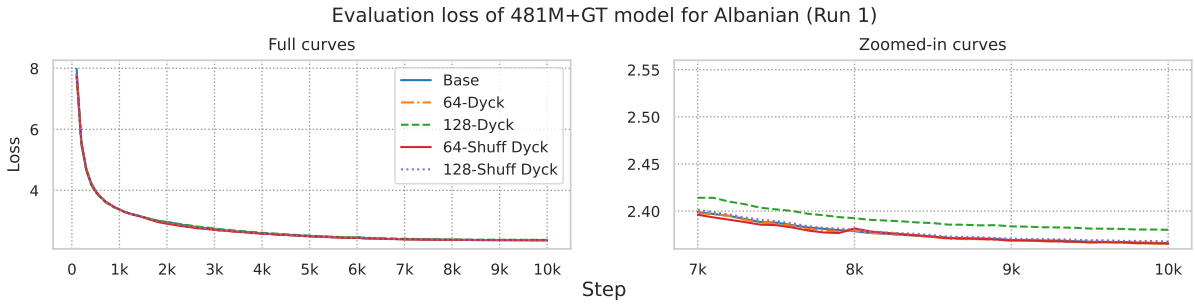


(b) Evaluation Loss

Figure 9: Loss curves of the 481M+LT model for Albanian. The right-side figure is the zoomed-in version of the left one, from step 7,000 to 10,000.



(a) Training Loss



(b) Evaluation Loss

Figure 10: Loss curves of the 481M+GT model for Albanian. The right-side figure is the zoomed-in version of the left one, from step 7,000 to 10,000.

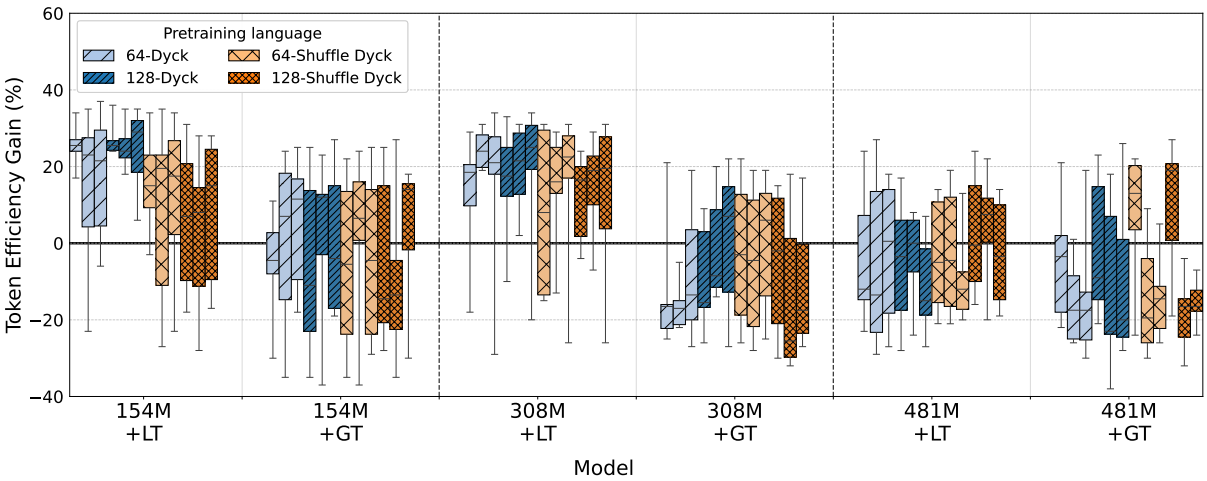


Figure 11: Efficiency gain across all models, pretraining, and natural languages.

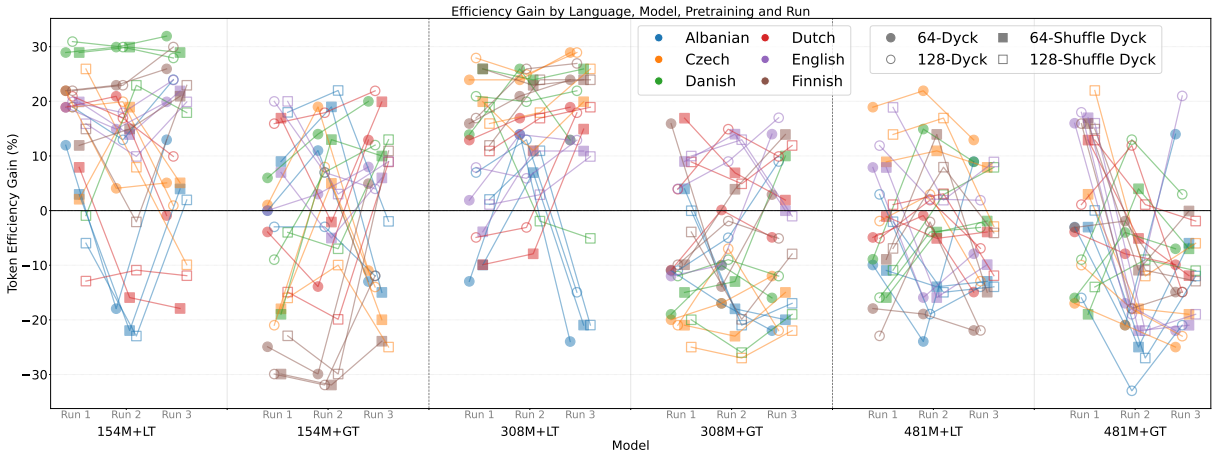


Figure 12: Efficiency gain across all models, pretraining, and natural languages (with language-specific details). The lines connect the identical setups across the three runs; horizontal lines thus indicate stable behavior, while lines crossing the x-axis indicate highly unstable cases where some training runs lead to a gain and some to a loss.

Table 6: Token efficiency gain across experiments

Model	Language	Gain % ↑
481M+GT_128-dyck_run2	Albanian	-32.980
154M+GT_128-dyck_run2	Finnish	-31.915
154M+GT_64-shuff-dyck_run2	Finnish	-31.915
154M+GT_128-dyck_run1	Finnish	-29.915
154M+GT_64-shuff-dyck_run1	Finnish	-29.915
154M+GT_128-shuff-dyck_run2	Finnish	-29.911
154M+GT_64-dyck_run2	Finnish	-29.911
481M+GT_128-shuff-dyck_run2	Albanian	-26.983
308M+GT_128-shuff-dyck_run2	Czech	-26.982
308M+GT_128-shuff-dyck_run2	Danish	-25.985
308M+GT_128-shuff-dyck_run1	Czech	-24.981
481M+GT_64-dyck_run3	Czech	-24.974
481M+GT_64-shuff-dyck_run2	Albanian	-24.974
154M+GT_128-shuff-dyck_run3	Czech	-24.973
154M+GT_64-dyck_run1	Finnish	-24.919
308M+LT_64-dyck_run3	Albanian	-23.966
481M+LT_64-dyck_run2	Albanian	-23.951
154M+GT_64-shuff-dyck_run3	Finnish	-23.904
308M+GT_64-shuff-dyck_run2	Czech	-22.982
481M+GT_128-dyck_run3	Czech	-22.982
154M+LT_128-shuff-dyck_run2	Albanian	-22.956
481M+LT_128-dyck_run1	Finnish	-22.931
154M+GT_128-shuff-dyck_run1	Finnish	-22.918
481M+GT_64-dyck_run3	English	-21.977
481M+GT_128-shuff-dyck_run2	English	-21.977
481M+GT_64-shuff-dyck_run2	English	-21.977
308M+GT_128-dyck_run3	Czech	-21.970
308M+GT_128-shuff-dyck_run3	Czech	-21.970
308M+GT_64-dyck_run3	Albanian	-21.968
154M+LT_64-shuff-dyck_run2	Albanian	-21.956
481M+LT_128-dyck_run3	Finnish	-21.936
481M+LT_64-dyck_run3	Finnish	-21.936
481M+GT_64-shuff-dyck_run3	English	-20.982
308M+GT_128-dyck_run1	Czech	-20.978
154M+GT_128-dyck_run1	Czech	-20.978
308M+GT_64-shuff-dyck_run1	Czech	-20.978
481M+GT_64-dyck_run2	Czech	-20.974
308M+GT_128-shuff-dyck_run2	Albanian	-20.971
481M+GT_128-dyck_run3	Albanian	-20.971
481M+GT_64-dyck_run2	Albanian	-20.971
308M+LT_128-shuff-dyck_run3	Albanian	-20.950
308M+LT_64-shuff-dyck_run3	Albanian	-20.950
308M+GT_128-shuff-dyck_run1	Danish	-19.980
308M+GT_64-dyck_run1	Czech	-19.979
154M+GT_64-shuff-dyck_run3	Czech	-19.976
308M+GT_64-shuff-dyck_run3	Albanian	-19.974
308M+GT_128-shuff-dyck_run2	Finnish	-19.906
154M+GT_128-shuff-dyck_run2	Dutch	-19.902
308M+GT_128-shuff-dyck_run3	Danish	-18.985
481M+GT_64-shuff-dyck_run3	Czech	-18.981
308M+GT_64-dyck_run1	Danish	-18.980
481M+GT_64-shuff-dyck_run1	Danish	-18.980
154M+GT_64-shuff-dyck_run1	Danish	-18.980
481M+GT_128-shuff-dyck_run3	English	-18.976
481M+GT_128-dyck_run2	English	-18.976
481M+LT_128-dyck_run2	Albanian	-18.949
481M+LT_64-dyck_run2	Finnish	-18.929
308M+GT_64-shuff-dyck_run2	Albanian	-17.980
154M+GT_64-shuff-dyck_run1	Czech	-17.978
481M+GT_64-shuff-dyck_run2	Czech	-17.969
481M+GT_128-dyck_run2	Czech	-17.969
154M+LT_64-dyck_run2	Albanian	-17.955
481M+LT_64-dyck_run1	Finnish	-17.925
154M+LT_64-shuff-dyck_run3	Dutch	-17.908
481M+GT_128-dyck_run2	Finnish	-17.902
308M+GT_64-dyck_run2	Albanian	-16.983
481M+GT_64-dyck_run1	Czech	-16.979
308M+GT_64-dyck_run2	Czech	-16.973
481M+GT_64-dyck_run2	English	-16.970
308M+GT_128-shuff-dyck_run3	Albanian	-16.968
308M+GT_64-dyck_run3	Danish	-15.982
481M+GT_64-dyck_run1	Danish	-15.981
154M+GT_128-shuff-dyck_run1	Czech	-15.979
481M+GT_128-dyck_run1	Albanian	-15.977
154M+LT_64-shuff-dyck_run2	Dutch	-15.919
481M+LT_128-dyck_run1	Danish	-15.916
481M+LT_64-shuff-dyck_run1	Danish	-15.916
481M+LT_64-shuff-dyck_run2	English	-15.915
481M+LT_64-dyck_run2	English	-15.915
308M+GT_64-shuff-dyck_run1	Danish	-14.981
308M+GT_64-shuff-dyck_run3	Czech	-14.980
154M+GT_64-shuff-dyck_run3	Albanian	-14.975
308M+LT_128-dyck_run3	Albanian	-14.948
481M+LT_128-shuff-dyck_run2	Albanian	-14.948
481M+LT_64-shuff-dyck_run3	Finnish	-14.931
481M+GT_128-dyck_run3	Finnish	-14.916
481M+GT_64-dyck_run3	Finnish	-14.916
481M+LT_64-dyck_run3	Dutch	-14.912
481M+GT_128-dyck_run3	Dutch	-14.910
154M+GT_128-shuff-dyck_run1	Dutch	-14.909
154M+GT_128-dyck_run3	Czech	-13.982
481M+GT_128-shuff-dyck_run1	Danish	-13.980
481M+LT_64-shuff-dyck_run2	Albanian	-13.948
481M+LT_128-dyck_run3	Albanian	-13.948
481M+LT_128-shuff-dyck_run3	Albanian	-13.948
308M+GT_64-dyck_run2	Finnish	-13.909
154M+GT_64-dyck_run2	Dutch	-13.902
154M+GT_64-dyck_run3	Albanian	-12.981
308M+GT_64-shuff-dyck_run2	Danish	-12.980
308M+LT_64-dyck_run1	Albanian	-12.957
481M+LT_64-shuff-dyck_run3	Albanian	-12.947
481M+LT_128-dyck_run3	Czech	-12.920
481M+GT_128-shuff-dyck_run3	Finnish	-12.913
154M+LT_128-shuff-dyck_run1	Dutch	-12.904
308M+GT_128-dyck_run3	Danish	-11.984
308M+GT_64-dyck_run1	English	-11.983
308M+GT_128-dyck_run1	Danish	-11.981
481M+GT_128-shuff-dyck_run2	Czech	-11.976
308M+GT_64-dyck_run3	Czech	-11.976
154M+GT_128-dyck_run3	Albanian	-11.969
481M+GT_128-shuff-dyck_run3	Albanian	-11.968
481M+LT_128-shuff-dyck_run3	Dutch	-11.915
154M+LT_128-shuff-dyck_run3	Dutch	-11.914
154M+GT_128-dyck_run3	Finnish	-11.912
481M+GT_64-shuff-dyck_run3	Dutch	-11.910
481M+GT_128-shuff-dyck_run3	Danish	-10.988
154M+GT_64-dyck_run3	Czech	-10.978
308M+GT_128-dyck_run1	Albanian	-10.975
308M+GT_64-dyck_run1	Albanian	-10.975
481M+LT_64-shuff-dyck_run1	Albanian	-10.958

Model	Language	Gain % ↑
481M+LT_128-shuff-dyck_run1	Danish	-10.912
481M+GT_128-shuff-dyck_run2	Finnish	-10.911
481M+GT_64-shuff-dyck_run2	Finnish	-10.911
154M+LT_128-shuff-dyck_run2	Dutch	-10.903
308M+GT_64-dyck_run1	Dutch	-10.899
308M+GT_64-dyck_run2	Danish	-9.993
481M+GT_128-dyck_run1	Czech	-9.979
308M+GT_64-dyck_run2	English	-9.973
154M+GT_128-shuff-dyck_run2	Czech	-9.971
481M+LT_64-dyck_run1	Albanian	-9.958
308M+LT_64-shuff-dyck_run1	Albanian	-9.958
154M+LT_128-shuff-dyck_run3	Czech	-9.919
308M+GT_128-dyck_run1	Finnish	-9.912
308M+GT_64-shuff-dyck_run1	Finnish	-9.912
308M+LT_64-shuff-dyck_run1	Dutch	-9.908
481M+GT_64-dyck_run3	Dutch	-9.906
481M+LT_64-shuff-dyck_run3	English	-9.897
154M+GT_128-dyck_run1	Danish	-8.982
481M+GT_128-dyck_run1	Danish	-8.982
481M+GT_128-shuff-dyck_run2	Danish	-8.982
308M+GT_128-dyck_run2	Danish	-8.982
481M+LT_64-shuff-dyck_run1	Finnish	-8.925
481M+LT_64-dyck_run1	Danish	-8.915
308M+GT_128-shuff-dyck_run3	Finnish	-7.927
308M+LT_64-shuff-dyck_run2	Dutch	-7.909
481M+GT_64-dyck_run2	Dutch	-7.902
481M+LT_64-dyck_run3	English	-7.900
481M+GT_64-shuff-dyck_run3	Danish	-6.990
481M+GT_64-dyck_run3	Danish	-6.990
154M+GT_128-shuff-dyck_run2	Danish	-6.987
308M+GT_128-dyck_run2	Czech	-6.979
481M+LT_128-shuff-dyck_run1	Finnish	-6.927
481M+LT_128-dyck_run3	Dutch	-6.915
481M+GT_128-shuff-dyck_run3	Czech	-5.983
481M+GT_64-shuff-dyck_run3	Albanian	-5.971
154M+LT_128-shuff-dyck_run1	Albanian	-5.961
481M+GT_64-shuff-dyck_run2	Dutch	-5.094
308M+GT_128-dyck_run3	Finnish	-5.090
481M+LT_128-dyck_run1	Dutch	-5.089
481M+LT_64-shuff-dyck_run2	Dutch	-5.088
308M+LT_128-shuff-dyck_run3	Danish	-5.082
154M+GT_64-shuff-dyck_run2	English	-5.024
308M+GT_128-dyck_run2	Albanian	-4.974
308M+LT_128-dyck_run1	Dutch	-4.914
481M+LT_64-dyck_run1	Dutch	-4.914
308M+GT_64-dyck_run3	Dutch	-4.901
481M+LT_128-shuff-dyck_run3	Finnish	-4.087
481M+GT_64-dyck_run2	Danish	-4.009
154M+GT_128-shuff-dyck_run1	Danish	-3.980
481M+LT_64-shuff-dyck_run2	Danish	-3.927
481M+LT_128-dyck_run2	Danish	-3.927
481M+LT_64-shuff-dyck_run3	Dutch	-3.918
308M+GT_128-shuff-dyck_run1	Finnish	-3.910
154M+GT_64-dyck_run1	Dutch	-3.908
481M+GT_64-dyck_run1	Dutch	-3.908
308M+LT_64-shuff-dyck_run1	English	-3.885
481M+GT_64-dyck_run1	Finnish	-3.093
308M+LT_128-dyck_run2	Dutch	-3.085
481M+LT_128-dyck_run3	Danish	-3.081
481M+GT_64-dyck_run1	Albanian	-3.026
481M+GT_64-shuff-dyck_run1	Albanian	-3.026

Model	Language	Gain % ↑
154M+GT_128-dyck_run1	Albanian	-2.975
154M+GT_128-dyck_run2	Albanian	-2.974
481M+LT_128-shuff-dyck_run3	Czech	-2.920
154M+GT_64-shuff-dyck_run2	Dutch	-2.096
154M+LT_128-shuff-dyck_run2	Finnish	-2.072
481M+LT_128-dyck_run2	Finnish	-2.072
481M+LT_128-shuff-dyck_run1	Albanian	-2.040
154M+GT_128-shuff-dyck_run3	Albanian	-1.969
308M+LT_128-shuff-dyck_run2	Danish	-1.927
481M+LT_128-dyck_run1	Czech	-1.919
481M+GT_128-shuff-dyck_run3	Dutch	-1.917
481M+GT_64-dyck_run2	Finnish	-1.916
308M+GT_128-dyck_run2	Finnish	-1.916
481M+LT_64-shuff-dyck_run3	Danish	-1.913
481M+LT_64-shuff-dyck_run1	Dutch	-1.088
308M+GT_128-shuff-dyck_run3	English	-1.021
154M+LT_128-shuff-dyck_run1	Danish	-0.918
481M+LT_64-dyck_run2	Dutch	-0.906
154M+LT_64-dyck_run3	Dutch	-0.905
481M+GT_64-shuff-dyck_run3	Finnish	-0.073
308M+GT_128-shuff-dyck_run1	Albanian	-0.027
154M+GT_64-dyck_run1	Albanian	-0.027
308M+GT_64-shuff-dyck_run3	English	-0.026
154M+GT_64-dyck_run1	English	0.016
481M+GT_128-shuff-dyck_run1	Albanian	0.027
308M+GT_64-dyck_run2	Dutch	0.104
154M+LT_128-dyck_run3	Czech	0.910
154M+GT_64-dyck_run1	Czech	1.021
481M+LT_128-shuff-dyck_run1	Dutch	1.088
481M+GT_128-dyck_run1	Dutch	1.094
481M+GT_128-shuff-dyck_run2	Dutch	1.099
308M+LT_128-shuff-dyck_run1	English	1.108
308M+LT_64-dyck_run1	English	1.888
481M+LT_128-dyck_run3	English	1.893
481M+LT_128-shuff-dyck_run2	English	1.908
481M+LT_128-dyck_run2	Dutch	1.915
308M+GT_64-shuff-dyck_run3	Dutch	1.917
154M+LT_128-shuff-dyck_run3	Albanian	1.955
308M+LT_128-shuff-dyck_run1	Albanian	2.040
154M+LT_64-shuff-dyck_run1	Czech	2.080
481M+LT_128-dyck_run2	English	2.101
308M+LT_128-shuff-dyck_run2	English	2.906
481M+LT_128-shuff-dyck_run2	Danish	2.919
308M+GT_64-dyck_run3	Finnish	2.925
481M+LT_128-dyck_run2	Czech	2.935
481M+LT_128-dyck_run1	Albanian	2.962
154M+LT_64-shuff-dyck_run1	Albanian	2.962
481M+GT_128-dyck_run3	Danish	2.977
481M+GT_64-shuff-dyck_run1	Czech	2.979
154M+GT_64-dyck_run2	English	3.020
154M+GT_128-shuff-dyck_run2	English	3.020
481M+LT_128-shuff-dyck_run2	Dutch	3.085
308M+GT_64-shuff-dyck_run2	Finnish	3.904
308M+GT_128-dyck_run1	Dutch	3.908
154M+LT_64-shuff-dyck_run3	Albanian	3.956
308M+GT_64-shuff-dyck_run1	Albanian	3.975
154M+GT_128-dyck_run3	English	3.981
308M+GT_128-dyck_run1	English	3.983
481M+GT_64-shuff-dyck_run2	Danish	3.988
154M+LT_64-dyck_run2	Czech	4.078
308M+GT_128-shuff-dyck_run2	Dutch	4.902

Model	Language	Gain % ↑
154M+GT_64-dyck_run3	Finnish	4.921
154M+GT_64-shuff-dyck_run2	Czech	4.975
154M+LT_64-shuff-dyck_run3	Czech	5.072
154M+LT_64-dyck_run3	Czech	5.072
308M+LT_128-dyck_run2	English	5.901
154M+GT_64-shuff-dyck_run3	English	5.980
154M+GT_64-dyck_run1	Danish	5.982
308M+GT_64-shuff-dyck_run2	Dutch	6.907
308M+LT_64-shuff-dyck_run2	Albanian	6.946
308M+LT_128-dyck_run1	Albanian	6.961
154M+GT_128-dyck_run2	English	6.978
154M+GT_64-shuff-dyck_run1	English	6.983
154M+GT_128-dyck_run2	Danish	6.987
308M+LT_128-dyck_run1	English	7.891
481M+LT_64-dyck_run1	English	7.891
481M+LT_64-shuff-dyck_run1	English	7.891
154M+LT_64-shuff-dyck_run1	Dutch	7.910
481M+LT_64-shuff-dyck_run3	Czech	7.912
481M+LT_128-shuff-dyck_run3	Danish	7.914
154M+LT_128-shuff-dyck_run2	Czech	7.935
481M+LT_128-shuff-dyck_run2	Finnish	7.940
154M+GT_128-dyck_run2	Czech	7.975
154M+GT_64-dyck_run3	English	7.980
481M+LT_128-shuff-dyck_run3	English	8.899
308M+GT_128-shuff-dyck_run1	Dutch	8.903
154M+GT_128-shuff-dyck_run3	Dutch	8.910
481M+LT_64-shuff-dyck_run1	Czech	8.920
481M+LT_64-dyck_run3	Danish	8.922
481M+LT_64-dyck_run3	Albanian	8.962
154M+GT_64-shuff-dyck_run1	Albanian	8.976
308M+GT_128-dyck_run3	Albanian	8.977
308M+GT_128-dyck_run2	English	8.977
154M+GT_128-shuff-dyck_run3	English	8.979
308M+GT_64-shuff-dyck_run1	English	8.983
308M+LT_128-shuff-dyck_run3	English	9.897
308M+GT_128-dyck_run3	Dutch	9.906
154M+LT_128-shuff-dyck_run2	English	9.908
154M+LT_128-dyck_run3	Dutch	9.910
308M+GT_128-shuff-dyck_run1	English	9.982
154M+GT_64-shuff-dyck_run3	Danish	9.986
308M+GT_64-shuff-dyck_run3	Danish	9.993
308M+LT_64-shuff-dyck_run3	English	10.896
308M+LT_64-shuff-dyck_run2	English	10.908
308M+LT_128-shuff-dyck_run1	Dutch	10.908
154M+GT_128-shuff-dyck_run3	Finnish	10.918
308M+LT_64-shuff-dyck_run2	Czech	10.933
481M+LT_64-shuff-dyck_run2	Czech	10.933
308M+LT_128-shuff-dyck_run2	Albanian	10.947
154M+GT_64-dyck_run2	Albanian	10.981
481M+LT_128-dyck_run1	English	11.889
308M+GT_128-shuff-dyck_run3	Dutch	11.910
481M+GT_128-dyck_run2	Dutch	11.912
308M+LT_128-shuff-dyck_run1	Finnish	11.926
154M+LT_64-shuff-dyck_run1	Finnish	11.926
154M+LT_64-dyck_run1	Albanian	11.957
154M+GT_128-dyck_run3	Danish	11.986
308M+LT_64-dyck_run3	English	12.893
308M+LT_128-dyck_run3	English	12.893
308M+LT_64-dyck_run1	Dutch	12.904
481M+GT_128-shuff-dyck_run1	Dutch	12.905
481M+GT_64-shuff-dyck_run1	Dutch	12.905

Model	Language	Gain % ↑
154M+GT_64-dyck_run3	Dutch	12.914
481M+LT_64-dyck_run3	Czech	12.920
308M+LT_64-dyck_run3	Danish	12.924
154M+LT_128-dyck_run2	Albanian	12.945
308M+LT_128-dyck_run2	Albanian	12.947
154M+LT_64-dyck_run3	Albanian	12.955
308M+GT_128-shuff-dyck_run2	English	12.974
481M+GT_128-dyck_run2	Danish	12.980
154M+GT_64-shuff-dyck_run2	Danish	12.985
154M+GT_128-shuff-dyck_run3	Danish	12.986
308M+LT_64-dyck_run1	Danish	13.911
154M+LT_64-shuff-dyck_run2	English	13.912
308M+LT_64-dyck_run2	English	13.919
308M+GT_64-shuff-dyck_run3	Finnish	13.922
481M+LT_128-shuff-dyck_run1	Czech	13.926
481M+LT_64-shuff-dyck_run2	Finnish	13.939
308M+LT_64-dyck_run2	Albanian	13.948
481M+GT_64-dyck_run3	Albanian	13.977
308M+GT_64-dyck_run3	English	13.984
308M+GT_64-shuff-dyck_run2	English	13.984
154M+GT_64-dyck_run2	Danish	13.985
308M+GT_128-dyck_run2	Dutch	14.897
154M+LT_64-dyck_run2	English	14.910
308M+LT_64-shuff-dyck_run3	Dutch	14.912
481M+GT_128-shuff-dyck_run1	Finnish	14.913
481M+LT_64-dyck_run2	Danish	14.917
154M+LT_128-shuff-dyck_run1	Finnish	14.931
154M+LT_64-shuff-dyck_run2	Finnish	14.933
154M+LT_128-shuff-dyck_run1	English	15.891
154M+GT_128-dyck_run1	Dutch	15.910
481M+GT_128-dyck_run1	Finnish	15.915
481M+GT_64-shuff-dyck_run1	Finnish	15.915
308M+GT_64-dyck_run1	Finnish	15.915
308M+LT_128-shuff-dyck_run1	Czech	15.926
308M+LT_64-dyck_run1	Finnish	15.927
481M+GT_64-dyck_run1	English	15.983
481M+GT_128-shuff-dyck_run1	English	15.983
308M+GT_64-shuff-dyck_run1	Dutch	16.911
154M+GT_64-shuff-dyck_run1	Dutch	16.911
308M+LT_128-shuff-dyck_run2	Dutch	16.915
308M+LT_64-dyck_run2	Dutch	16.915
154M+LT_128-dyck_run2	Dutch	16.917
308M+LT_128-dyck_run1	Finnish	16.926
481M+LT_128-shuff-dyck_run2	Czech	16.933
481M+GT_64-shuff-dyck_run1	English	16.983
308M+GT_128-dyck_run3	English	16.985
154M+GT_128-dyck_run2	Dutch	17.904
308M+LT_128-dyck_run3	Dutch	17.909
154M+LT_128-dyck_run2	English	17.911
154M+LT_128-shuff-dyck_run3	Danish	17.919
308M+LT_128-shuff-dyck_run2	Czech	17.925
154M+GT_128-shuff-dyck_run1	Albanian	17.976
481M+GT_128-dyck_run1	English	17.983
154M+LT_64-dyck_run1	English	18.884
481M+LT_128-shuff-dyck_run1	English	18.884
154M+LT_64-dyck_run1	Dutch	18.906
308M+LT_128-shuff-dyck_run1	Danish	18.918
308M+LT_128-shuff-dyck_run3	Dutch	18.918
308M+LT_64-dyck_run3	Dutch	18.918
154M+LT_128-dyck_run1	Czech	18.921
481M+LT_64-dyck_run1	Czech	18.921

Model	Language	Gain % ↑
154M+LT_64-shuff-dyck_run2	Czech	18.928
154M+LT_128-dyck_run1	Albanian	18.954
154M+GT_64-dyck_run2	Czech	18.976
154M+GT_64-shuff-dyck_run2	Albanian	18.977
154M+LT_64-shuff-dyck_run1	English	19.882
154M+LT_128-dyck_run1	English	19.882
154M+LT_128-shuff-dyck_run3	English	19.895
154M+LT_64-dyck_run3	English	19.895
154M+GT_64-shuff-dyck_run3	Dutch	19.910
308M+LT_64-shuff-dyck_run3	Czech	19.912
308M+LT_64-shuff-dyck_run1	Czech	19.922
308M+LT_128-dyck_run2	Danish	19.924
154M+LT_128-dyck_run2	Czech	19.926
154M+GT_128-shuff-dyck_run1	English	19.982
154M+GT_128-dyck_run1	English	19.982
154M+GT_64-dyck_run3	Danish	19.985
154M+LT_128-dyck_run1	Dutch	20.907
154M+LT_64-dyck_run2	Dutch	20.916
308M+LT_128-dyck_run1	Danish	20.922
308M+LT_64-dyck_run2	Finnish	20.929
154M+LT_64-shuff-dyck_run3	Finnish	20.930
481M+GT_128-dyck_run3	English	20.982
154M+LT_64-shuff-dyck_run3	English	21.893
154M+GT_128-dyck_run3	Dutch	21.910
308M+LT_128-dyck_run3	Danish	21.919
154M+LT_64-dyck_run1	Czech	21.922
481M+LT_64-dyck_run2	Czech	21.924
154M+LT_128-dyck_run1	Finnish	21.929
154M+LT_64-dyck_run1	Finnish	21.929
154M+GT_128-shuff-dyck_run2	Albanian	21.979
481M+GT_128-shuff-dyck_run1	Czech	21.979
154M+LT_128-shuff-dyck_run2	Danish	22.926
308M+LT_64-shuff-dyck_run2	Finnish	22.928
154M+LT_128-shuff-dyck_run3	Finnish	22.930
154M+LT_64-dyck_run2	Finnish	22.932
154M+LT_128-dyck_run2	Finnish	22.932
154M+LT_128-dyck_run3	English	23.892
308M+LT_64-dyck_run1	Czech	23.922
308M+LT_64-shuff-dyck_run2	Danish	23.924
308M+LT_64-dyck_run2	Czech	23.924
308M+LT_64-shuff-dyck_run3	Finnish	23.936
308M+LT_64-dyck_run3	Finnish	23.936
308M+LT_128-shuff-dyck_run2	Finnish	23.936
308M+LT_128-shuff-dyck_run3	Finnish	23.936
154M+LT_128-dyck_run3	Albanian	23.959
308M+LT_128-dyck_run2	Czech	24.916
154M+LT_128-shuff-dyck_run1	Czech	25.920
308M+LT_64-shuff-dyck_run3	Danish	25.920
308M+LT_64-dyck_run2	Danish	25.923
308M+LT_64-shuff-dyck_run1	Danish	25.924
308M+LT_64-shuff-dyck_run1	Finnish	25.927
308M+LT_128-shuff-dyck_run3	Czech	25.927
154M+LT_64-dyck_run3	Finnish	25.931
308M+LT_128-dyck_run2	Finnish	25.935
308M+LT_128-dyck_run3	Finnish	26.927
308M+LT_128-dyck_run1	Czech	27.921
154M+LT_128-dyck_run3	Danish	27.923
308M+LT_128-dyck_run3	Czech	28.920
308M+LT_64-dyck_run3	Czech	28.920
154M+LT_64-shuff-dyck_run3	Danish	28.923
154M+LT_64-shuff-dyck_run1	Danish	28.924

Model	Language	Gain % ↑
154M+LT_64-dyck_run1	Danish	28.924
154M+LT_64-shuff-dyck_run2	Danish	29.917
154M+LT_128-dyck_run2	Danish	29.917
154M+LT_64-dyck_run2	Danish	29.917
154M+LT_128-dyck_run3	Finnish	29.929
154M+LT_128-dyck_run1	Danish	30.923
154M+LT_64-dyck_run3	Danish	31.920

C Description of Linguistic Features

We estimate 11 linguistic metrics from 1M token samples per language from training and test data. One additional exception feature is ⑩ **Morphological richness** which is estimated on the Bible dataset (Christodouloupoulos and Steedman, 2015), covering languages in a multi-parallel manner.

Below is a detailed list of linguistic features we analyze. Most features are considered in two variants: (1) punctuation is taken into account when processing the data (`*_with_punct`); and (2) punctuation is ignored (`*_without_punct`). The head of a punctuation token is typically the syntactic head of the clause or the phrase it punctuates, which can make the properties replicate measures of sentence length rather than providing structural insights.

We use the multi-lingual `gemma-3-12B-pt` model (Team et al., 2025) to measure ⑪ **Average perplexity** and ⑫ **Variance of the perplexity** to quantify the predictability and repetitiveness of data (i.e., higher values should indicate more diverse data).

① **Percentage of crossing dependencies:** ratio of crossing dependent pairs to all dependent pairs.
(`%crossing_dep_with_punct`; `%crossing_dep_without_punct`)

② **Mean dependency distance:** average distance between the head and the child in the dependent pair.
(`mean_dep_dist_with_punct`; `mean_dep_dist_without_punct`)

③ **Maximum dependency distance:** maximum distance between the head and the child in the dependent pair.
(`max_dep_dist_with_punct`; `max_dep_dist_without_punct`)

④ **Mean dependency tree depth:** average depth of the dependency tree.
(`mean_tree_depth_with_punct`; `mean_tree_depth_without_punct`)

⑤ **Maximum dependency tree depth:** maximum depth of the dependency tree.
(`max_tree_depth_with_punct`; `max_tree_depth_without_punct`)

⑥ **Vocabulary size:** the number of unique tokens.
(`vocab_size_with_punct`; `vocab_size_without_punct`)

⑦ **Average sentence length:** average number of tokens in the sentence.
(`avg_sentence_length_with_punct`; `avg_sentence_length_without_punct`)

⑧ **Average number of children per head:** average number of children per head across all dependency pairs.
(`mean_children_with_punct`; `mean_children_without_punct`)

⑨ **Maximum number of children per head:** maximum number of children per head across all dependency pairs.
(`max_children_with_punct`; `max_children_without_punct`)

⑩ **Morphological richness:** the number of distinct word forms seen in a parallel corpus relative to English (Bojar, 2015).
(`morph_richness`)

⑪ **Average perplexity:** the average of perplexities measured on batches of 2,048 tokens.
(`avg_perplexity`)

⑫ **Variance of the perplexity:** the variance of perplexities measured on batches of 2,048 tokens.
(`var_perplexity`)

D Details of Language Specific Analysis

Figures 13 and 14 provide the average correlations between efficiency gain across models pretrained on {64,128}-Shuffle Dyck and linguistic features (see Appendix C) estimated on evaluation and training data, respectively.

Figures 15 and 16 provide the average correlations between efficiency gain across models pretrained on {64,128}-Dyck and linguistic features (see Appendix C) estimated on evaluation and training data, respectively.

Figures 17 and 18 provide the average correlations between efficiency gain across models with the Gemma tokenizer and linguistic features (see Appendix C) estimated on evaluation and training data, respectively.

Figures 19 and 20 provide the average correlations between efficiency gain across models with the Llama tokenizer and linguistic features (see Appendix C) estimated on evaluation and training data, respectively.

Figures 21 and 22 provide the correlations between efficiency gain across experiments and lin-

guistic features (see Appendix C) estimated on evaluation and training data, respectively.

Linguistic features are sorted from top to bottom and from left to right (i.e., higher variance at the bottom right side) for Figures 13 to 20, and from left to right (i.e., higher variance at the right side) for Figures 21 and 22. They are sorted based on their reliability which is quantified by the variance of the feature value across selected natural languages.

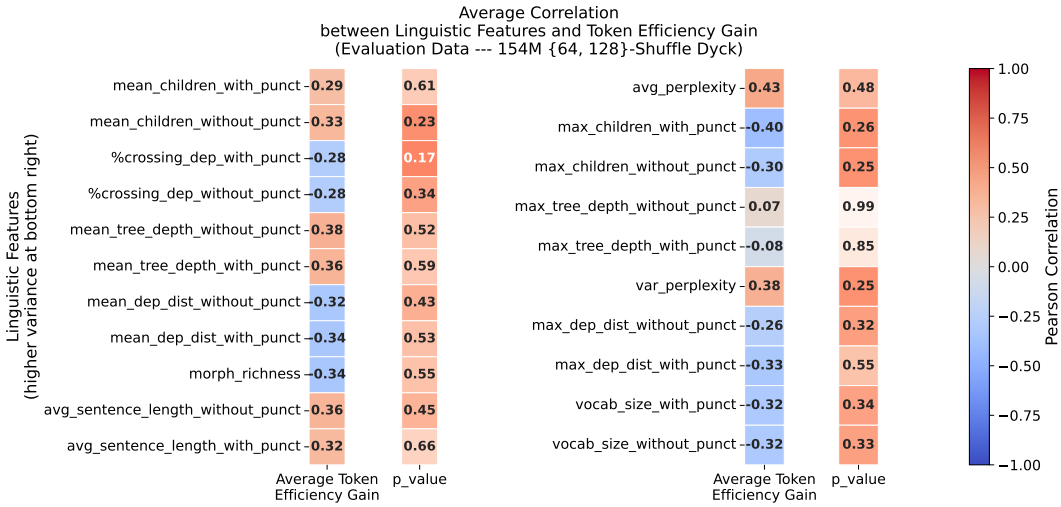


Figure 13: Average correlation between linguistic features of evaluation data and token efficiency gain of 154M models pretrained on {64,128}-Shuffle Dyck. `p-values` have been corrected with the false discovery rate method.

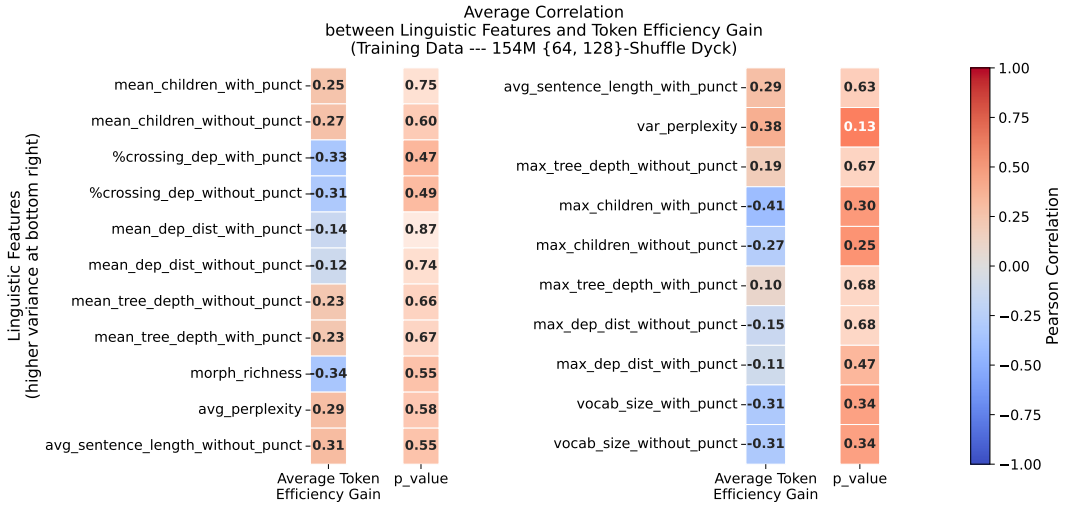


Figure 14: Average correlation between linguistic features of training data and token efficiency gain of 154M models on {64,128}-Shuffle Dyck. `p-values` have been corrected with the false discovery rate method.

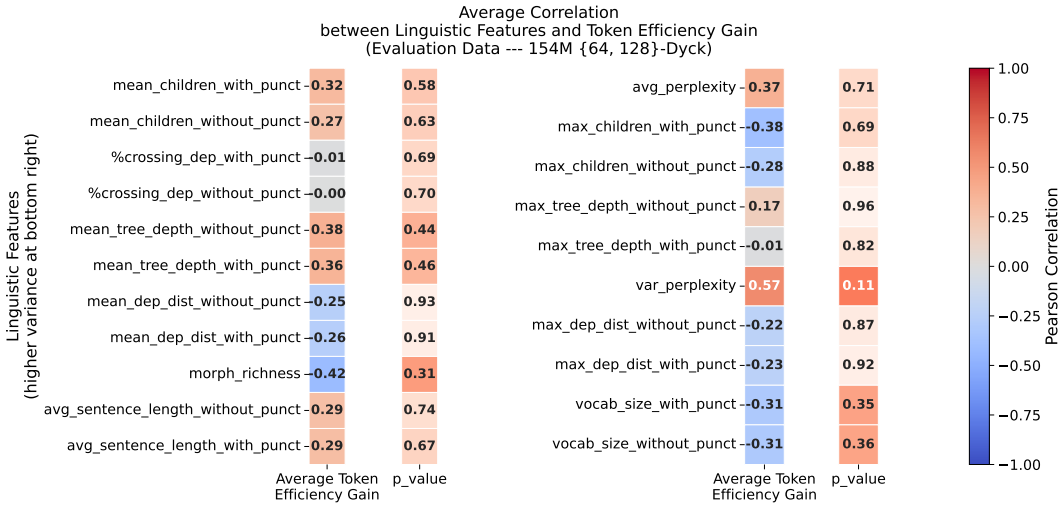


Figure 15: Average correlation between linguistic features of evaluation data and token efficiency gain of 154M models pretrained on {64,128}-Dyck. `p-values` have been corrected with the false discovery rate method.

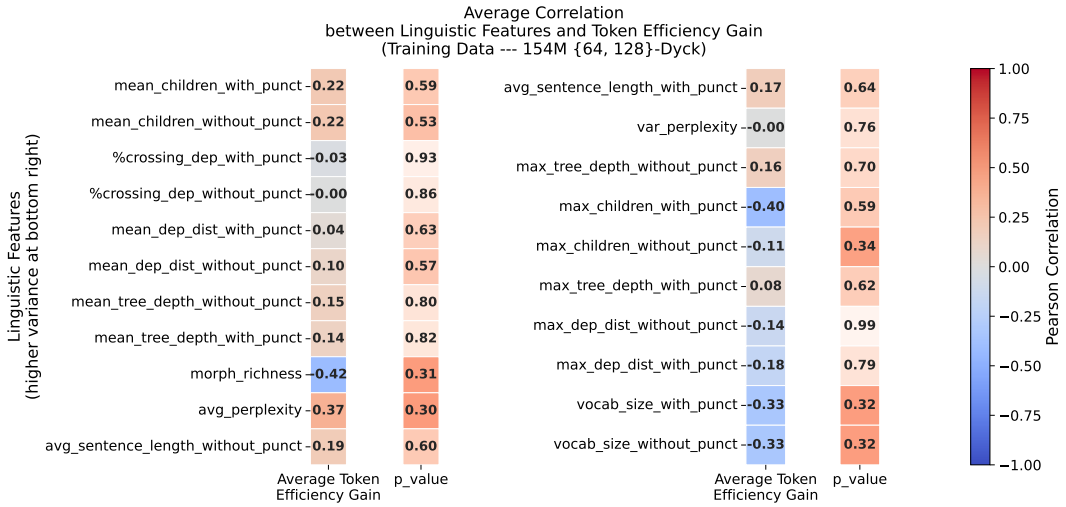


Figure 16: Average correlation between linguistic features of training data and token efficiency gain of 154M models on {64,128}-Dyck. `p-values` have been corrected with the false discovery rate method.

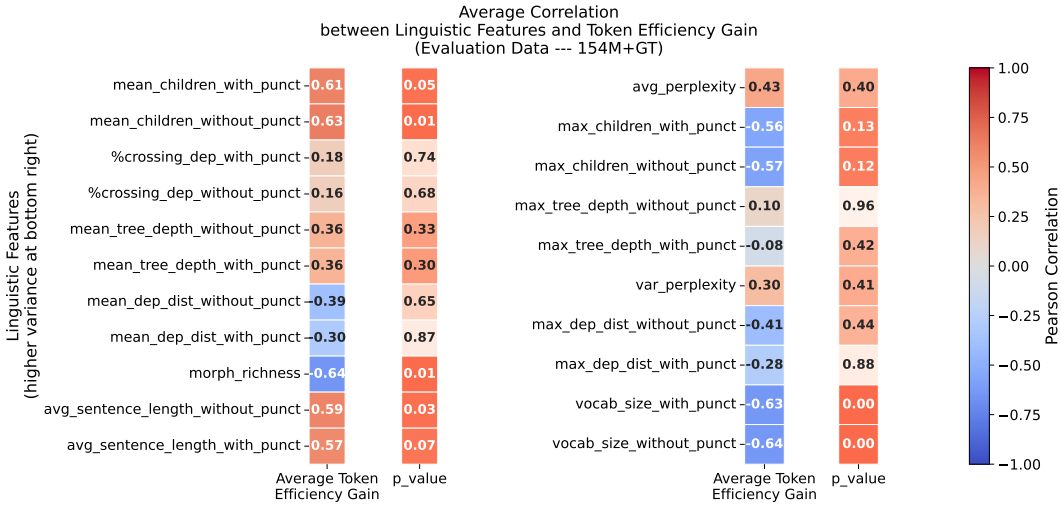


Figure 17: Average correlation between linguistic features of evaluation data and token efficiency gain of 154M models with the Gemma tokenizer. `p-values` have been corrected with the false discovery rate method.

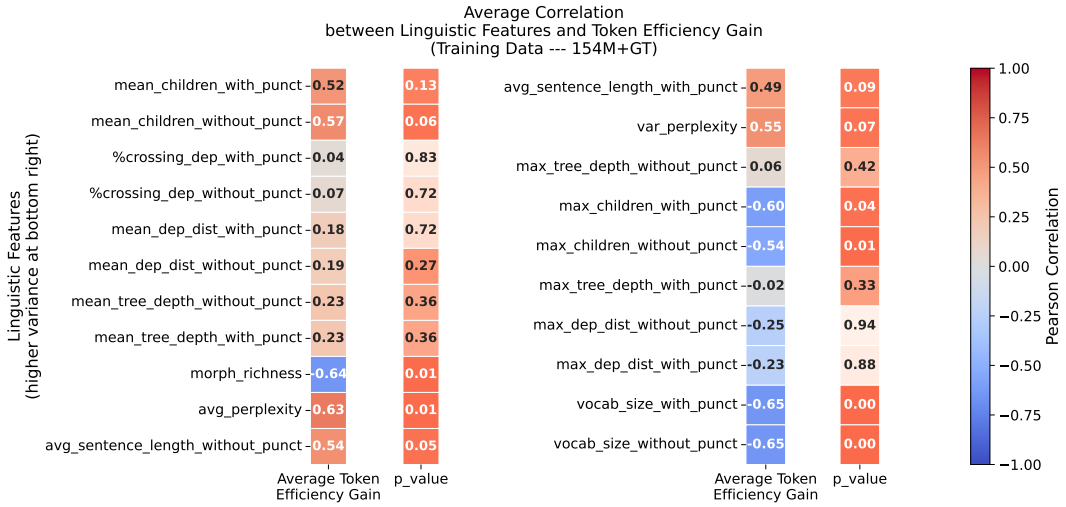


Figure 18: Average correlation between linguistic features of training data and token efficiency gain of 154M models with the Gemma tokenizer. `p-values` have been corrected with the false discovery rate method.

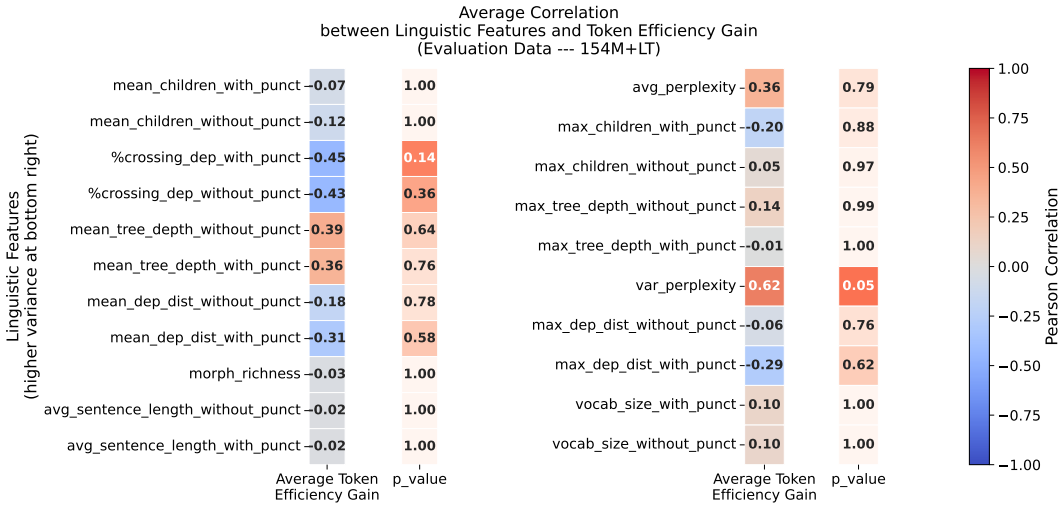


Figure 19: Average correlation between linguistic features of evaluation data and token efficiency gain of 154M models with the Llama tokenizer. `p_values` have been corrected with the false discovery rate method.

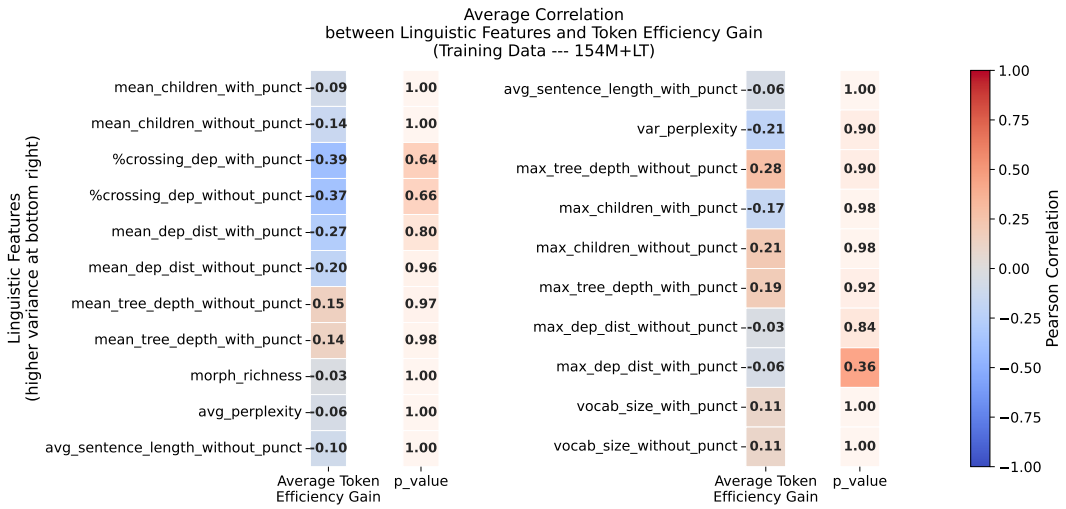


Figure 20: Average correlation between linguistic features of training data and token efficiency gain of 154M models with the Llama tokenizer. `p-values` have been corrected with the false discovery rate method.

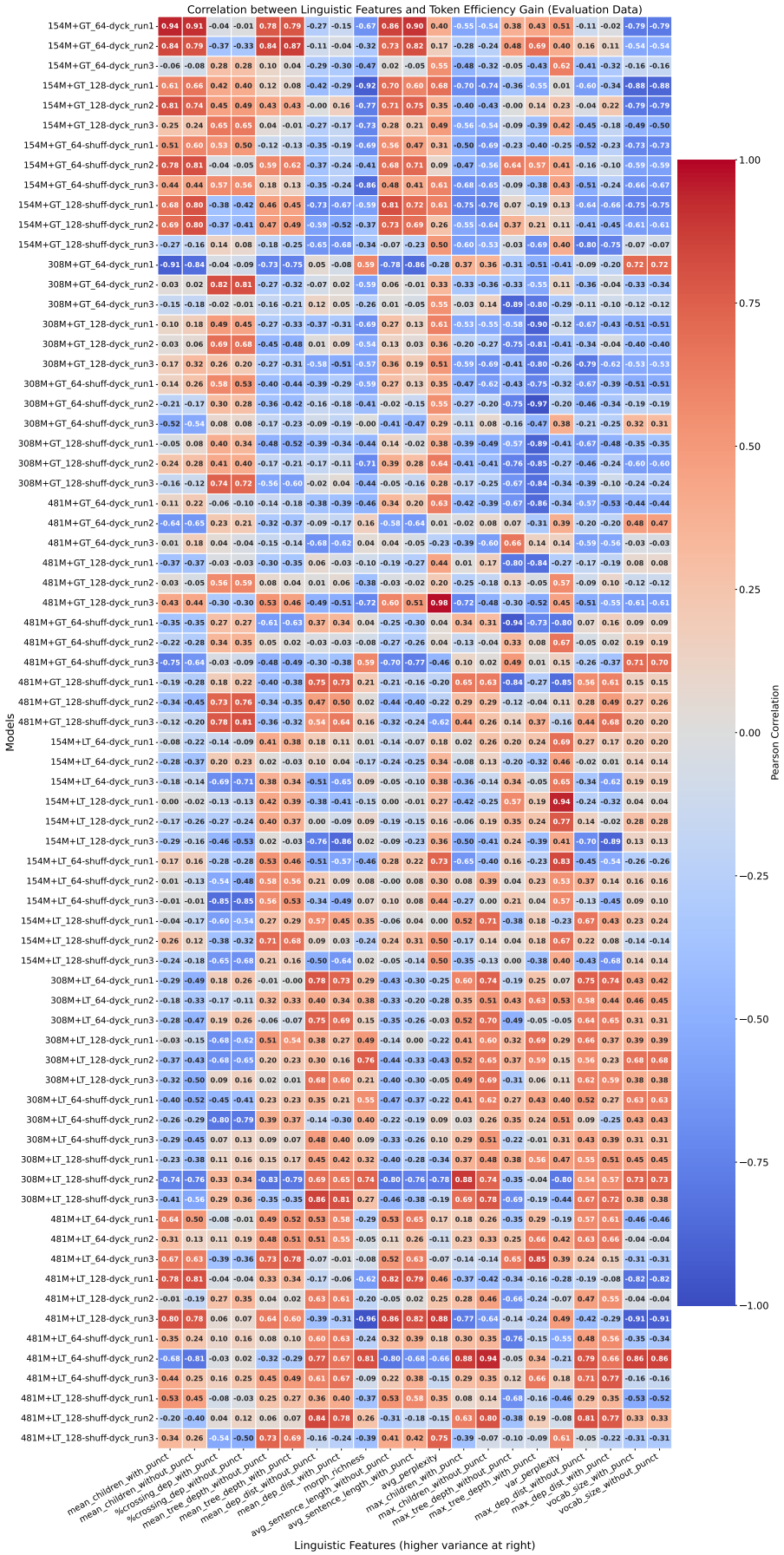


Figure 21: Correlation between linguistic features and token efficiency gain (evaluation data)

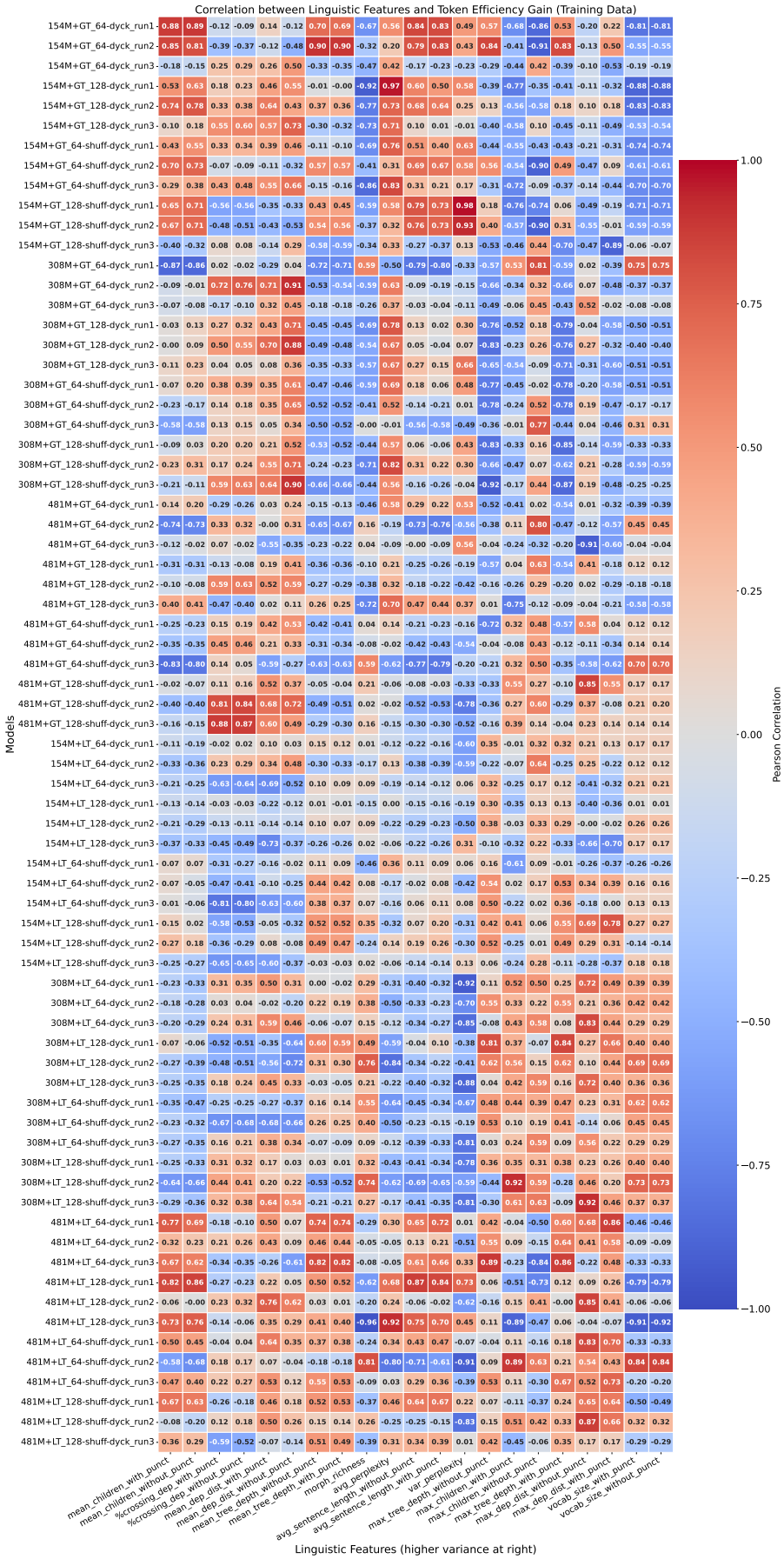


Figure 22: Correlation between linguistic features and token efficiency gain (training data)