LLMs on the Line: Data Determines Loss-to-Loss Scaling Laws

Anonymous authors
Paper under double-blind review

Abstract

Scaling laws guide the development of large language models (LLMs) by offering estimates for the optimal balance of model size, tokens, and compute. More recently, loss-to-loss scaling laws that relate losses across pretraining datasets and downstream tasks have emerged as a powerful tool for understanding and improving LLM performance. In this work, we investigate which factors most strongly influence loss-to-loss scaling. Our experiments reveal that the pretraining data and tokenizer determine the scaling trend. In contrast, model size, optimization hyperparameters, and even significant architectural differences, such as between transformer-based models like Llama and state-space models like Mamba, have limited impact. Consequently, practitioners should carefully curate suitable pretraining datasets for optimal downstream performance, while architectures and other settings can be freely optimized for training efficiency.

1 Introduction

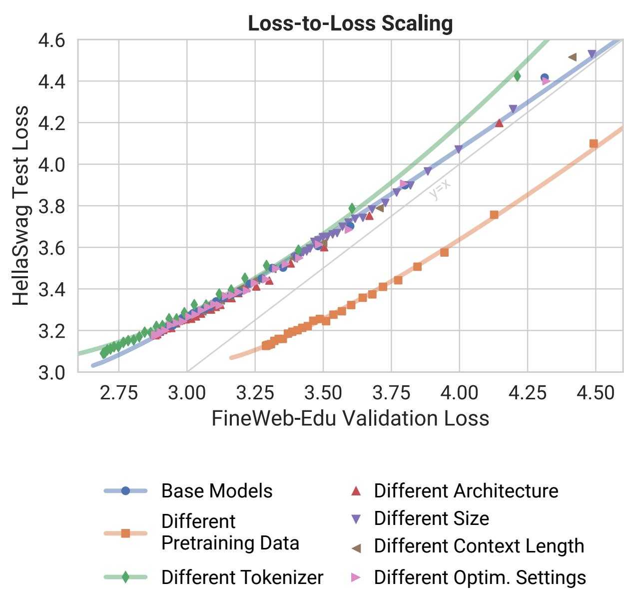

Figure 1: **LLMs' loss-to-loss scaling follows power laws primarily shaped by the choice of pretraining data and tokenizer**. Using Llama trained on FineWeb-Edu as a baseline, we intervene on various factors to assess their impact on train-to-test loss scaling. Changing the pretraining data has the largest effect, followed by the choice of tokenizer. Switching the architecture, e.g., from Llama to Mamba, has limited impact, while factorslike model size, context length, and optimizer settings exert little-to-no influence.

Scaling laws have long guided Large Language Model (LLM) pretraining, determining model and data size under a fixed compute budget (Kaplan et al., 2020; Hoffmann et al., 2022; Grattafiori et al., 2024). Typically, scaling laws relate model performance, usually measured as training or validation loss, to total compute measured in floating point operations (FLOPs). FLOPs account for both parameter count and the number of training tokens. While useful for pretraining, scaling laws do not capture how well a model ultimately performs on downstream tasks (Gadre et al., 2024; Schaeffer et al., 2024; Du et al., 2025). Consequently, multiple works have begun to investigate *downstream scaling laws*: Scaling laws that directly predict downstream loss from FLOPs (Schaeffer et al., 2024; Gadre et al., 2024).

Brandfonbrener et al. (2024) show that *downstream scaling laws* can be decomposed into compute-to-train-loss scaling laws and (train)-loss-to-(test)-loss scaling laws. The combination of *compute-to-loss* and *loss-to-loss* scaling laws enables efficient and accurate prediction of a model's downstream performance. Moreover, holistic *downstream scaling laws* often optimize for a single task or average performance across tasks (Gadre et al., 2024; Schaeffer et al., 2024), whereas *loss-to-loss* (especially test-to-test) scaling laws can help tune a model's performance across a broader range of downstream tasks, e.g., to ensure broad or robust generalization.

While the impact of design choices like pretraining distribution, architecture, tokenizer, optimizer settings, etc. on compute-to-loss scaling laws is fairly well understood (Kaplan et al., 2020; Hoffmann et al., 2022; Tay et al., 2022; Wang et al., 2024; Porian et al., 2025; Du et al., 2025), a similar understanding is missing for loss-to-loss scaling laws. To close this gap, we extend the work of Brandfonbrener et al. (2024); Du et al. (2025), which analyze loss-to-loss relationships within a single architectural and training setup. Adding to that, our study systematically explores how multiple factors influence scaling laws across a diverse range of architectures and training configurations.

Our analysis additionally draws inspiration from a body of work in robustness evaluation of vision (and later language) models (Taori et al., 2020; Miller et al., 2021; Fang et al., 2022; Awadalla et al., 2022). These works show that model performance on different distributions is frequently strongly correlated, and most model and training settings have little-to-no impact on the task-to-task scaling trend of model performance. We treat loss-to-loss curves similarly and perform a series of interventions using over 6000 model checkpoints to understand what design choices causally affect scaling trends.

We make three main observations, illustrated in Fig. 1:

1. LLMs' loss-to-loss scaling consistently follows shifted power laws.
2. Pretraining data and tokenizer are the most salient factors for these scaling laws.
3. In contrast, architecture plays a minor role, while model size, context length, and optimizer settings have negligible impact on loss-to-loss scaling.

Further, we put our observations in the context of downstream scaling laws and discuss the relationship between loss-to-loss and compute-to-loss scaling laws. Our results indicate that different LLM architectures might encode very similar inductive biases, freeing practitioners to optimize architectures for training efficiency without adversely affecting downstream scaling laws.

2 From Scaling Laws to Interventions

Compute-to-Train Scaling Laws Scaling laws aim to optimize model size and token allocation within a fixed compute budget (expressed in FLOPs) by modeling the relationship between parameters, training tokens, and training loss (Hestness et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022). However, these laws are inherently shaped by the data distribution, architecture, and optimization settings (Tay et al., 2022; Wang et al., 2024; Brandfonbrener et al., 2024; Porian et al., 2025), making their application across setups non-trivial.

Compute-to-Downstream Scaling Laws Recent works extend scaling laws to directly predict downstream task performance from compute (Gadre et al., 2024; Isik et al., 2024; Du et al., 2025). While some initial works attempt to map compute budgets to accuracy on individual tasks, multiple tasks, or aggregate benchmarks, this mapping is usually noisy due to several transformations in the accuracy computation that degrade the statistical relationship (Schaeffer et al., 2024). More recent efforts instead use the model's average loss on the correct answers of the task as a proxy (Madaan et al., 2024; Brandfonbrener et al., 2024). Such compute-to-downstream scaling laws provide a more practical perspective on scaling but are still specific to a given training setup.

Loss-to-Loss Scaling Laws Loss-to-loss scaling laws aim to improve the transferability of scaling insights between training setups by examining the relationship between training (or validation) and test losses, between different validation losses, or between different test losses (Brandfonbrener et al., 2024). This perspective is crucial for several reasons. First, train-to-train (or validation-to-validation) scaling implies how scaling laws transfer across datasets (Brandfonbrener et al., 2024). Second, incorporating train-to-test (or validation-to-test) scaling laws alongside compute-to-train scaling laws provides more precise insight into how compute budgets translate to downstream performance and can help study emergent abilities of models (Du et al., 2025). Third, while compute-to-loss scaling laws often target a single downstream task or average task performance, train-to-test and test-to-test scaling laws can help tune a model's performance across diverse tasks, e.g., to foster the development of generalist LLMs with a balanced task performance.

Accuracy on the Line Our work is inspired by robustness research in image classification. Prior studies (Taori et al., 2020; Miller et al., 2021; Fang et al., 2022) demonstrate a strong and consistent correlation between in-distribution and out-of-distribution (OOD) *accuracy* across various image classification models and settings. We are not the first to observe the similarity to LLMs, where recent works (Gadre et al., 2024; Brandfonbrener et al., 2024; Du et al., 2025) highlight strong scaling trends (linear or power-law-like) between *losses*. However, these studies are typically constrained to a single architecture or training setup. In contrast, we examine trends across a wide range of architectures and training conditions (see §4), showing for the first time that loss-to-loss scaling follows consistent laws across settings.

Robustness Interventions Accuracy-to-accuracy relationships in the vision, vision-language, and language domain have also been used to study how scaling laws shift under robustness interventions like dataset size, adversarial training, architectural details, loss functions, supervision type, or OOD shifts (Taori et al., 2020; Fang et al., 2022; Awadalla et al., 2022; Mayilvahanan et al., 2024a;b; Wiedemer et al., 2024). For vision-language models, Taori et al. (2020); Fang et al. (2022) find that most interventions do not impact OOD performance; only increasing data diversity has a significant positive impact. Their findings suggest that curating better datasets is crucial for training vision and vision-language models that generalize broadly.

Motivated by these insights, we aim to uncover the factors determining loss-to-loss scaling to guide practitioners in developing models for specific downstream performance. Our insights complement the findings from Awadalla et al. (2022), who show that accuracy-accuracy scaling trends in comprehension tasks are agnostic to architecture type (e.g., encoder-only, encoder-decoder, decoder-only) after fine-tuning. In contrast to their study, we focus on zero-shot generalization across a diverse set of tasks, specifically investigating state-of-the-art decoder-only architectures such as GPT Radford et al. (2019), Llama (Grattafiori et al., 2024), and Mamba Gu & Dao (2024); Dao & Gu (2024).

3 Fitting Loss-to-Loss Scaling Laws

We focus our analysis on train-to-train and train-to-test scaling. Combined with known compute-to-train scaling laws, these loss-to-loss scaling laws paint a complete picture of a model's downstream performance given a compute budget and characterize a model's downstream performance distribution across tasks (Brandfonbrener et al., 2024).

As is standard in the recent literature, we report test loss as a proxy for downstream performance. Following Brandfonbrener et al. (2024); Madaan et al. (2024); Schaeffer et al. (2024), we track the test loss as a model's loss on only the correct answer given the question as context. This is sometimes called the *cloze formulation* of a task since the model is essentially evaluated on its ability to fill in blanks.

Brandfonbrener et al. (2024) predict train-to-train and train-to-test scaling laws to follow a shifted power law[1]

$$L_y\left(f_p^{N,D}\right) \approx K \cdot \left(L_x\left(f_p^{N,D}\right) - E_{x|p}\right)^{\kappa} + E_{y|p}, \tag{1}$$

where L_x, L_y are the losses on datasets $\mathcal{D}_x, \mathcal{D}_y$ shown on the x- and y-axis. $f_p^{N,D}$ is a model trained with N parameters on D tokens on the pretraining set $\mathcal{D}_p$, and K and κ are parameters to be fit. $E_{x|p}, E_{y|p}$ are the irreducible errors (i.e., minimum loss) that f_p trained on $\mathcal{D}_p$ can achieve on the datasets $\mathcal{D}_x, \mathcal{D}_y$.

Given a model configuration, we obtain sufficient samples to fit all parameters by recording checkpoints throughout training and across seeds. We first estimate $E_{x|p}, E_{y|p}$ from individual compute-to-loss scaling laws and then fit K, κ.

Note that to study the impact of various training settings, we show losses of *the same model* on *different datasets* on the x- and y-axis. This should not be confused with some curves in Brandfonbrener et al. (2024) that showed the losses of *different* compute-matched models on the x- and y-axis.

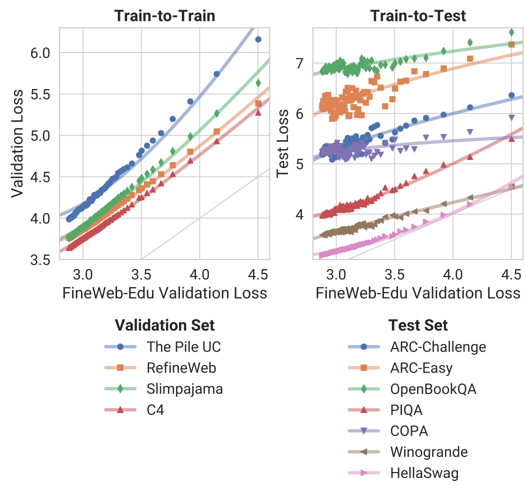

Figure 2: **Loss-to-loss scaling consistently obeys power laws.** We extend results from Brandfonbrener et al. (2024) by many architectures, training settings, and validation/test sets. We show illustrative shifted power laws for Mamba trained on FineWeb-Edu here; more configurations can be found in App. C.

With this setup, we can now analyze the loss-to-loss scaling laws of models trained with different configurations. Brandfonbrener et al. (2024) only showed loss-to-loss scaling a single architecture with a fixed training recipe: Olmo (Groeneveld et al., 2024). We extend their analysis by multiple architectures, pretraining sets, tokenizers, and training settings, all listed in §4. As an illustrative example, we show loss-to-loss scaling for Mamba trained on FineWeb-Edu in Fig. 2; more examples with additional test sets are listed in App. C and throughout §4.

Overall, across training setups—defined by a model, dataset, tokenizer, and optimization hyperparameters—shifted power laws describe loss-to-loss scaling well (Eq. (1)). On some datasets, the performance of models with high loss (towards the top right of each curve) is not captured perfectly by the power law formulation proposed by Brandfonbrener et al. (2024). This is unsurprising, given that these data points typically represent models in early training stages but might hint at a refined formulation of Eq. (1) for the high-loss regime.

Note also that loss-to-loss scaling follows a power law even for datasets on which the model never reaches high accuracy. E.g., Mamba in Fig. 2 never surpasses chance performance on ARC-Challenge and OpenBookQA, yet Eq. (1) describes the test loss equally well. This underlines the usefulness of loss-to-loss scaling laws to study model behavior.

Takeaway 1 Across architectures and training settings, loss-to-loss scaling generally follows a shifted power law as described in Eq. (1).

4 A Causal Analysis of Loss-to-Loss Scaling

We now perform interventions on the model and training configurations to find what factors cause the exact shape of loss-to-loss scaling laws.

Our basic procedure is outlined in Fig. 3. As mentioned in §2, our approach is motivated by similar studies in the robustness literature. In contrast to that setting, we here lack paired in-distribution and

[1] Brandfonbrener et al. (2024) do not state a train-to-train scaling law for non-compute-matched data. Our form here follows from their Eq. 4 when assuming an irreducible error as in Eqs. 6, 7.

out-of-distribution datasets. Instead, we simply consider all combinations of validation and test sets. For ease of visualization when intervening on the pretraining data, we always show FineWeb-Edu validation loss on the x-axis, even for models trained on different pretraining distributions. This choice is arbitrary and does not affect our results; see App. D. Similarly, we here report results for scaling laws of *average* validation and test loss; results for individual losses can be found in App. E.

For our analysis, we consider the impact of pretraining data, tokenizer, architecture, model size, context length, and optimizer settings.

Pretraining Sets Our models are trained on FineWeb-Edu (Penedo et al., 2024), C4 (Dodge et al., 2021), and an uncopyrighted version of The Pile dubbed The Pile UC. Some models from Hugging Face are trained on the original version of The Pile (Gao et al., 2020) and The Pile Deduped (Biderman et al., 2023), a deduplicated version.

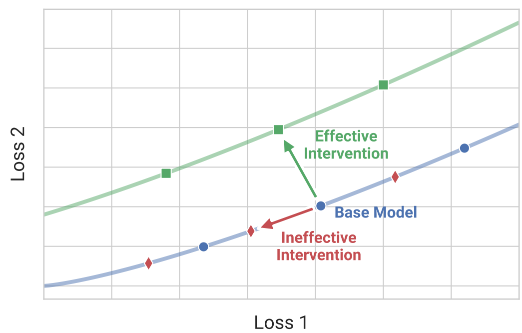

Figure 3: **Schematic of our causal analysis**. Checkpoints of a **base model** trained on different numbers of tokens and with different seeds lie on the same loss-to-loss line. Better-performing models (typically with higher compute) lie closer to the origin. We intervene on training settings (e.g., pretraining data, architecture) and retrain from scratch, yielding new models whose checkpoints again constitute lines. An **effective intervention** produces models on a new line; an **ineffective intervention** yields models that lie on the base line.

Validation Sets Models are evaluated on 5000 sequences sampled from the validation sets of FineWeb-Edu, C4, The Pile, RefinedWeb (Penedo et al., 2023), and SlimPajama (Shen et al., 2024).

Test Sets We use LM Harness framework (Gao et al., 2024) to assess model performance on HellaSwag (Zellers et al., 2019), COPA (Gordon et al., 2011), WinoGrande (Sakaguchi et al., 2019), PIQA (Bisk et al., 2019), OpenBookQA (Mihaylov et al., 2018), as well as ARC-Easy and ARC-Challenge (Clark et al., 2018).

Architectures We train Llama-3 (Grattafiori et al., 2024) with 417 M parameters and Mamba (Gu & Dao, 2024) with 420 M parameters using the Lingua framework (Videau et al., 2024), following Chinchilla scaling laws (Hoffmann et al., 2022). We supplement our analysis with pretrained GPT (Black et al., 2021; 2022; Biderman et al., 2023), Llama (Penedo et al., 2024), and Mamba Gu & Dao (2024); Dao & Gu (2024) variants from Hugging Face (Wolf et al., 2020).

Tokenizers We train Llama and Mamba with either a `tiktoken` tokenizer (128 k vocabulary size) or the `gpt2` tokenizer (50 257 vocabulary size). Pretrained models from Hugging Face use an almost identical GPT-2 tokenizer, dubbed `gpt2-HF`. This version does not explicitly pad text with beginning and end-of-sequence tokens. A few Hugging Face GPT models instead use the `gpt-neox` tokenizer with a slightly different vocab size of 50 254, which results in a different internal mapping compared to `gpt2`,

4.1 Pretraining Data, Tokenizer, and Architecture

First, we jointly examine the effect of pretraining data, architecture, and tokenizer. Since we face limited compute to train models from scratch, we do not have checkpoints for all possible combinations of these factors. Instead, we analyze the effect of an intervention on each factor when matching models in the two other factors. Note that we do not have sufficient checkpoints for some Hugging Face models to fit a power law. Nevertheless, in all these cases, the available data points follow a clearly discernible trend.

Effect of Pretraining Data Fig. 4 illustrates the substantial impact pretraining data has on loss-to-loss scaling. Across architectures and compute (in different columns), changing the pretraining data leads to a large shift in the loss-to-loss curve. The only exception is the last column, where we

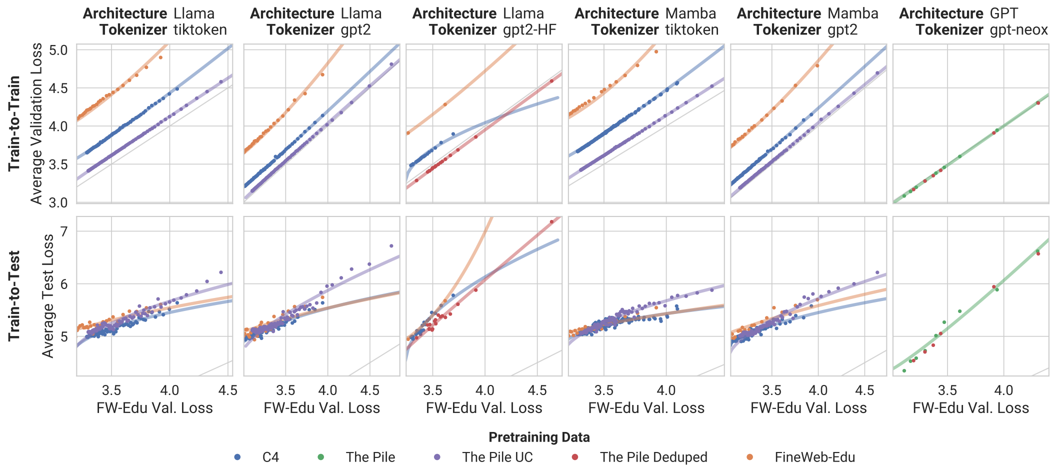

Figure 4: **Pretraining data has a substantial impact on loss-to-loss scaling laws**. Models are matched on architecture and tokenizer.

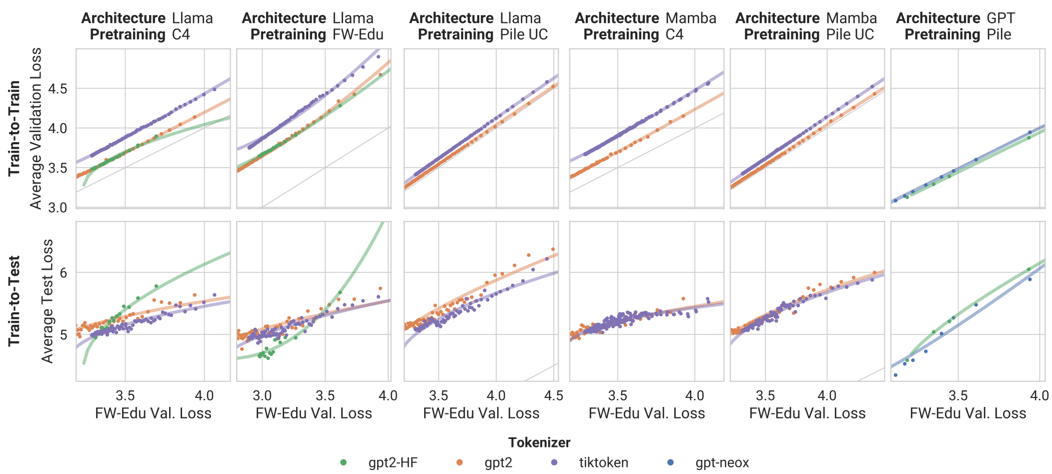

Figure 5: **The tokenizer has a moderate impact on loss-to-loss scaling laws**. Models are matched on pretraining data and architecture.

compare Hugging Face models trained on The Pile and a deduplicated version. Models trained on either version lie on the same curve, suggesting that the deduplication procedure successfully reduced the dataset size while producing a similar distribution that does not significantly impact loss-to-loss scaling.

> **Takeaway 2** With fixed architecture and tokenizer, changing the pretraining data leads to *substantial* shifts in loss-to-loss scaling laws.

Effect of Tokenizer Fig. 5 shows that the tokenizer, too, affects loss-to-loss scaling laws, albeit less strongly than pretraining data. It is interesting to see how slight deviations in the tokenizer can have a pronounced effect, particularly for train-to-train scaling laws. While the slight vocabulary size difference between `gpt2-HF` and `gpt-neox` has little impact on loss-to-loss scaling (last column), the different handling of special tokens in `gpt2` and `gpt2-HF` does. To the best of our knowledge, this effect has not been observed before and could be explored in future work.

> **Takeaway 3** With fixed architecture and pretraining data, changing the tokenizer leads to *moderate* changes in loss-to-loss scaling laws.

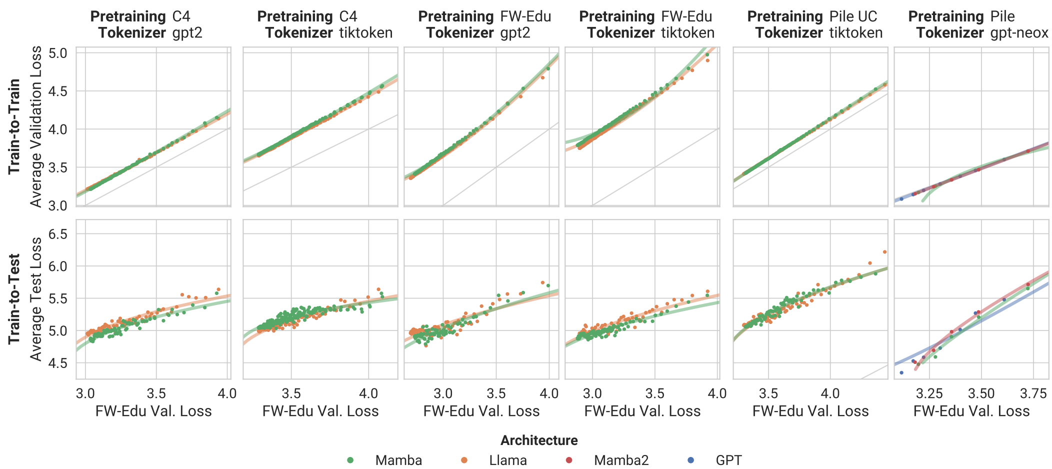

Figure 6: **Architecture has limited impact on loss-to-loss scaling laws**. Models are matched on pretraining data and tokenizer.

Effect of Architecture Lastly, Fig. 6 illustrates that changing the architecture results in only very slight changes in the loss-to-loss curves across pretraining data and tokenizer settings. Unlike pretraining data and tokenizer, architecture has little influence on train-to-train and train-to-test scaling. This is particularly surprising given the significant architectural differences between Llama or GPT (transformer-based models) and Mamba (a state-space model). These results raise an important question: Do current architectures encode distinct inductive biases or converge to similar solutions given the same training data? Further research is needed to understand the implications of this finding.

> **Takeaway 4** With fixed pretraining data and tokenizer, changing the architecture has *limited* impact on loss-to-loss scaling laws — raising questions about the distinctiveness of their inductive biases.

4.2 Model Size, Context Length, and Optimization

We now examine the effect of other common design decisions, such as the number or width of layers, the context length, optimizer, learning schedule, learning rate, and weight decay. In contrast to §4.1, we can perform these interventions separately since we can compare among our own Llama and Mamba models whose training settings are matched by default. To provide a more succinct overview, we only show train-to-train scaling laws in this section; additional train-to-test scaling laws for the same intervention can be found in App. F. We also do not show fitted power laws here since we display many more models per plot than in §4.1, and the scaling trends are clearly discernible.

Effect of Model Size We first examine the influence of model size by training Llama and Mamba models with varying depths and widths (see App. B for details). Fig. 7 shows the results: Despite significant differences in parameter count, the loss-to-loss scaling trends remain unchanged. These findings align well with Du et al. (2025), who observed that model size has little effect on loss-to-loss scaling for GPT models. We extend this conclusion to Llama and Mamba and across multiple pretraining distributions.

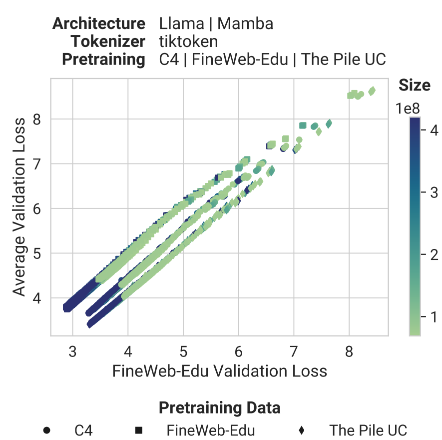

Figure 7: **Model size does not affect loss-to-loss scaling**. The distinct lines correspond to different pretraining distributions (see Fig. 4), reinforcing that their influence is consistent across scales.

Effect of Context Length We next investigate the effect of varying the context length between 1024, 2048, and 3076 tokens. As shown in Fig. 8, this change does not meaningfully affect the loss-to-loss scaling curves.

Effect of Optimization Settings Finally, we evaluate a range of common optimization settings: We consider the Adam (Kingma & Ba, 2017) and AdamW (Loshchilov & Hutter, 2019) optimizers, cosine (Loshchilov & Hutter, 2017) and WSD (Hu et al., 2024) schedules, learning rates of 0.0003 and 0.003, and a weight decay of 0.1 or 0.033. In our training setup, models using the Adam optimizer generally did not converge, and we exclude them from the analysis. Variations of the other settings do not affect loss-to-loss scaling coefficients, as shown in Fig. 9.

Given the limited impact of the factors studied in this section, the conclusions from §4.1 should generalize well across variations in model size, context length, and optimization settings. For example, the substantial impact of the pretraining distribution can also be observed in Figs. 7 and 8.

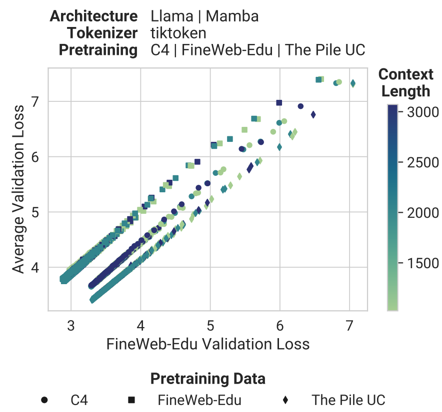

Figure 8: **Context length does not affect loss-to-loss scaling**. Again, distinct lines correspond to different pretraining distributions (compare Fig. 4), validating their consistent impact.

Takeaway 5 Model size, context length, and optimization settings have negligible impact on loss-to-loss scaling laws.

5 Discussion and Future Work

Our findings add to the understanding of loss-to-loss scaling laws and reinforce prior results from vision and vision-language research (Taori et al., 2020; Fang et al., 2022) on the importance of choosing the pretraining data.

Implications for Optimizing Downstream Performance Our results emphasize that the data distribution is the key for achieving a desireable loss-to-loss scaling and a in turn achieve a great downstream performance. Conversely, since architecture has little impact on the train-to-test conversion, it can be freely optimized for better compute scaling without affecting downstream scaling or performance.

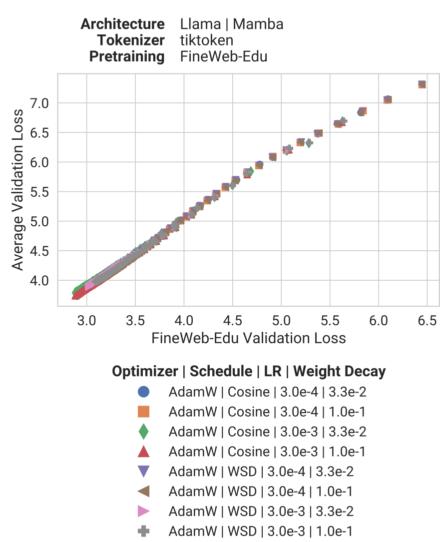

Figure 9: **Optimization settings do not affect loss-to-loss scaling**.

Implications for Balancing Performance If the aim is not only optimal average downstream performance but also a specific weighting between different tasks, e.g., to ensure a balanced downstream performance, individual train-to-test scaling laws can be used to tune a model's performance. Here, too, the pretraining data has the largest impact and practitioners should thus consider the final application of their model already during the data curation stage. Ultimately, our findings underscore that pretraining data curation, rather than architectural innovation, can be the primary driver in developing robust, generalist models.

On Architectural Biases The limited impact of even drastically different architectures on loss-to-loss scaling behavior illustrated in §4.1 and Fig. 6 suggest that architectures trained on the same data may implicitly learn highly similar representations. This might seem intuitive, as all models minimize the same loss function. One might expect them to converge toward comparable solutions when the training loss approaches zero (Roeder et al., 2020). However, even checkpoints of our smaller models, when trained on fewer tokens, follow the same scaling across architectures. Understanding whether this implies representational and behavioral similarity remains an intriguing open question. Beyond this, it remains to be seen whether it is possible to formulate architectures that fit the data well but exhibit different scaling trends.

On New Training Paradigms Our study intentionally focuses on models trained with standard loss functions and conventional training settings to guide practitioners. The limited impact of existing paradigms does not preclude innovative training approaches from improving loss-to-loss scaling. In fact, a recent work by Saunshi et al. (2024) demonstrates that gradually increasing model depth and initializing based on layers from a smaller model produces markedly different scaling behavior, particularly in how perplexity translates to downstream accuracy. Similar structured growth approaches could offer new pathways for improving scaling efficiency and generalization for decoder-only LLMs trained with next-token prediction. We leave this exercise for future work.

On the Exhaustiveness of Interventions in §4.1 Our study clearly distinguishes between factors with substantial and limited impact on loss-to-loss scaling. While our conclusions are inherently shaped by the specific settings we explored, the observed trends provide strong empirical evidence for these distinctions. Given the strong and consistent impact of pretraining data and tokenizer, we can confidently conclude that these interventions affect loss-to-loss scaling. While we observed only a limited impact of the architecture, this effect was also consistent across major state-of-the-art architectures including Llama, GPT, and Mamba — which collectively represent the dominant paradigms in large-scale language modeling. Given this exhaustive set, it is hard to argue that other architectures would meaningfully alter loss-to-loss scaling.

On the Exhaustiveness of Interventions in §4.2 Across the wide range of size configurations (App. B) we test, all models exhibit very consistent loss-to-loss scaling. Similarly, the effect we observed for different context lengths is very consistent within our test range (1024, 2048, 3076), which aligns with commonly used configurations (Black et al., 2021; Wang & Komatsuzaki, 2021; Biderman et al., 2023; Penedo et al., 2024; Black et al., 2022). While we acknowledge the possibility that larger models or longer context lengths could influence loss-to-loss scaling, such an effect — if present — is unlikely. For optimization settings, we again consider configurations widely used in LLM training (Shoeybi et al., 2020; Karpathy, 2022; Videau et al., 2024), including variations in optimizer type, learning rate, weight decay, and scheduling. While our results indicate that these choices do not meaningfully alter loss-to-loss scaling within the explored settings, we acknowledge that the space of optimization techniques is vast, and our list is not exhaustive. It remains possible that a principled optimization strategy, different from current best practices, could induce new scaling behaviors. However, our findings suggest that optimization settings are not a primary driver of loss-to-loss scaling trends within the bounds of conventional language model training.

6 Conclusion

In this work, we systematically investigate loss-to-loss scaling in LLMs, identifying key factors that shape its behavior. Our large-scale interventional analysis — spanning over 6000 model checkpoints across architectures, tokenizers, and training setups — reveals that loss-to-loss scaling consistently follows shifted power-law trends, enabling predicting test performance from training loss.

We identify pretraining data and tokenizer as the dominant factors shaping these scaling laws, highlighting the importance of data curation. Architecture has limited impact, with models as different as LLaMA (transformer-based) and Mamba (a state-space model) exhibiting nearly identical scaling when trained on the same data and tokenizer. Model size, context length, and optimization settings have negligible influence, such that loss-to-loss scaling remains stable across different configurations.

Our findings underline the importance of pretraining data for downstream performance and robustness and suggest that different LLM might share similar architectural biases. Given our observations, practitioners should prioritize curating high-quality pretraining data to optimize downstream performance, while architectures and training settings can be adjusted freely for efficiency.

REFERENCES

Anas Awadalla, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt. Exploring the landscape of distributional robustness for question answering models, 2022. URL https://arxiv.org/abs/2210.12517.

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. Pythia: A suite for analyzing large language models across training and scaling, 2023. URL https://arxiv.org/abs/2304.01373.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. 2021. URL https://api.semanticscholar.org/CorpusID:245758737.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. Gpt-neox-20b: An open-source autoregressive language model, 2022. URL https://arxiv.org/abs/2204.06745.

David Brandfonbrener, Nikhil Anand, Nikhil Vyas, Eran Malach, and Sham Kakade. Loss-to-loss prediction: Scaling laws for all datasets, 2024. URL https://arxiv.org/abs/2411.12925.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL https://arxiv.org/abs/2405.21060.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus, 2021. URL https://arxiv.org/abs/2104.08758.

Zhengxiao Du, Aohan Zeng, Yuxiao Dong, and Jie Tang. Understanding emergent abilities of language models from the loss perspective, 2025. URL https://arxiv.org/abs/2403.15796.

Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (clip), 2022. URL https://arxiv.org/abs/2205.01397.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, Thomas Kollar, Yair Carmon, Achal Dave, Reinhard Heckel, Niklas Muennighoff, and Ludwig Schmidt. Language models scale reliably with over-training and on downstream tasks, 2024. URL https://arxiv.org/abs/2403.08540.

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. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024. URL https://zenodo.org/records/12608602.

Andrew S. Gordon, Zornitsa Kozareva, and Melissa Roemmele. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, 2011. URL https://api.semanticscholar.org/CorpusID:434646.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. URL https://arxiv.org/abs/2312.00752.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically, 2017. URL https://arxiv.org/abs/1712.00409.

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. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm: Unveiling the potential of small language models with scalable training strategies, 2024. URL https://arxiv.org/abs/2404.06395.

Berivan Isik, Natalia Ponomareva, Hussein Hazimeh, Dimitris Paparas, Sergei Vassilvitskii, and Sanmi Koyejo. Scaling laws for downstream task performance of large language models, 2024. URL https://arxiv.org/abs/2402.04177.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Andrej Karpathy. nanogpt, 2022. URL https://github.com/karpathy/nanoGPT.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017. URL https://arxiv.org/abs/1608.03983.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Lovish Madaan, Aaditya K. Singh, Rylan Schaeffer, Andrew Poulton, Sanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. Quantifying variance in evaluation benchmarks, 2024. URL https://arxiv.org/abs/2406.10229.

Prasanna Mayilvahanan, Thaddäus Wiedemer, Evgenia Rusak, Matthias Bethge, and Wieland Brendel. Does clip's generalization performance mainly stem from high train-test similarity?, 2024a. URL https://arxiv.org/abs/2310.09562.

Prasanna Mayilvahanan, Roland S. Zimmermann, Thaddäus Wiedemer, Evgenia Rusak, Attila Juhos, Matthias Bethge, and Wieland Brendel. In search of forgotten domain generalization, 2024b. URL https://arxiv.org/abs/2410.08258.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018. URL https://arxiv.org/abs/1809.02789.

John Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: On the strong correlation between out-of-distribution and in-distribution generalization, 2021. URL https://arxiv.org/abs/2107.04649.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023. URL https://arxiv.org/abs/2306.01116.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale, 2024. URL https://arxiv.org/abs/2406.17557.

Tomer Porian, Mitchell Wortsman, Jenia Jitsev, Ludwig Schmidt, and Yair Carmon. Resolving discrepancies in compute-optimal scaling of language models, 2025. URL https://arxiv.org/abs/2406.19146.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Geoffrey Roeder, Luke Metz, and Diederik P. Kingma. On linear identifiability of learned representations, 2020. URL https://arxiv.org/abs/2007.00810.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/1907.10641.

Nikunj Saunshi, Stefani Karp, Shankar Krishnan, Sobhan Miryoosefi, Sashank J. Reddi, and Sanjiv Kumar. On the inductive bias of stacking towards improving reasoning, 2024. URL https://arxiv.org/abs/2409.19044.

Rylan Schaeffer, Hailey Schoelkopf, Brando Miranda, Gabriel Mukobi, Varun Madan, Adam Ibrahim, Herbie Bradley, Stella Biderman, and Sanmi Koyejo. Why has predicting downstream capabilities of frontier ai models with scale remained elusive?, 2024. URL https://arxiv.org/abs/2406.04391.

Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Zhengzhong Liu, Hongyi Wang, Bowen Tan, Joel Hestness, Natalia Vassilieva, Daria Soboleva, and Eric Xing. Slimpajama-dc: Understanding data combinations for llm training, 2024. URL https://arxiv.org/abs/2309.10818.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020. URL https://arxiv.org/abs/1909.08053.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification, 2020. URL https://arxiv.org/abs/2007.00644.

Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Won Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Q. Tran, Dani Yogatama, and Donald Metzler. Scaling laws vs model architectures: How does inductive bias influence scaling?, 2022. URL https://arxiv.org/abs/2207.10551.

Mathurin Videau, Badr Youbi Idrissi, Daniel Haziza, Luca Wehrstedt, Jade Copet, Olivier Teytaud, and David Lopez-Paz. Meta Lingua: A minimal PyTorch LLM training library, 2024. URL https://github.com/facebookresearch/lingua.

Ben Wang and Aran Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021.

Siqi Wang, Zhengyu Chen, Bei Li, Keqing He, Min Zhang, and Jingang Wang. Scaling laws across model architectures: A comparative analysis of dense and moe models in large language models, 2024. URL https://arxiv.org/abs/2410.05661.

Thaddäus Wiedemer, Yash Sharma, Ameya Prabhu, Matthias Bethge, and Wieland Brendel. Pretraining frequency predicts compositional generalization of CLIP on real-world tasks. In *NeurIPS 2024 Workshop on Compositional Learning: Perspectives, Methods, and Paths Forward*, 2024. URL https://openreview.net/forum?id=NDXoM1wYgl.

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. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.

You may include other additional sections here.

Table 1: Details of the models we trained from scratch.

Architecture	Width	Depth	No. of Parameters (in millions)
Llama	1024	12	416
Llama	1024	8	365
Llama	1024	4	314
Llama	512	12	172
Llama	512	8	158
Llama	512	4	145
Llama	256	12	76
Llama	256	8	72
Llama	256	4	59
Mamba	1024	24	420
Mamba	1024	16	367
Mamba	1024	8	315
Mamba	512	24	172
Mamba	512	16	158
Mamba	512	8	145
Mamba	256	24	76
Mamba	256	16	73
Mamba	256	8	69

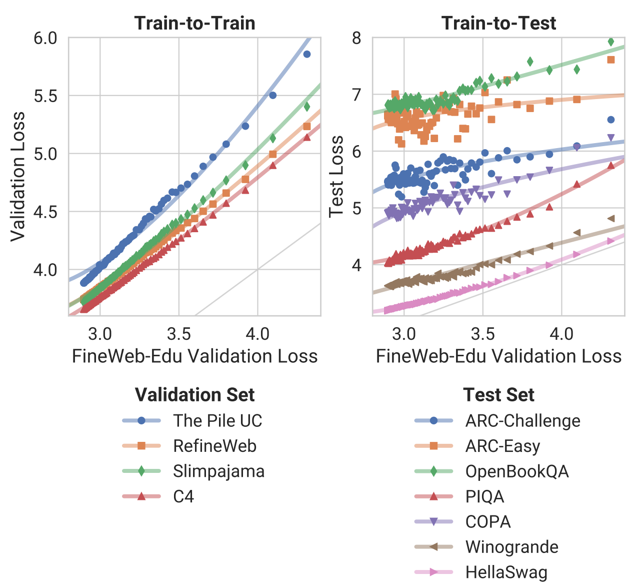

Figure 10: **Loss-to-Loss Scaling for FineWeb-Edu-trained Llama**.

A Appendix

B Model Details

In addition to the models discussed in §4.1, we add additional details of Llama and Mamba models we trained of different depths and widths.

C Loss-to-Loss Scaling Across Settings

We supplement Fig. 2 from §3 with additional architecture-pretraining pairings in Figs. 10 to 15.

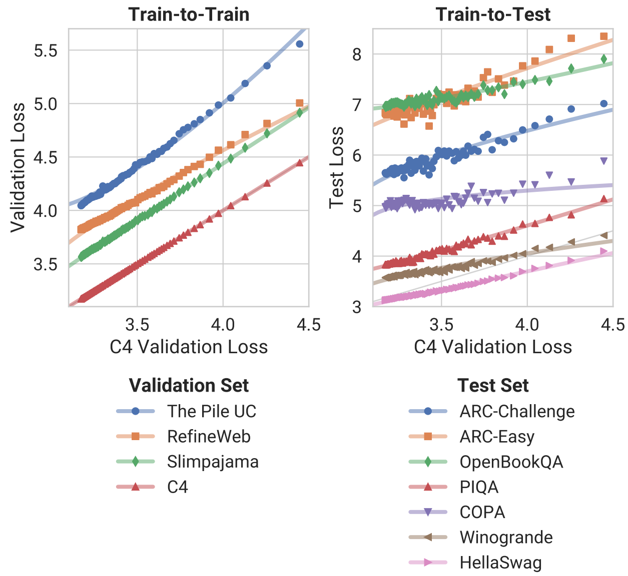

Figure 11: **Loss-to-Loss Scaling for C4-trained Llama**.

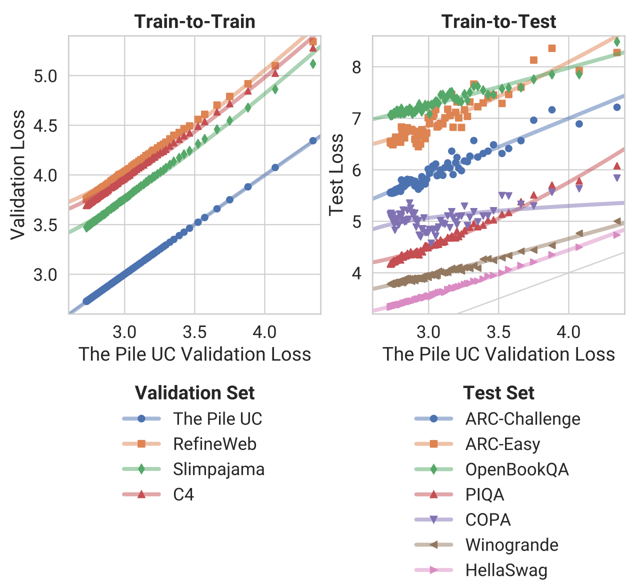

Figure 12: **Loss-to-Loss Scaling for The Pile-trained Llama**.

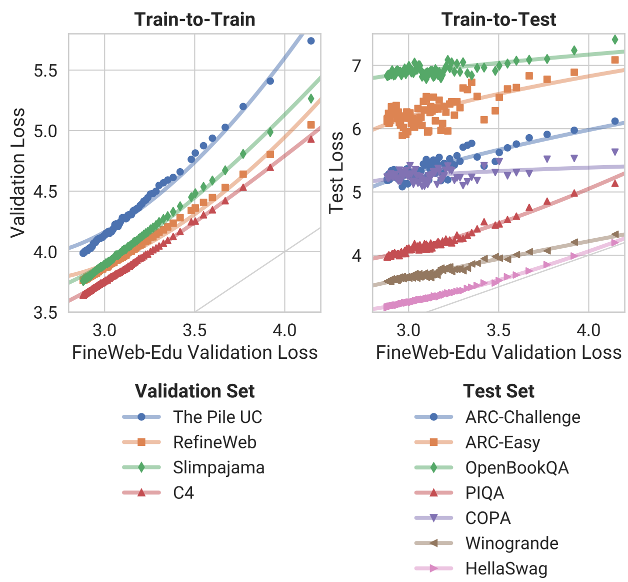

Figure 13: **Loss-to-Loss Scaling for FineWeb-Edu-trained Mamba**.

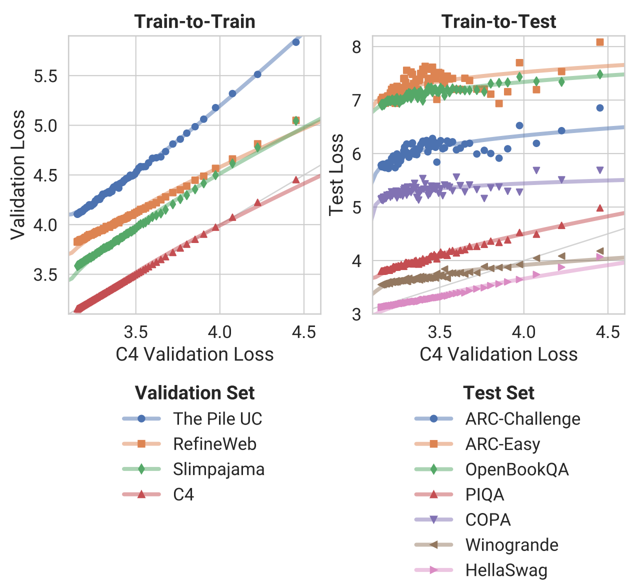

Figure 14: **Loss-to-Loss Scaling for The Pile-trained Mamba**.

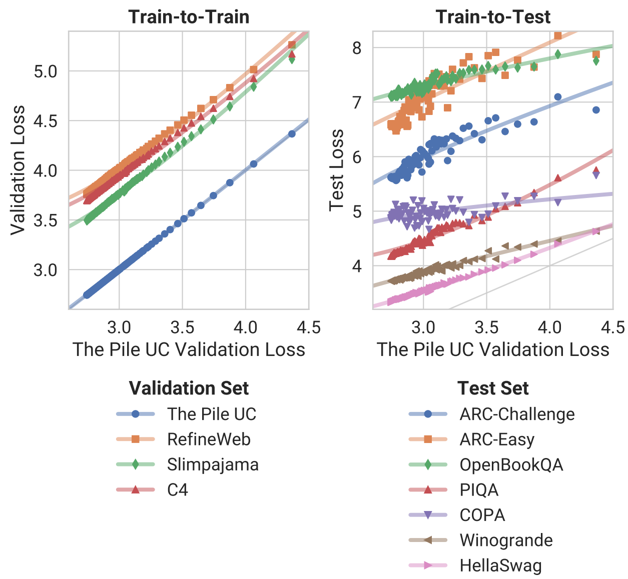

Figure 15: **Loss-to-Loss Scaling for The Pile-trained Mamba**.

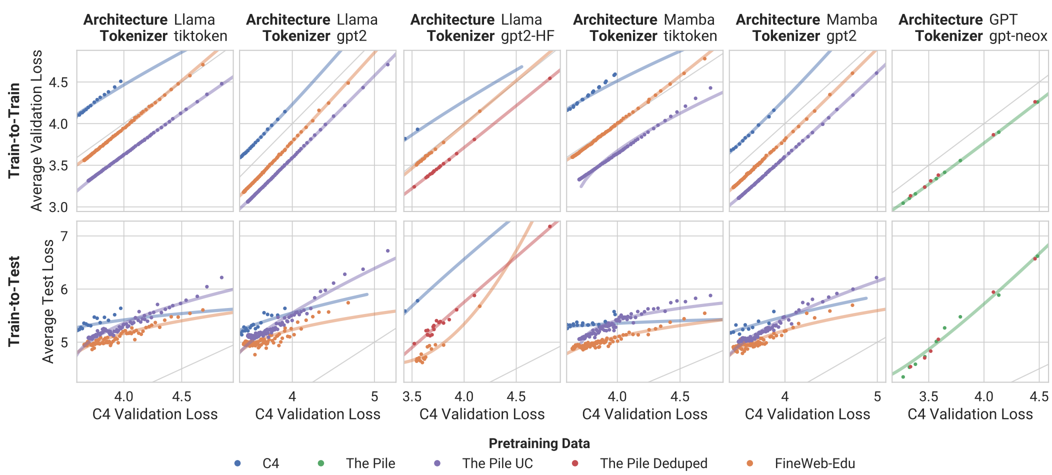

Figure 16: **Pretraining data has a substantial impact on loss-to-loss scaling laws**.

D INTERVENTION RESULTS FOR DIFFERENT CHOICES OF X-AXIS

We show variations of Figs. 4 to 6 from §4.1 with C4 validation loss as the x-axis in Figs. 16 to 18. Variations for Figs. 7 to 9 from §4.2 are shown in Figs. 19 to 21.

E INTERVENTION RESULTS WITHOUT AVERAGING

The causal analysis in §4 was performed on scaling laws for *average* validation or test loss. Figs. 22 to 27 show illustrative results on scaling laws for individual datasets.

F ADDITIONAL TRAIN-TO-TEST SCALING LAWS

We provide train-to-test scaling laws for the interventions performed in §4.2 and Figs. 7 to 9 in Figs. 28 to 33.

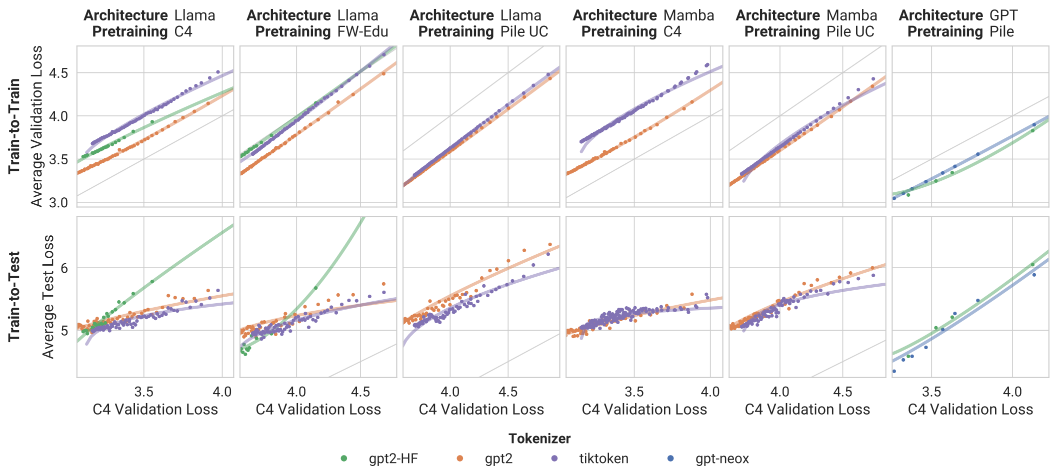

Figure 17: **The tokenizer has a moderate impact on loss-to-loss scaling laws**.

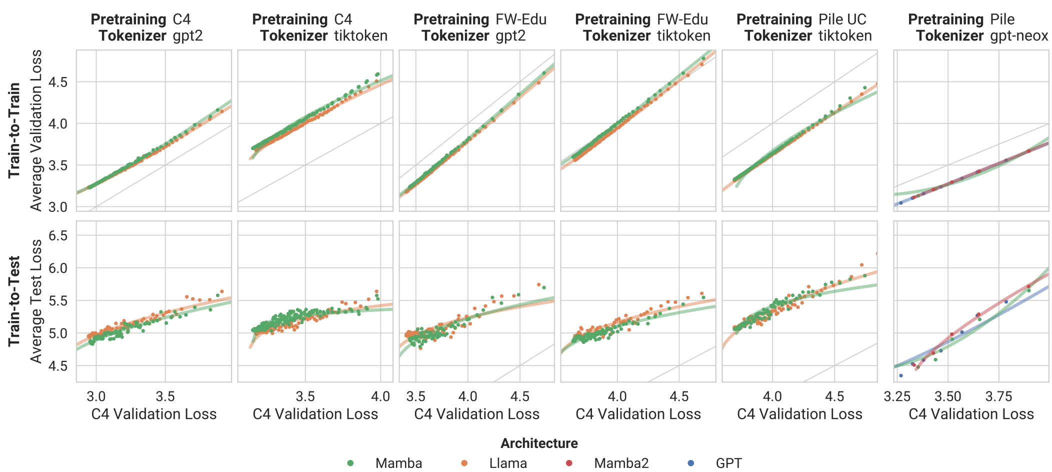

Figure 18: **Architecture has limited impact on loss-to-loss scaling laws**.

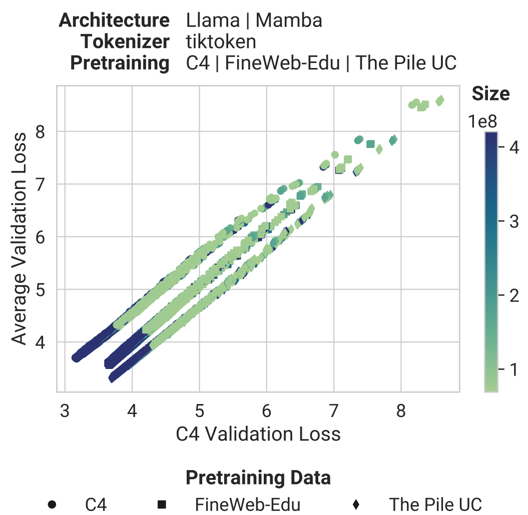

Figure 19: **Model size does not affect train-to-test scaling**.

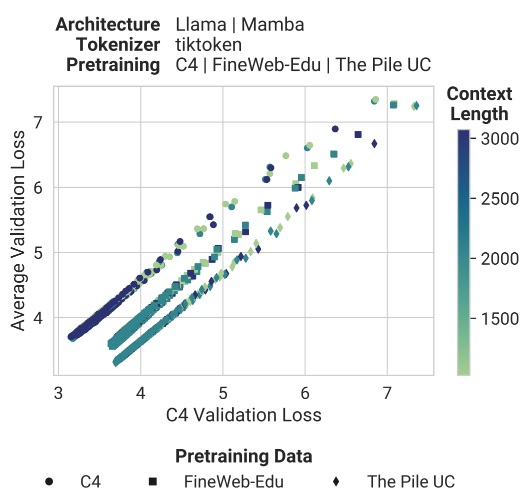

Figure 20: **Context length does not affect train-to-test scaling**.

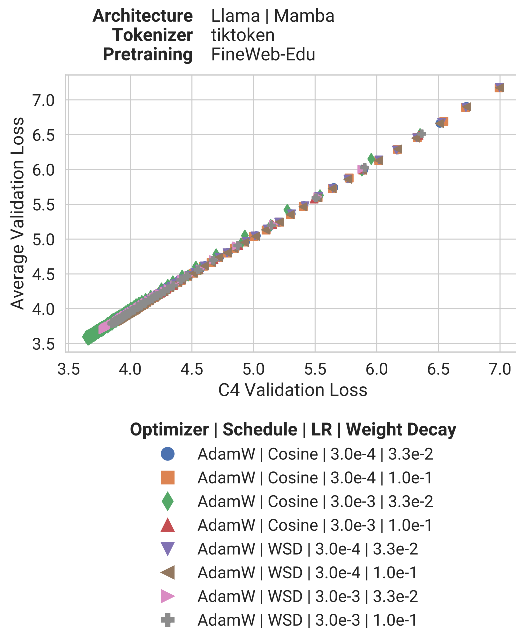

Figure 21: **Optimizer settings do not affect train-to-test scaling**.

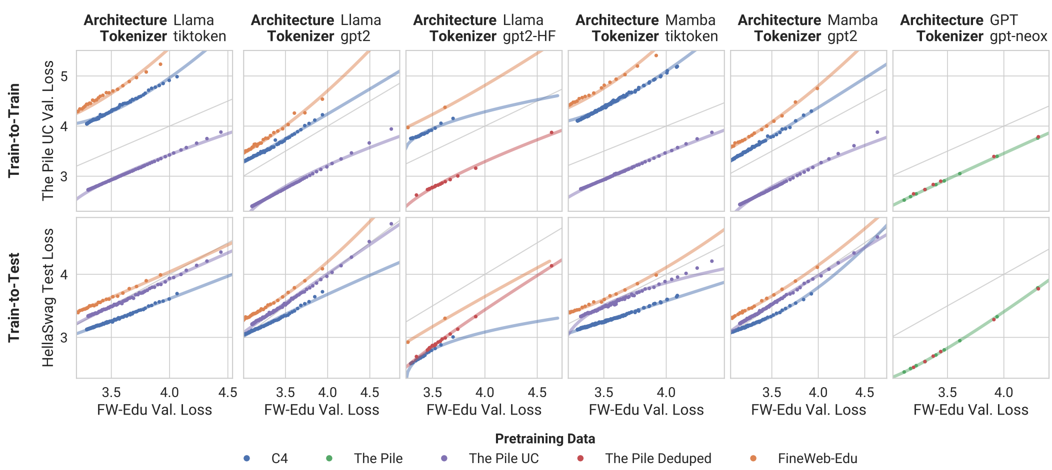

Figure 22: **Pretraining data has a substantial impact on loss-to-loss scaling laws**.

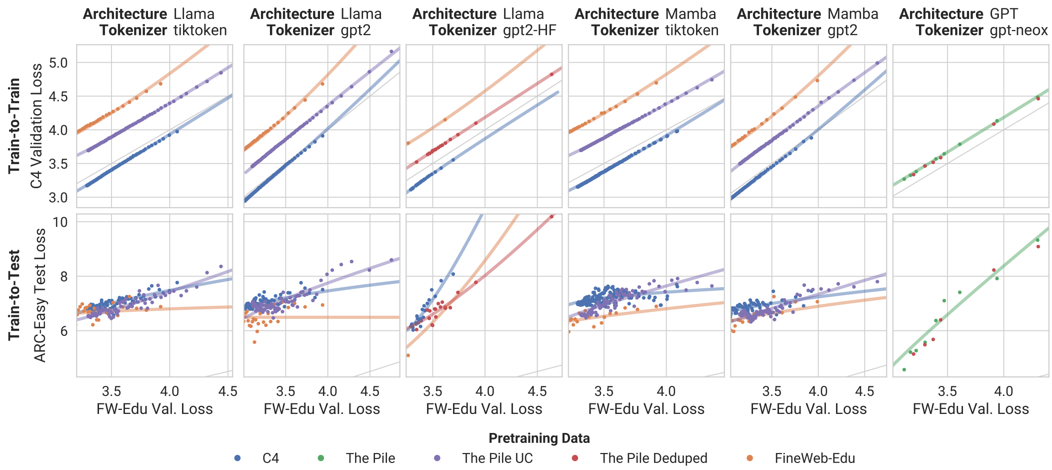

Figure 23: **Pretraining data has a substantial impact on loss-to-loss scaling laws.**

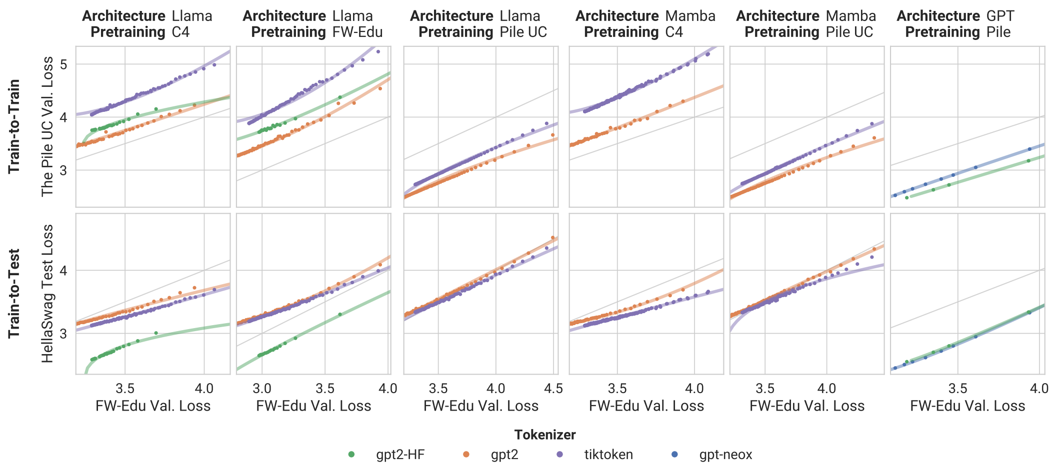

Figure 24: **The tokenizer has a moderate impact on loss-to-loss scaling laws.**

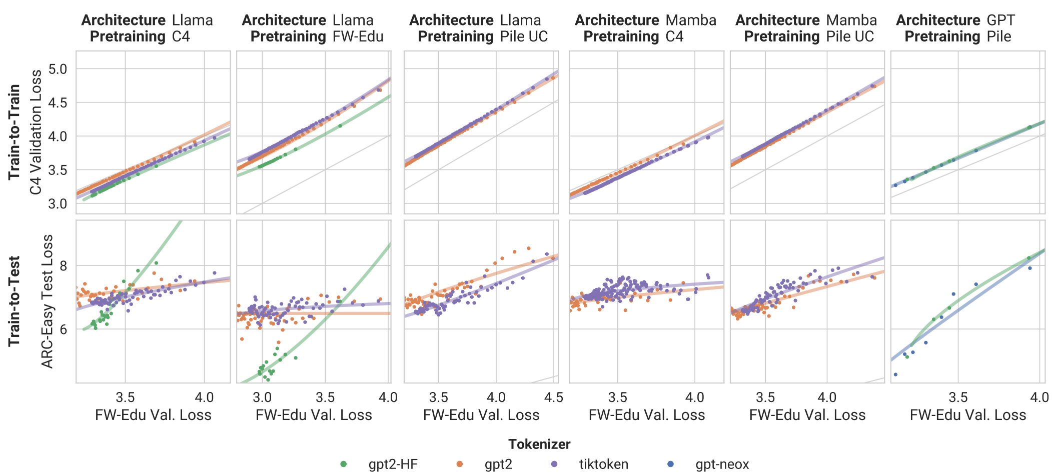

Figure 25: **The tokenizer has a moderate impact on loss-to-loss scaling laws.**

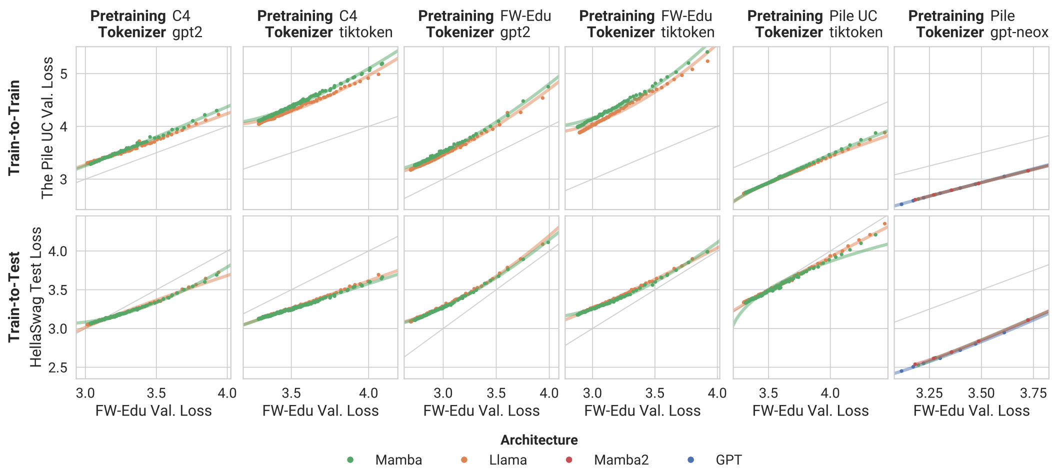

Figure 26: **Architecture has limited impact on loss-to-loss scaling laws**.

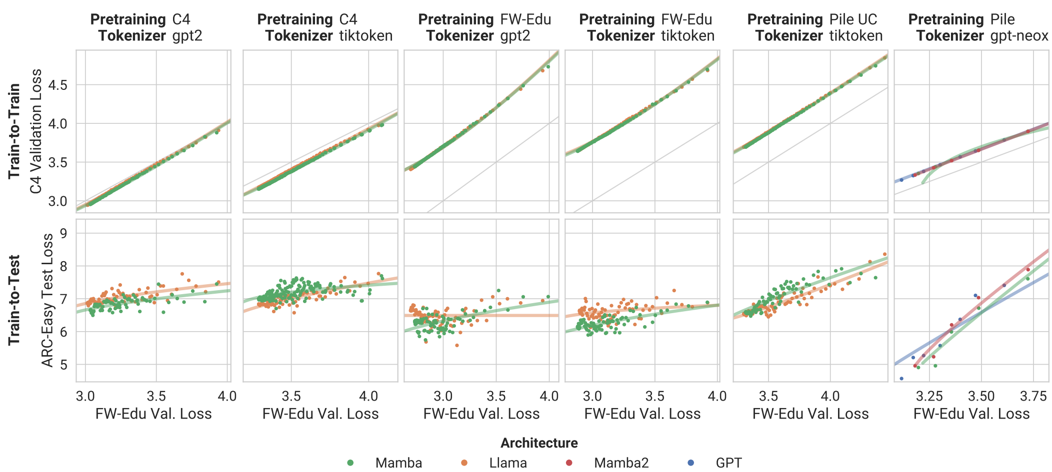

Figure 27: **Architecture has limited impact on loss-to-loss scaling laws**.

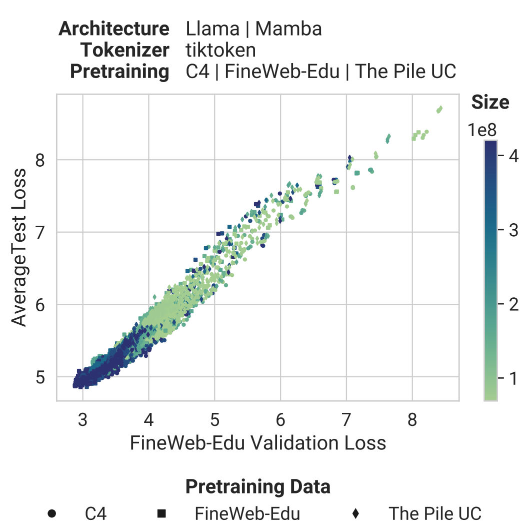

Figure 28: **Model size does not affect train-to-test scaling**.

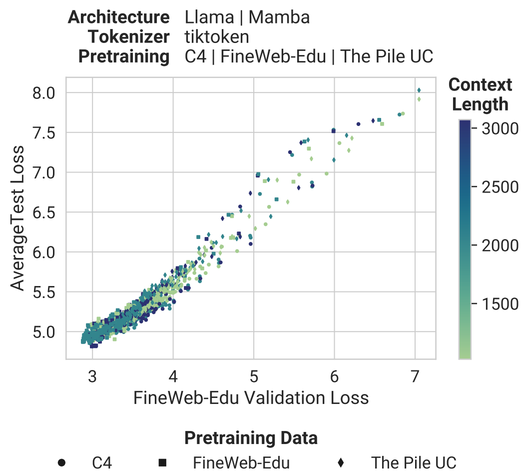

Figure 29: **Context length does not affect train-to-test scaling**.

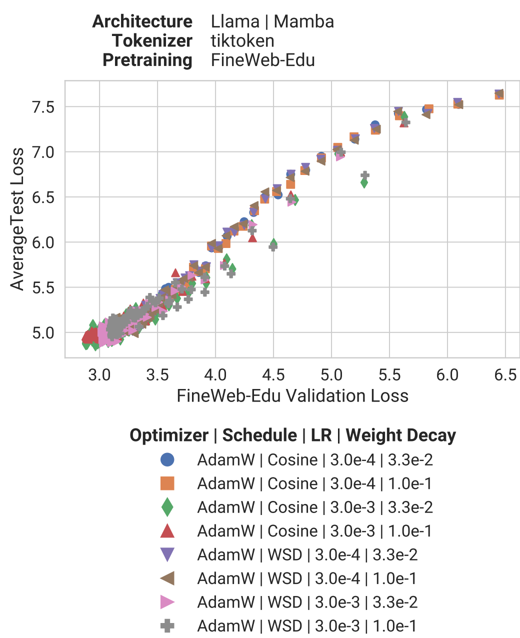

Figure 30: **Optimizer settings do not affect train-to-test scaling**.

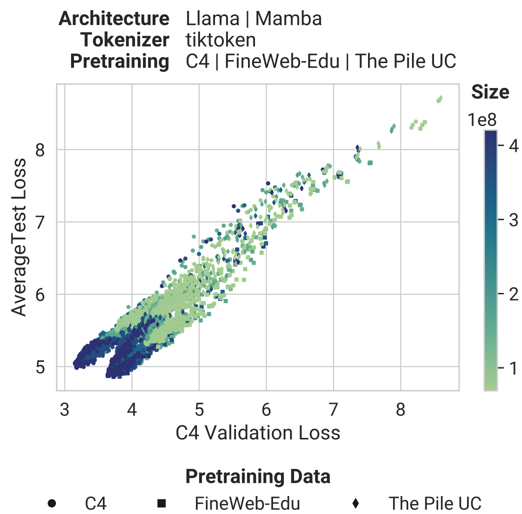

Figure 31: **Model size does not affect train-to-test scaling**.

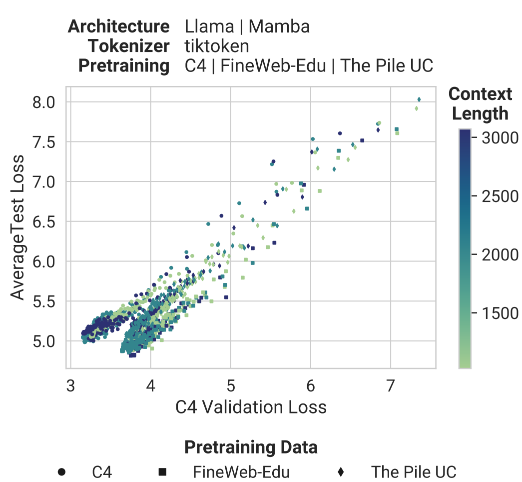

Figure 32: **Context length does not affect train-to-test scaling**.

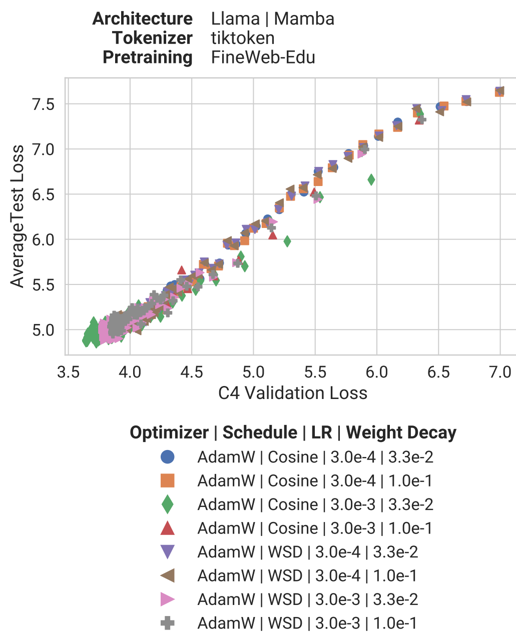

Figure 33: **Optimizer settings do not affect train-to-test scaling**.