Not-Just-Scaling Laws: Towards a Better Understanding of the Downstream Impact of Language Model Design Decisions

Emmy Liu[1], Amanda Bertsch[1], Lintang Sutawika[1], Lindia Tjuatja[1], Patrick Fernandes[1,2,3]
Lara Marinov[1], Michael Chen[1], Shreya Singhal[1], Carolin Lawrence[4],
Aditi Raghunathan[1], Kiril Gashteovski[4, 5], Graham Neubig[1]
[1] Carnegie Mellon University, Language Technologies Institute
[2]Instituto Superior Técnico (Lisbon ELLIS Unit), [3] Instituto de Telecomunicações
[4] NEC Laboratories Europe, Germany
[5] CAIR, Ss. Cyril and Methodius University of Skopje, North Macedonia
Correspondence: emmy@cmu.edu

Abstract

Improvements in language model capabilities are often attributed to increasing model size or training data, but in some cases smaller models trained on curated data or with different architectural decisions can outperform larger ones trained on more tokens. What accounts for this? To quantify the impact of these design choices, we meta-analyze 92 open-source pretrained models across a wide array of scales, including state-of-the-art open-weights models as well as less performant models and those with less conventional design decisions. We find that by incorporating features besides model size and number of training tokens, we can achieve a relative 3-28% increase in ability to predict downstream performance compared with using scale alone. Analysis of model design decisions reveal insights into data composition, such as the trade-off between language and code tasks at 15-25% code, as well as the negative association of web data with truthfulness. Broadly, our framework lays a foundation for more systematic investigation of how model development choices shape final capabilities. [1]

1 Introduction

The effectiveness of language model training depends critically on decisions made during pretraining. For instance, the effectiveness of scaling up data depends on its composition – processing even a trillion tokens would be ineffective if they all consisted of the word "the". Language model performance has been found to be fairly predictable through *scaling laws* (Kaplan et al. (2020), section 2) – extrapolations of model performance based on the parameter counts and number of tokens the models were trained on. However, scaling laws based on only these two aspects do not always explain downstream task performance (Diaz and Madaio, 2024; Isik et al., 2024). The research community has made progress in understanding how training decisions impact downstream performance with respect to data composition. For instance, controlled studies have demonstrated that training on code data improves performance on certain reasoning benchmarks (Aryabumi et al., 2024; Petty et al., 2024), meta-features of data such as age and the use of toxicity filters affect performance on many QA tasks (Longpre et al., 2024), and the balance of multilingual data affects performance on English and other languages (Chang et al., 2023; Yue et al., 2025). These works uncover valuable insights, but they tend to focus on changing only a single aspect of the training recipe while keeping the rest fixed. Although rigorous, this is costly in compute and development time. We instead ask: can we leverage past findings from open language models to examine which training decisions are associated with better downstream performance?

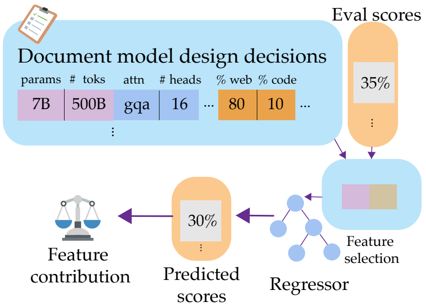


Figure 1: We document design decisions from open-weights models related to both architecture and data composition, and train predictors for downstream task performance. This allows us to examine the impact of model design choices individually.

To do so, we first *catalog* features regarding the model architecture, and data of 92 base pretrained LMs from varied families (§3). The resulting database of model features spans most major original pretrained decoder-only models released open-weights between the years 2019-2024.

We then develop methodology to *predict per-*

[1]Code and data are available at `https://anonymous.4open.science/r/llm-pretraining-behaviours-FE80/`

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 16408–16439
November 4-9, 2025 ©2025 Association for Computational Linguistics

formance of these models across a wide array of benchmarks both based on traditional scaling factors as well as architectural decisions and data composition (§4). Specifically, we train regression models that take in the extracted features and predict the benchmark results, and further use model interpretability techniques to identify the most salient features in making these predictions.

We evaluate this methodology on predicting performance across 12 popular LLM benchmarks, and demonstrate that it is *not just scaling* that determines model performance – on all benchmarks the regressor with all features outperforms a regressor based solely on scaling model features (§5.1). Our analysis of feature importance reveals potential impacts of data domains on task performance, reconfirming empirical results such as the best ratio of code to use in pretraining (§5.2). Furthermore, we find that features extracted from a model's generated text – such as the frequency of question-related words or the proportion of web-like text—help predict performance on various benchmarks. This suggests that a model's generation patterns can reflect underlying biases from its pretraining data that, in turn, influence downstream performance.

By documenting open-source models trained by the entire community and extracting insights, we provide a practical resource for model developers to learn from collective experience. We discuss this and future work in (§9).

2 Scaling Laws

2.1 Definition

We define scaling laws here as a relationship between the number of parameters N and the number of tokens D of a language model family, and the expected language-modelling loss at convergence $L(N, D)$.[2] Importantly, these laws are typically examined while holding all other factors constant: keeping the same model architecture, training data, and model parameters. Originally, Kaplan et al. (2020) showed that over a wide range of transformer-based models, this relationship can be expressed as a power law:

$$L(N, D) = \left(\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right)^{\alpha_D} \tag{1}$$

Later, Hoffmann et al. (2022a) introduced a similar law, which differed in the coefficients fitted, but was also based on a power law.

However, scaling laws are not absolute, and the exact functional form and fitted coefficients may depend on the architecture type, size range (Pearce and Song, 2024), or other considerations such as inference costs. See (§8.2) for further discussion.

2.2 Maybe it's Not Just Scaling?

Are parameter count and number of training tokens really all that are needed to accurately predict a model's downstream performance? Intuitively the answer is "no" – there are a number of design decisions that go into model training, and all of them could have an effect on model performance.

Model Architecture Details While the majority of modern language models follow the transformer architecture, there are some details that differ. For instance, the variety (Zhang and Sennrich, 2019) and position (Xiong et al., 2020) of layer normalization, and the type of positional encoding (Su et al., 2021; Press et al., 2022) make significant differences in model performance. Previous work, such as Gu and Dao (2023), has demonstrated empirically that holding all other factors equal, models that make better architecture decisions (Touvron et al., 2023a) outperform those that make worse decisions (Vaswani et al., 2017).

Data Composition In addition, data composition and quality plays a major role in the final quality of a model. For instance, past work has demonstrated that training on some quantity of code improves performance on English reasoning tasks (Ma et al., 2023). Also, work has demonstrated that filtering for "educational" content allows for more efficient learning and higher performance on knowledge-based question answering tasks (Gunasekar et al., 2023).

Task Setting Finally, there is an interplay of all the above factors with how model performance is measured. While previous work on scaling laws has mostly measured loss values, downstream users usually care about task performance, rather than validation loss on a pretraining dataset. Although there is often a correlation between the two for many tasks, certain tasks may be harder to predict from a model's loss alone (Bhagia et al., 2024). Moreover, certain tasks exhibit pathological scaling behaviour, such as inverse or U-shaped scaling

[2] Please see §8.2 for more detailed discussion; scaling laws can and do take into account other factors in various works, but for simplicity we call N and D scale-related here, while all other decisions in §3.2 are contrasted with these.

(Caballero et al., 2023; Wei et al., 2023; McKenzie et al., 2024), or simply more unpredictable performance (Isik et al., 2024).

We ask: *can we more effectively predict the performance of LLMs by devising a new set of "laws" that are not just reliant on scaling-based factors?*

3 Building a Database of Publicly-Available Language Models

To address our research question, we built a database of publicly available language models spanning 11M to 110B parameters,[3] limited to distinct pretrained decoder-only base models.[4] This section describes our inclusion criteria, model featurization, and evaluation approach.

3.1 Data Collection

To ensure that our analysis was consistent, we applied the following criteria:

Pretrained-only: Only base models that were pretrained from scratch were included. Fine-tuned variants, merged models, and models with additional post-training were excluded.

Architecture: Only transformer-based decoder-only models were included to maintain uniformity. Mixture-of-experts (MoEs) or other architectures were excluded.

Publicly available information: Only models with publicly available metadata, documented through configuration files or papers, were included. In particular, both the total number of parameters and total number of tokens trained on were required for inclusion. A full list of models and model families can be found in Appendix A.

3.2 Characterizing Models and Data

We represent each model by the architectural choices it makes, as well as its choice of pretraining data. Formally, let $\mathcal{A}$ be the set of features related to model architecture, and $\mathcal{D}$ be the set of features related to the model's pretraining dataset. For each task T we want to approximate a model M's true score s_T with a prediction $\widehat{s_T}$:

$$\widehat{s_T}(M) = f_\theta([\mathcal{A}_M; \mathcal{D}_M]). \quad (2)$$

This reduces to typical scaling laws when $\mathcal{A} = \{\#\,\text{params}\}$, $\mathcal{D} = \{\#\,\text{tokens}\}$, and f_θ is a power law.

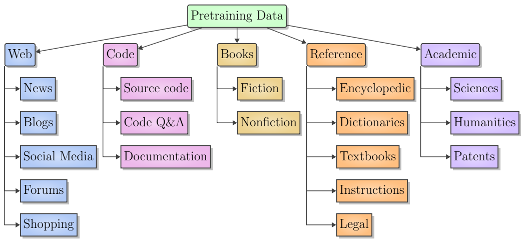


Figure 2: Taxonomy of pretraining data categories. We sorted data sources into this taxonomy based on model documentation.

In total, we document 92 open models along the dimensions of model features, high-level dataset features, and features derived from that model's no-context generations. For the full set of features and definitions, please see Appendix B.

3.2.1 Features from Model Documentation

We first collect information about each model by reading source papers/blogs when available (see Appendix A for original citations), as well as data listed on the Hugging Face Hub (Wolf et al., 2020).

Architectural Features: These features capture design decisions that determine model structure. For example, *total parameters* (including embedding parameters), the number of transformer layers, the embedding and feed-forward dimensions, and details such as the type of layer normalization or attention variant used.

Data Features: These features summarize pretraining data composition. Representative examples include *total tokens trained on* and the percentage breakdown of tokens sourced from various domains defined in Figure 2, as well as the proportion of English-language tokens. Our pretraining data domains were derived from common subdomains in open pretraining datasets (Gao et al., 2020; Soldaini et al., 2024). We use the top level domains (web, code, books, reference, academic) as this tends to be the granularity at which data composition is described in papers.

3.2.2 Exploring Data Composition via Generation

Although many models document some data composition details, relatively few release their full pretraining corpus, resulting in missing values for many models in our study.

[3] Including embedding parameters.

[4] By distinct, we mean unique combinations of training data and architecture. Models trained on deduplicated datasets are counted separately, but not variants with different curricula/initializations.

Commonsense Reasoning / NLI		
ANLI (Nie et al., 2020)	~ 163k	Brier Score
HellaSwag (Zellers et al., 2019)	~ 70k	Accuracy
Winogrande (Sakaguchi et al., 2019)	~ 44k	Accuracy
XNLI (Conneau et al., 2018)	~ 2.5k	Brier Score
Math / Logic		
GSM8K (Cobbe et al., 2021)	8 000	Accuracy
LogiQA2 (Wang et al., 2020)	~ 8k	Brier Score
MathQA (Saxton et al., 2019)	~ 37k	Brier Score
General Knowledge		
ARC Challenge (Clark et al., 2018)	~ 2.6k	Accuracy
Lambada (Paperno et al., 2016)	~ 10k	Accuracy
MMLU (Hendrycks et al., 2020)	~ 2.85k	Accuracy
Other		
TruthfulQA (Lin et al., 2021)	817	Accuracy
HumanEval (Chen et al., 2021)	164	Accuracy

Table 1: Overview of LM evaluation datasets with approximate sample counts and evaluation metrics. Datasets ANLI, XNLI, LogiQA2, and MathQA use Brier Score, while the others use Accuracy.

We propose estimating a model's training data by generating from it with only a beginning-of-sequence token in context (or end-of-sequence if BOS is unavailable). Using temperature-based sampling with $T = 1$,[5] we sample 5-10k free-generations per model and categorize them using an LM-based classifier (see Appendix E) according to the domains in Figure 2. A human annotator independently validated the classifier on 300 documents from pretraining datasets (Appendix F), showing high agreement (Cohen's $\kappa = 0.746$).

We also extract lower-level linguistic features such as words per sentence, constituency tree depth, and dependency length. Our validation analysis (Appendix G) shows that domain-level features correlate well with actual pretraining data composition (e.g., web content correlation: $r = 0.916$, $p = 7.55 \times 10^{-12}$), while lower-level stylistic features show weaker correlations. However, holistic model-level correlations across all features are strong (typically $r > 0.8$), supporting our use of free-generations as proxies for pretraining composition, while not substituting free-generation features for pretraining features.

3.3 Evaluation Datasets and Metrics

To assess how design choices affect reasoning capabilities, we evaluated models on datasets from the Open LLM leaderboard (Myrzakhan et al., 2024) that capture diverse aspects of reasoning (Table 1).[6] We collect results for some models directly from the leaderboard, and for models not on the leaderboard we use the Eleuther LM eval harness (Gao et al., 2023) to conduct evaluations with exactly the same setting. In addition, if there were multiple versions of a task or sub-tasks, we evaluated all of them and averaged them to get the overall task score. For the full list of evaluation datasets and settings, see Appendix C.

For an evaluation dataset T where the i-th sample is y_i and model M, we define $s_T(M)$ with:

Accuracy We use unnormalized, exact-match accuracy $s_{T,\text{acc}} = \frac{1}{|T|}\sum_{i=1}^{|T|} \mathbb{1}\{y_i = \hat{y}_i\}$ for the majority of tasks. We use pass@1 for Humaneval, but group it with accuracy tasks for convenience.

Brier score For tasks where smaller models struggle to achieve non-zero accuracy, we follow Schaeffer et al. (2023) in using multiclass brier score as an alternate continuous metric for multiple-choice tasks (Brier, 1950).[7] For a task with K classes, let p_{ik} be the predicted probability for class k on sample i. Then $s_{T,BS} = \frac{1}{|T|}\sum_{i=1}^{|T|}\sum_{k=1}^{K}(p_{ik} - \mathbb{1}\{y_i = k\})^2$.

3.4 Heterogeneity in Task-specific Scaling

Before adding in other factors, we examine differences in scaling along N and D between our selected tasks. We fit a Kaplan et al. (2020) style law to each task. As seen in Figure 3, we see that different tasks may exhibit marked differences both in how well they follow scaling trends, as well as their individual scaling contours. For instance, TruthfulQA appears to exhibit U-shaped scaling, while Humaneval has more "outlier" models. A full list of R^2 values for tasks can be found in Appendix D.

4 Predictive Modeling

Next, given our database we fit a regressor to try to predict performance. In traditional scaling laws, regressors are fit based on power laws. However, we are now dealing with a larger number of features, some of which may not be captured well by simple parametric forms. Hence, we follow previous work on performance prediction (Xia et al., 2020; Ye et al., 2021) utilizing tree-based regressors based on XGBoost (Chen and Guestrin, 2016).[8]

[5] this should in principle recover $P_{raw}(x_t \mid x_{<t}) \approx P(x_t \mid x_{<t})$ in the limit of sampling infinitely from an LM that captures its distribution perfectly.

[6] We excluded Arithmetic and Minerva math tasks (Brown et al., 2020; Hendrycks et al., 2021) as we focused on base models, and few achieved non-zero scores.

[7] Note that lower is better for brier score. Multiclass brier score ranges between 0-2.

[8] We also performed preliminary experiments with LightGBM (Ke et al., 2017) but it yielded very similar results in both prediction accuracy and feature importance. The LightGBM version of the main results can be found in Appendix K.

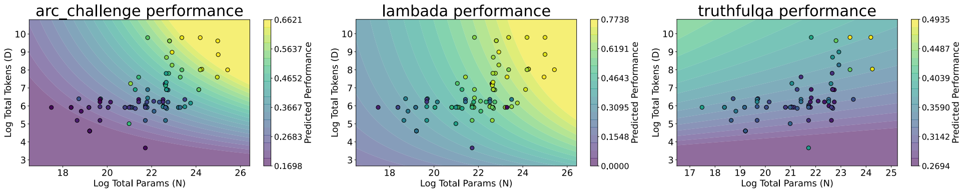


Figure 3: Performance of plotted against their total parameters and tokens. The background colour represents Equation 1 fitted to the task, and the marker colours indicate true performance. Some tasks have different performance trends with scale. Within each task, individual models may also perform unexpectedly.

For each evaluation benchmark, we train a model to predict the performance metric on that task based on architectural features $\mathcal{A}$ and data features $\mathcal{D}$. For each task setting, we perform 3-fold cross-validation due to the relatively small number of models, with a nested inner cross-validation over the training set in each fold. The inner cross-validation conducted grid search over a small set of hyperparameters, allowing the model to slightly vary per task. See Appendix I for more details.

Evaluation To evaluate the predictors, we use Mean Absolute Error averaged across all models and folds. In other words, for a task with N models evaluated, $\text{MAE}_T = \frac{1}{|T|}\sum_{i=1}^{N} |s_T(M_i) - \widehat{s_T}(M_i)|$. We compare the scaling-laws predictor as well as the all-features predictor against each other, but also against the **median baseline**, which simply predicts the median score of the models in the training set for each model in the test set of that fold, and the **log-linear baseline**, which fits a log-linear function to the number of parameters and number of tokens.

Iterative Feature Selection As the full set of features is very large, we sequentially selected features from the full set greedily based on which reduced MAE the most, averaged across 5 random seeds. Features were added until no reduction of at least 1×10^{-4} was observed. We started using only the two scaling laws features, and refer to this as the **scaling-laws** model, though it does not have the form of a traditional power law.[9] By then incorporating additional architectural or data features, we can then directly quantify the incremental predictive power afforded by these extra features. We refer to the model with the set of features as the **all-features** model. In all cases, we ran models with the same hyperparameter grid and the same random seeds and splits.

Significance Testing Because the relative difference between baselines is small, we test both predictors across many seeds (50). We then ran paired t-tests on the overall MAE values for each seed, and corrected for multiple comparisons across tasks with the False Discovery Rate (Benjamini and Hochberg, 1995).

5 Results

5.1 Predictor Performance

Incorporating scale-independent features consistently improves benchmark performance. We find that incorporating extra features alongside traditional scaling laws features leads to substantial improvements in prediction accuracy across multiple benchmarks, as seen in Table 2. The all-features predictor outperforms the scaling-laws-only predictor in all evaluated cases, with improvements ranging from approximately 3% (MathQA) to about 28% (Lambada) relative error reduction. Notably, the strongest improvements were observed in language modeling and common-sense reasoning tasks.

Certain tasks are more strongly dependent on non-scale features. This pattern of improvements suggests that architectural and training data features may be more informative for predicting performance on certain types of tasks more strongly linked to a particular "genre" of data. Large improvements were observed for both code generation (HumanEval, 15% improvement) as well as natural-language based reasoning tasks (e.g. Lambada, 28% improvement). Even tasks with narrower domains, such as mathematical reasoning (GSM8k, +16%) or knowledge-intensive evaluations (MMLU, +11–14%), see consistent, if more moderate, enhancements. The Brier score benchmarks, however, show smaller improvements (around 3–6%). This may be because the Brier score is inherently less sensitive to emergent effects

[9] As we use a tree-based predictor to accommodate diverse feature types (including non-numeric ones), our approach prioritizes interpolation within observed bounds (10M-100B parameters, 50B-3T tokens) rather than extrapolation. Exploring other prediction methods remains future work.

Benchmark	Setting	Baseline MAE	Log-Linear MAE	Scaling Laws MAE	All Features MAE	p-val (corrected)
			Accuracy			
Arc Challenge	25-shot	13.23%	4.91%	4.36% ± 0.12%	**3.67% ± 0.09%***	3.67×10^{-19}
GSM8k	5-shot	15.65%	11.03%	6.04% ± 0.21%	**5.10% ± 0.23%***	5.41×10^{-14}
Hellaswag	10-shot	12.26%	4.29%	3.93% ± 0.13%	**3.18% ± 0.09%***	4.44×10^{-20}
Humaneval	0-shot	11.79%	8.61%	8.08% ± 0.22%	**6.93% ± 0.22%***	4.44×10^{-20}
Lambada	0-shot	16.89%	9.60%	9.51% ± 0.33%	**6.85% ± 0.25%***	2.87×10^{-22}
MMLU (0-shot)	0-shot	11.98%	9.12%	4.76% ± 0.20%	**4.10% ± 0.17%***	5.26×10^{-13}
MMLU (5-shot)	5-shot	12.25%	8.39%	3.97% ± 0.18%	**3.54% ± 0.14%***	2.09×10^{-10}
TruthfulQA	0-shot	3.72%	3.40%	2.75% ± 0.08%	**2.29% ± 0.06%***	1.02×10^{-17}
Winogrande	5-shot	10.14%	3.99%	3.39% ± 0.08%	**3.09% ± 0.07%***	5.26×10^{-13}
			Brier score			
XNLI	0-shot	7.22	5.45	5.11 ± 0.11%	**4.30**± **0.11%***	1.37×10^{-2}
ANLI	0-shot	9.48	5.95	6.18 ± 0.19%	**5.86**± **0.21%***	3.16×10^{-9}
MathQA	0-shot	7.57	3.89	2.83 ± 0.06%	**2.75**± **0.07%***	1.63×10^{-4}
LogiQA2	0-shot	12.62	8.77	4.74 ± 0.12%	**4.60**± **0.15%***	3.84×10^{-4}

Table 2: Comparison of MAE values (mean ± 95% CI) for Scaling Laws and All Features predictors alongside Baseline MAE and Log-Linear MAE. Lower MAE is bolded; * indicates significance ($p < 0.05$). Brier score values are multiplied by 100 to be on a similar scale to accuracy.

in model performance, the specific choice of tasks limits the room for improvement, or a combination of both factors.

5.2 What Features Does Task Performance Depend On?

To understand factors influencing task performance, we examine Shapley (1953) (SHAP) values, which show how feature values affect predictions. Results for Arc Challenge, HumanEval, Winogrande, and TruthfulQA appear in Figure 4, with remaining benchmarks in Appendix L.

A little code goes a long way, but too much is harmful to NLI. The percentage of code in pretraining is a critical non-scaling feature. Higher code composition benefits Humaneval performance but is associated with lower scores on natural language reasoning tasks including Arc Challenge, Hellaswag, Winogrande, and Lambada. As shown in Figure 5, models with over 20-25% code show gains on Humaneval but penalties on language benchmarks. A moderate 15-25% code proportion appears to balance these competing demands.

Other data domains show task-specific effects. From free-generation features, we observed recent models trained on synthetic data (Phi (Gunasekar et al., 2023), SmolLM (Allal et al., 2024)) generate more question words, suggesting training on question-answering content. Reference-like or question-loaded generations correlate with better performance on Arc Challenge and Winogrande, while web-like generations correlate with worse TruthfulQA performance (Figure 4).

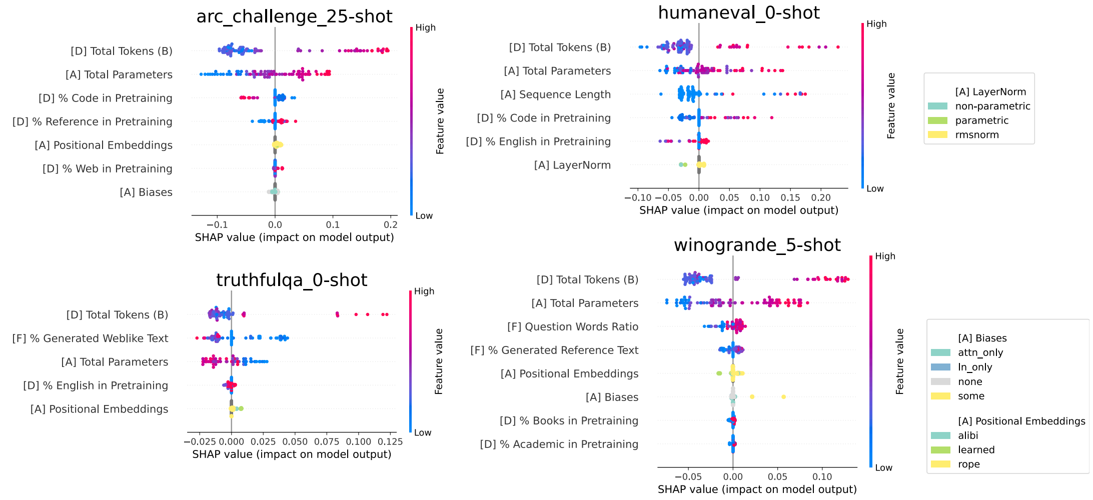
Non-scale architectural decisions have minor effects. Most highly influential features were data-related or architectural features related to scale (e.g., dimension). However, both the type of layer norm and the positional embedding were deemed to have a significant effect in some cases.

6 Validating Performance Predictions with Confirmatory Experiments

To validate findings from the meta-analysis, we also ran confirmatory pretraining runs with 460M-parameter models on the Dolma dataset. We aimed to validate two data distributional findings: (1) Around 8% code is optimal when only considering natural language inference, but 15-25% may be best when balancing code and natural language, and (2) TruthfulQA performance decreases with an increasing proportion of web data. As this is a small scale model and accuracy differences may not be significant, we convert the relevant datasets to use loss-based evaluations. Due to computational constraints, we train each checkpoint for 10B tokens, but use a cosine learning rate schedule scaled to a 100B token run. See Appendix M for details and exact loss figures. Overall, in Figure 6 find that the confirmatory runs largely validated our meta-analysis predictions, with the exception that our margin-based loss for truthfulQA was slightly lower for the 50% web data checkpoint compared to the 30% checkpoint, though the trend for accuracy is as expected. This provides preliminary evidence that our analysis method could be used to intelligently predict LM training design decisions a-priori.

7 Publication Bias Sensitivity Check

One other potential caveat to observational studies is that positive results are more likely to be reported:



Feature Name	Description
[D] Total Tokens (B)	Total number of tokens used during pretraining, measured in billions (log scale).
[A] Total Parameters	Total number of parameters in the model (log scale).
[D] % Code in Pretraining	Percentage of pretraining data that consists of code.
[F] Question Words Ratio	Ratio of question-related words generated by the model.
[A] Dimension	Embedding dimension.
[A] Sequence length	Sequence length.
[A] LayerNorm	Type of layer normalization used (non-parametric, parametric, rmsnorm).
[A] Biases	Presence of bias parameters in the model.
[A] Positional Embeddings	Type of positional embedding used (alibi, learned, rope).
[F] % English Generated	Percentage of English text generated by the model.
[D] % Academic in Pretraining	Percentage of pretraining data from academic sources.
[D] % Reference in Pretraining	Percentage of pretraining data from reference sources.
[F] % Generated Weblike Text	Percentage of web-like text generated by the model.
[F] % Generated Reference Text	Percentage of reference-like text generated by the model.
[D] % Books in Pretraining	Percentage of pretraining data from books.
[D] % English in Pretraining	Percentage of English text in the pretraining data.

Figure 4: In all tasks, the number of parameters and pretraining tokens heavily influences the predictions made by the regressor. The percentage of code in pretraining often influences predictions negatively for NLI tasks but positively for Humaneval. [D], [A] and [F] denote features derived from data, architecture, or free-generations of a model respectively.

especially in the case of architecture, people are likely to release a new model when the model does well. There are two cases of publication bias, using architecture as an example. There are two sources of publication bias, first in design space: some architectural variants that perform particularly poorly or well may never be trained or reported; these are inherently unobservable and we cannot comment on them. We focus on the second source of bias (small-scale bias), which is that among the results that we *do* observe, noisier task-level estimates are more likely to look large, and thus be emphasized. We focus on this type of bias correction, which is often examined in meta-analyses.

For each default benchmark/setting in our main suite, we computed a task-level contrast (one-vs-rest for a given architectural level, e.g. RoPE vs non-RoPE positional embeddings) via precision-weighted WLS with controls for log parameters and log pretraining tokens. We then meta-regressed these task-level effects y_i on their standard errors SE_i via a standard method known as PET-PEESE (Stanley and Doucouliagos 2014). Across the default *accuracy* tasks ($k \approx 6$ per feature), the pooled, bias-corrected effects are small: RoPE $\approx +0.015$, RMSNorm $\approx +0.023$, GQA $\approx +0.034$ (with wider uncertainty), while other levels are near zero. Results can be found in Appendix N. These adjustments leave our qualitative story unchanged: non-scale architectural choices have, on average, modest benchmark-robust effects compared to data composition and scale.

8 Related Work

8.1 Empirical Data Composition Results

Prior work has examined code in pretraining (Ma et al., 2023; Aryabumi et al., 2024) and domain ablations (Longpre et al., 2024). Data filtering improves performance beyond scaling alone (Sorscher et al., 2023; Goyal et al., 2024). Our results show code enhances natural language reasoning at moderate proportions (optimal ratio 15-25%), refining previous estimates of 25% (Aryabumi et al., 2024).

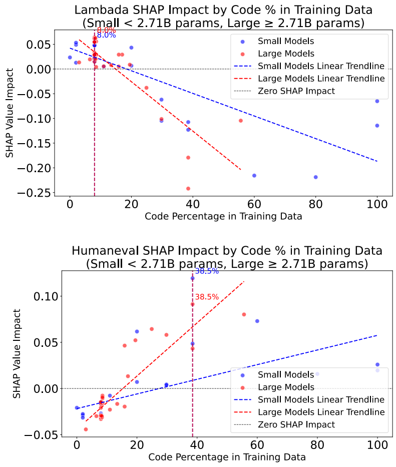


Figure 5: SHAP impact of code percentage on Lambada (reprentative NL task) and Humaneval on our regressors.

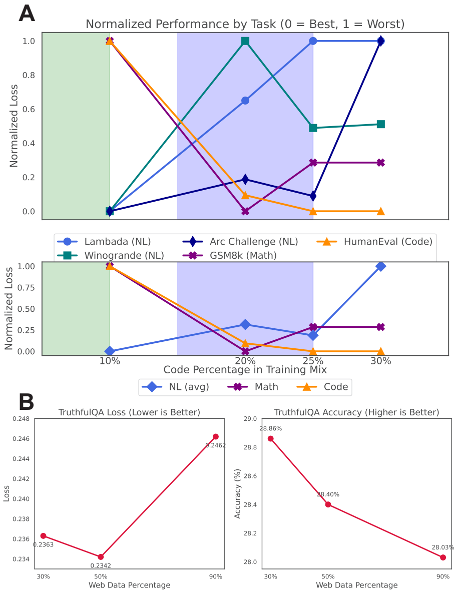


Figure 6: Loss on 460M parameter models, early checkpoints taken at 10% of final training length. Panel A depicts the effects of code vs. natural language data on natural language datasets and coding, while Panel B depicts the effects of web vs. non-web based data on truthfulQA. In Panel A, the optimal ranges for natural language tasks only (green) and for balancing natural language and code (blue) are highlighted.

Our approach of pooling insights from existing models complements empirical ablations by identifying promising axes for further testing.

8.2 Observational and Task-Specific Scaling Law Fitting

Task-specific scaling laws research shows parameter allocation affects machine translation outcomes (Ghorbani et al., 2021), and multitasking benefits English-target languages (Fernandes et al., 2023). Work on downstream tasks emphasizes alignment between pretraining and downstream data (Hernandez et al., 2021; Isik et al., 2024). Various studies address data repetition (Muennighoff et al., 2024), multiple domains (Goyal et al., 2024), and factors like sparsity (Frantar et al., 2023), precision (Kumar et al., 2024), and inference costs (Hoffmann et al., 2022b), while some find stability across training hyperparameters (DeepSeek-AI et al., 2024a).

Ruan et al. (2024) also use observations from open-source models to predict task performance, but derive their predictions of one task's performance from performance on other tasks. We find a similar result in identifying two axes of performance– general natural language ability and coding ability but are motivated instead by tracing these capabilities back to pretraining decisions.

8.3 Pretraining Data Selection

Domain mixing has been studied in pretraining, and other works have formulated this as a regression problem (Ye et al., 2024; Liu et al., 2025) or used proxy models to select domain weights in the course of training (Xie et al., 2023; Albalak et al., 2023; Jiang et al., 2024b; Yu et al., 2025). In contrast, we retrospectively analyze how domain composition and training decisions influence performance across tasks, which is a complementary perspective to optimizing data weights for a single model during training.

8.4 Tracing Capabilities to Data

Specific language model capabilities have been linked to patterns in pretraining data. Performance on numerical reasoning and syntactic rule learning depends on frequency of numerical terms in the training data (Kassner et al., 2020; Wei et al., 2021). Ruis et al. (2024) found that influential data for reasoning is dispersed across numerous documents and is associated with procedural content. Similarly, Chen et al. (2024) observed that "parallel structures" are closely tied to in-context learning abilities. We currently focus on broader data domains, but our framework can be extended with more granular tasks or refined data features.

9 Conclusion and Future Work

We perform the first systematic analysis of the performance of open language models across diverse tasks and tie their performance to architectural and data-compositional design decisions. Looking into the future, there are a number of clear directions. First, our database (§3) can be further expanded as new models and benchmarks are released, and we will release the code and data to help spur community efforts for more systematic data documentation. Second, we hope our work will help discover hypotheses to be tested in more controlled settings – existing models intertwine a number of design decisions, and further controlled pre-training experiments that only involve one axis of variation could further clarify the effect of each feature. Finally, within our study, the great majority of pretrained models focused on dense transformer architectures, while alternative architectures such as mixture-of-experts (Jiang et al., 2024a; DeepSeek-AI et al., 2024b) and state-space models (Gu and Dao, 2023) have also seen significant research interest. How to appropriately featurize these more various model architectures and use the information in performance prediction is an interesting challenge that may uncover further insights. Lastly, although pretraining data analysis and selection has mainly been focused on empirical findings so far, building a better understanding of how training affects model capabilities through large-scale empirical studies could also facilitate interpretability experiments and possible interventions on learned representations, with controlled axes of variation providing case studies.

Limitations

Our current work has several limitations that can be improved in future work. First, although we document many open models, our sample size remains limited, particularly for larger (>50B) parameter models. This limits our ability to draw robust conclusions about scaling behaviour in large models. Additionally, the models that we have are not evenly distributed across number of parameters, data size, and data distributions, with certain size ranges and data distributions being overrepresented. There are also likely selection effects in which models are made open-weights, as well as likely time effects in popular architectural decisions or data compositions in different time periods.

Second, our methodology also imposes some limitations. Because we do not systematically train all our own models (though we have a few of our own in Appendix A), our analyses are observational in nature. While we can observe interesting relationships between design choices and performance, making causal claims requires experimental validation. Additionally, while tree-based regressors are effective for capturing complex feature interactions, they limit our ability to extrapolate beyond the range of model sizes (in parameters and tokens) seen in our dataset.

Last, we note that the scope of our work also has limitations. Namely, we focus on base pretrained decoder-only dense transformer models, which excludes significant architectural variants such as mixture-of-experts models, non-transformer based architectures, as well as post-trained models. Additionally, we examine mostly English-language models as we do not focus on multilinguality in this work. Our feature set, while extensive, may also not capture all relevant details of model design and training, particularly optimization details as of now.

These limitations suggest directions for future work: expanding the database to include more diverse model types and language coverage, developing more targeted functional forms that allow better extrapolation while also taking as input a heterogeneous feature set, as well as conducting targeted experiments with new pretrained models to validate the impact of specific design choices.

Ethical Considerations

In this work, we focus on understanding why models may perform well on standard benchmarks, but do not focus on other important considerations such as safety or societal bias.

Furthermore, our analysis focuses on English-language models and benchmarks. This limitation reflects but may also reinforce the field's existing bias toward English, potentially contributing to underinvestment in developing effective architectures for other languages.

Author Contribution Statement

We follow a slightly modified version of the CRediT author statement.

- **EL**: conceptualization (lead), data curation (equal), methodology/software (lead), writing – original draft (lead), writing – review and editing (lead), visualization (lead), funding acquisition (supporting)
- **AB**: conceptualization (supporting), methodology/software (supporting), data curation (equal), writing – review and editing (supporting)
- **LS**: conceptualization (supporting), methodology/software (supporting), data curation (equal)
- **LT**: conceptualization (supporting), methodology/software (supporting), data curation (equal)
- **PF**: conceptualization (supporting), methodology/software (supporting), data curation (equal), writing – review and editing (supporting)
- **LM**: methodology/software (supporting), data curation (equal)
- **MC**: methodology/software (supporting), data curation (equal)
- **SS**: methodology/software (supporting), data curation (equal)
- **CL**: writing – review and editing (supporting), project administration (supporting)
- **AR**: conceptualization (supporting), writing – review and editing (supporting), supervision (supporting)
- **KG**: conceptualization (supporting), writing – review and editing (supporting), supervision (supporting), project administration (supporting)
- **GN**: conceptualization (lead), methodology/software (supporting), writing – review and editing (supporting), supervision (lead), project administration (lead), funding acquisition (lead)

Acknowledgments

This work was supported by a fellowship from NEC Laboratories Europe and the BRIDGE CMU-AIST research project. EL was supported by the Natural Science and Engineering Research Council of Canada (NSERC), [funding reference number 578085]. AB was supported by a grant from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE2140739. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors.

Use of AI Assistants

Claude 3.5 Sonnet and GPT-o3-mini-high were used to help revise and shorten several parts of this submission as well as to edit for clarity. The first draft was entirely human-written.

References

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yanpeng Li, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2025. Yi: Open foundation models by 01.ai. *Preprint*, arXiv:2403.04652.

Marah Abdin, Jyoti Aneja, Sébastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar, Mojan Javaheripi, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacroce, Harkirat Singh Behl, Adam Tauman Kalai, Xin Wang, Rachel Ward, Philipp Witte, Cyril Zhang, and Yi Zhang. 2023. Phi-2: The surprising power of small language models. Microsoft Research Blog.

Meta AI. 2023. Llama 3: Advancing foundation models. https://ai.facebook.com/blog/llama-3. Accessed: 2025-02-21.

Alon Albalak, Liangming Pan, Colin Raffel, and William Yang Wang. 2023. Efficient online data mixing for language model pre-training. *Preprint*, arXiv:2312.02406.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Leandro von Werra, and Thomas Wolf. 2024. Smollm - blazingly fast and remarkably powerful.

Viraat Aryabumi, Yixuan Su, Raymond Ma, Adrien Morisot, Ivan Zhang, Acyr Locatelli, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. 2024. To code, or not to code? exploring impact of code in pre-training. *Preprint*, arXiv:2408.10914.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *Preprint*, arXiv:2309.16609.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300.

Akshita Bhagia, Jiacheng Liu, Alexander Wettig, David Heineman, Oyvind Tafjord, Ananya Harsh Jha, Luca Soldaini, Noah A. Smith, Dirk Groeneveld, Pang Wei Koh, Jesse Dodge, and Hannaneh Hajishirzi. 2024. Establishing task scaling laws via compute-efficient model ladders. *Preprint*, arXiv:2412.04403.

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

BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Daniel McDuff, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. 2023. Bloom: A 176b-parameter open-access multilingual language model. *Preprint*, arXiv:2211.05100.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Glenn W. Brier. 1950. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Ethan Caballero, Kshitij Gupta, Irina Rish, and David Krueger. 2023. Broken neural scaling laws. *Preprint*, arXiv:2210.14891.

Cerebras Systems. 2023. Cerebras-gpt: A family of open, compute-efficient large language models. *Preprint*, arXiv:2304.03208.

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K. Bergen. 2023. When is multilinguality a curse? language modeling for 250 high- and low-resource languages. *Preprint*, arXiv:2311.09205.

Mark Chen, Jacob Tworek, et al. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794. ACM.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. 2024. Parallel structures in pre-training data yield in-context learning. *Preprint*, arXiv:2402.12530.

Peter Clark et al. 2018. Arc: A large-scale challenge dataset for ai2 reasoning. In *Proceedings of EMNLP*.

Karl Cobbe, Vineet Kosaraju, et al. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2106.03350.

Alexis Conneau et al. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of EMNLP*.

DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wenjie Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, A. X. Liu, Bo Liu, Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Junxiao Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Minghui Tang, Bingxuan Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Y. Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yiliang Xiong, Hanwei Xu, R. X. Xu, Yanhong Xu, Dejian Yang, Yuxiang You, Shuiping Yu, Xingkai Yu, B. Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghua Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. 2024a. Deepseek llm: Scaling open-source language models with longtermism. *Preprint*, arXiv:2401.02954.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Nolan Dey, Daria Soboleva, Faisal Al-Khateeb, Bowen Yang, Ribhu Pathria, Hemant Khachane, Shaheer Muhammad, Zhiming, Chen, Robert Myers, Jacob Robert Steeves, Natalia Vassilieva, Marvin Tom, and Joel Hestness. 2023. Btlm-3b-8k: 7b parameter performance in a 3b parameter model. *Preprint*, arXiv:2309.11568.

Fernando Diaz and Michael Madaio. 2024. Scaling laws do not scale. *Preprint*, arXiv:2307.03201.

Patrick Fernandes, Behrooz Ghorbani, Xavier Garcia, Markus Freitag, and Orhan Firat. 2023. Scaling laws for multilingual neural machine translation. In *Proceedings of the 40th International Conference on Machine Learning*, pages 10053–10071. PMLR.

Allen Institute for AI. 2024. Olmo: Open language model. `https://huggingface.co/allenai/OLMo-7B-hf`.

Elias Frantar, Carlos Riquelme, Neil Houlsby, Dan Alistarh, and Utku Evci. 2023. Scaling Laws for Sparsely-Connected Foundation Models. *International Conference on Learning Representations (ICLR)*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *Preprint*, arXiv: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. 2023. A framework for few-shot language model evaluation.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao

Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024a. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. 2024b. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

Behrooz Ghorbani, Orhan Firat, Markus Freitag, Ankur Bapna, Maxim Krikun, Xavier Garcia, Ciprian Chelba, and Colin Cherry. 2021. Scaling laws for neural machine translation. *Preprint*, arXiv:2109.07740.

Sachin Goyal, Pratyush Maini, Zachary C. Lipton, Aditi Raghunathan, and J. Zico Kolter. 2024. Scaling laws for data filtering – data curation cannot be compute agnostic. *Preprint*, arXiv:2404.07177.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Dan Hendrycks et al. 2020. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. 2021. Scaling Laws for Transfer. *arXiv preprint arXiv:2102.01293*.

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. 2022a. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Julian Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Tianyi Cai, Eric Rutherford, Daniel de la Casas, Lucy A Hendricks, Joshua

Welbl, Alec Clark, et al. 2022b. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Technology Innovation Institute. 2023. The falcon series of open language models. *Preprint*, arXiv:2306.01116.

Berivan Isik, Natalia Ponomareva, Hussein Hazimeh, Dimitris Paparas, Sergei Vassilvitskii, and Sanmi Koyejo. 2024. Scaling laws for downstream task performance of large language models. *arXiv preprint arXiv:2402.04177*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024a. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Yiding Jiang, Allan Zhou, Zhili Feng, Sadhika Malladi, and J. Zico Kolter. 2024b. Adaptive data optimization: Dynamic sample selection with scaling laws. *Preprint*, arXiv:2410.11820.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are Pretrained Language Models Symbolic Reasoners over Knowledge? *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tanishq Kumar, Zachary Ankner, Benjamin F. Spector, Blake Bordelon, Niklas Muennighoff, Mansheej Paul, Cengiz Pehlevan, Christopher Ré, and Aditi Raghunathan. 2024. Scaling laws for precision. *Preprint*, arXiv:2411.04330.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*.

Stephanie Lin et al. 2021. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of EMNLP*.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. 2025. Regmix: Data mixture as regression for language model pre-training. *Preprint*, arXiv:2407.01492.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, Richard Fan, Yi Gu, Victor Miller, Yonghao Zhuang, Guowei He, Haonan Li, Fajri Koto, Liping Tang, Nikhil Ranjan, Zhiqiang Shen, Xuguang Ren, Roberto Iriondo, Cun Mu, Zhiting Hu, Mark Schulze, Preslav Nakov, Tim Baldwin, and Eric P. Xing. 2023. Llm360: Towards fully transparent open-source llms. *Preprint*, arXiv:2312.06550.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. 2024. A Pretrainer's Guide to Training Data: Measuring the Effects of Data Age, Domain Coverage, Quality, & Toxicity. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Lalita Lowphansirikul, Charin Polpanumas, Nawat Jantrakulchai, and Sarana Nutanong. 2021. Wangchanberta: Pretraining transformer-based thai language models. *Preprint*, arXiv:2101.09635.

Yingwei Ma, Yue Liu, Yue Yu, Yuanliang Zhang, Yu Jiang, Changjian Wang, and Shanshan Li. 2023. At which training stage does code data help llms reasoning? *Preprint*, arXiv:2309.16298.

Ian R. McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Aaron Kirtland, Alexis Ross, Alisa Liu, Andrew Gritsevskiy, Daniel Wurgaft, Derik Kauffman, Gabriel Recchia, Jiacheng Liu, Joe Cavanagh, Max Weiss, Sicong Huang, The Floating Droid, Tom Tseng, Tomasz Korbak, Xudong Shen, Yuhui Zhang, Zhengping Zhou, Najoung Kim, Samuel R. Bowman, and Ethan Perez. 2024. Inverse scaling: When bigger isn't better. *Preprint*, arXiv:2306.09479.

Meta AI. 2022a. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

Meta AI. 2022b. Xglm: Cross-lingual generative language models. *Preprint*, arXiv:2204.07613.

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-05-05.

Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. 2024. Scaling data-constrained language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.

Aidar Myrzakhan, Sondos Mahmoud Bsharat, and Zhiqiang Shen. 2024. Open-llm-leaderboard: From multi-choice to open-style questions for llms evaluation, benchmark, and arena. *arXiv preprint arXiv:2406.07545*.

Yixin Nie, Jing Wang, Mohit Bansal, and Kai-Wei Chang. 2020. Adversarial natural language inference. In *Proceedings of ACL*.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2023. Codegen: An open large language model for code with multi-turn program synthesis. *Preprint*, arXiv:2203.13474.

Denis Paperno et al. 2016. Lambada: Word prediction requiring a broad discourse context. In *Proceedings of ACL (Long Papers)*.

Tim Pearce and Jinyeop Song. 2024. Reconciling kaplan and chinchilla scaling laws. *Preprint*, arXiv:2406.12907.

Jackson Petty, Sjoerd van Steenkiste, and Tal Linzen. 2024. How does code pretraining affect language model task performance? *Preprint*, arXiv:2409.04556.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Tianyi Qi, Yichao Ouyang, Yu Liang, Lifu Huang, Xiao Dang, Kevin Gimpel, and Noah A. Smith. 2020. Stanza: A python nlp toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. 2024. Observational scaling laws and the predictability of language model performance. *Preprint*, arXiv:2405.10938.

Laura Ruis, Maximilian Mozes, Juhan Bae, Siddhartha Rao Kamalakara, Dwarak Talupuru, Acyr Locatelli, Robert Kirk, Tim Rocktäschel, Edward Grefenstette, and Max Bartolo. 2024. Procedural knowledge in pretraining drives reasoning in large language models. *Preprint*, arXiv:2411.12580.

Rei Sakaguchi et al. 2019. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of ACL*.

Kei Sawada, Tianyu Zhao, Makoto Shing, Kentaro Mitsui, Akio Kaga, Yukiya Hono, Toshiaki Wakatsuki, and Koh Mitsuda. 2024. Release of pre-trained models for the Japanese language. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13898–13905. https://arxiv.org/abs/2404.01657.

William Saxton, Edward Grefenstette, et al. 2019. Mathqa: A challenge dataset for solving math word problems. In *Proceedings of EMNLP*.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? *Preprint*, arXiv:2304.15004.

Lloyd S Shapley. 1953. A value for n-person games. In *Contributions to the Theory of Games*, pages 307–317. Princeton University Press.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint*.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. 2023. Beyond neural scaling laws: beating power law scaling via data pruning. *Preprint*, arXiv:2206.14486.

T. D. Stanley and Hristos Doucouliagos. 2014. Meta-regression approximations to reduce publication selection bias. *Research Synthesis Methods*, 5(1):60–78.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. LLaMA: Open and Efficient Foundation Language Models. *ArXiv*, abs/2302.13971.

Hugo Touvron et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. https://ai.meta.com/llama2/. Accessed: 2025-02-21.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Yichong Wang, Yuxuan Feng, et al. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of ACL*.

Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. 2021. Frequency Effects on Syntactic Rule Learning in Transformers. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jason Wei, Najoung Kim, Yi Tay, and Quoc V. Le. 2023. Inverse scaling can become u-shaped. *Preprint*, arXiv:2211.02011.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi 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 Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8625–8646, Online. Association for Computational Linguistics.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. *Preprint*, arXiv:2305.10429.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10524–10533. PMLR.

Frank F. Xu, Uri Alon, Graham Neubig, and Vincent J. Hellendoorn. 2022. A systematic evaluation of large language models of code. *Preprint*, arXiv:2202.13169.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Yunhua Zhou, Jun Zhan, and Xipeng Qiu. 2024. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *Preprint*, arXiv:2403.16952.

Zihuiwen Ye, Pengfei Liu, Jinlan Fu, and Graham Neubig. 2021. Towards more fine-grained and reliable NLP performance prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3703–3714, Online. Association for Computational Linguistics.

Zichun Yu, Fei Peng, Jie Lei, Arnold Overwijk, Wen tau Yih, and Chenyan Xiong. 2025. Data-efficient pretraining with group-level data influence modeling. *Preprint*, arXiv:2502.14709.

Xiang Yue, Yueqi Song, Akari Asai, Seungone Kim, Jean de Dieu Nyandwi, Simran Khanuja, Anjali Kantharuban, Lintang Sutawika, Sathyanarayanan Ramamoorthy, and Graham Neubig. 2025. Pangea: A fully open multilingual multimodal llm for 39 languages. *Preprint*, arXiv:2410.16153.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of ACL*.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Preprint*, arXiv:1910.07467.

A List of all models

All models are listed in Table 3.

Table 3: Model Parameter Counts by Organization (sorted by size)

Organization	Model Name	Parameters
EleutherAI (Biderman et al., 2023)	pythia-14m	14M
EleutherAI	pythia-70m-deduped	70M
EleutherAI	pythia-70m	70M
facebook (Meta AI, 2022a)	opt-125m	125M
EleutherAI (Black et al., 2021)	gpt-neo-125m	125M
HuggingFaceTB (Allal et al., 2024)	SmolLM-135M	135M
EleutherAI	pythia-160m	160M
EleutherAI	pythia-160m-deduped	160M
None (this paper)	llama2_220M_nl_100_code_0	220M
None (this paper)	llama_220M_nl_80_code_20	220M
None (this paper)	llama2_220M_nl_40_code_60	220M
None (this paper)	llama2_220M_nl_20_code_80	220M
None (this paper)	llama2_220M_nl_0_code_100	220M
Salesforce (Nijkamp et al., 2023)	codegen-350M-mono	350M
Salesforce	codegen-350M-multi	350M
Salesforce	codegen-350M-nl	350M
facebook	opt-350m	350M
HuggingFaceTB	SmolLM-360M	360M
EleutherAI	pythia-410m-deduped	410M
EleutherAI	pythia-410m	410M
facebook (Meta AI, 2022b)	xglm-564M	564M
EleutherAI	pythia-1b-deduped	1B
bigscience (BigScience Workshop et al., 2023)	bloom-1b7	1B
EleutherAI	pythia-1b	1B
cerebras (Cerebras Systems, 2023)	Cerebras-GPT-1.3B	1.3B
microsoft (Li et al., 2023)	phi 1.5	1.3B
EleutherAI	gpt-neo-1.3B	1.3B
EleutherAI	pythia-1.4b	1.4B
EleutherAI	pythia-1.4b-deduped	1.4B
HuggingFaceTB	SmolLM-1.7B	1.7B
Salesforce	codegen-2B-mono	2B
Salesforce	codegen-2B-nl	2B
Salesforce	codegen-2B-multi	2B
google (Gemma Team et al., 2024b)	gemma-2-2b	2B
cerebras	Cerebras-GPT-2.7B	2.7B
EleutherAI	gpt-neo-2.7B	2.7B
NinedayWang (Xu et al., 2022)	PolyCoder-2.7B	2.7B
facebook	opt-2.7b	2.7B
microsoft (Abdin et al., 2023)	phi 2	2.7B
EleutherAI	pythia-2.8b	2.8B
EleutherAI	pythia-2.8b-deduped	2.8B
facebook	xglm-2.9B	2.9B
Qwen (Qwen et al., 2025)	Qwen2.5-3B	3B
cerebras (Dey et al., 2023)	btlm-3b-8k-base	3B

Continued on next page

Table 3 – Continued from previous page

Organization	Model Name	Parameters
openlm-research (Geng and Liu, 2023)	open_llama_3b_v2	3B
rinna (Sawada et al., 2024)	bilingual-gpt-neox-4b	4B
Dampish	StellarX-4B-V0	4B
facebook	xglm-4.5B	4.5B
Salesforce	codegen-6B-multi	6B
EleutherAI (Wang and Komatsuzaki, 2021)	gpt-j-6b	6B
Salesforce	codegen-6B-nl	6B
Salesforce	codegen-6B-mono	6B
cerebras	Cerebras-GPT-6.7B	6.7B
facebook	opt-6.7b	6.7B
EleutherAI	pythia-6.9b-deduped	6.9B
EleutherAI	pythia-6.9b	6.9B
Qwen (Bai et al., 2023)	Qwen-7B	7B
aisingapore (Lowphansirikul et al., 2021)	sea-lion-7b	7B
bigscience	bloom-7b1	7B
google (Gemma Team et al., 2024a)	gemma-7b	7B
mosaicml (MosaicML NLP Team, 2023)	mpt-7b	7B
openlm-research	open_llama_7b	7B
tiiuae (Institute, 2023)	falcon-7b	7B
allenai (for AI, 2024)	OLMo-7B-hf	7B
huggyllama (Touvron et al., 2023a)	llama-7b	7B
LLM360 (Liu et al., 2023)	Amber	7B
LLM360	CrystalCoder	7B
facebook	xglm-7.5B	7.5B
meta-llama (AI, 2023)	Meta-Llama-3-8B	8B
google	gemma-2-9b	9B
01-ai (01. AI et al., 2025)	Yi-9B	9B
EleutherAI	pythia-12b	12B
EleutherAI	pythia-12b-deduped	12B
cerebras	Cerebras-GPT-13B	13B
meta-llama (Touvron et al., 2023b)	Llama-2-13b-hf	13B
Qwen	Qwen1.5-14B	14B
Qwen	Qwen2.5-14B	14B
Salesforce	codegen-16B-nl	16B
Salesforce	codegen-16B-mono	16B
EleutherAI	gpt-neox-20b	20B
mosaicml	mpt-30b	30B
Qwen	Qwen2.5-32B	32B
Qwen	Qwen1.5-32B	32B
AbacusResearch	Jallabi-34B	34B
01-ai	Yi-34B	34B
01-ai	Yi-34B-200K	34B
meta-llama	Llama-2-70b-hf	70B
meta-llama (AI, 2023)	Meta-Llama-3.1-70B	70B

Continued on next page

Table 3 – Continued from previous page

Organization	Model Name	Parameters
meta-llama	Meta-Llama-3-70B	70B
Qwen (Qwen et al., 2025)	Qwen2-72B	72B
Qwen	Qwen2.5-72B	72B
Qwen	Qwen1.5-110B	110B

B List of all architectural and data features

B.1 Architectural Features

Note that features in this section are collected from official documentation (e.g. huggingface model/-data cards or original papers).

- **Total parameters** - the total number of parameters (embedding included) in the model. Note that we only include decoder-only dense models.
- **Dimension** - the embedding dimension.
- **Num heads** - the number of attention heads.
- **MLP ratio** - the ratio of $\frac{\text{FFN dimension}}{\text{embedding dimension}}$.
- **Positional Embeddings** - the type of positional embedding. This is either non-parametric (sinusoidal or fixed embeddings), learned (just learned as a vector per position), rope (rope embeddings), or alibi (technically not an embedding, but included here due to its functional purpose)
- **LayerNorm** - the type of layernorm applied. This is either non-parametric (just an arithmetic based normalization), parametric (similar, but with some learnable parameters such as scaling/biases), and RMSNorm (a simplified version of parametric)
- **Attention variant** - The broad type of attention used. This is either full (vanilla attention), local (each token position only attends to positions around it), mqa (multi-query attention), or gqa (grouped-query attention)
- **Biases** - whether or not bias terms are present in parts of the model. Either none (no biases), attn only (only in attention layers), ln only (only in layer norm)
- **Block type** - whether or not the transformer blocks are computed in parallel at all. Sequential indicates not, while parallel indicates some parallelism in attention or FFN layers.
- **Activation** - the activation function used. Either relu, gelu/gelu variations, silu, or swiglu.
- **Sequence length** - the sequence length.
- **Batch instances** - the batch size used during pretraining.

B.2 Data Features

Note that features in this section are collected from official documentation (e.g. huggingface model/-data cards or original papers).

- **Total tokens (B)** - total number of tokens used during pretraining, measured in billions (converted to log scale)
- **% Web in Pretraining** - Percentage of pretraining data from general web sources.
- **% Code in Pretraining** - Percentage of pretraining data that consists of code.
- **% Books in Pretraining** - Percentage of pretraining data from books.
- **% Reference in Pretraining** - Percentage of pretraining data from reference sources.
- **% Academic in Pretraining** - Percentage of pretraining data from academic sources.
- **% English in Pretraining** - Percentage of English text in the pretraining data.

B.3 Freegen-derived Features

These features are derived from model generations. For each model, 5–10k generations are extracted and the following metrics are aggregated (by mean and standard deviation). However, bigram entropy, the educational classifier score, and domain classifications are exceptions, as they are computed once across all generations.

We use Stanza (Qi et al., 2020) to generate the parse-based features after classifying generations by language. We only include languages that are supported by stanza in the final set of generations that the parse features are based on.

B.3.1 Generation Length & Basic Statistics

- **Mean Character Length** – Average number of characters per generation (capped at 2048).
- **Mean Tokens Generated** – Average number of tokens per generation.
- **Mean Sentences** – Average number of sentences per generation.
- **Mean Words** – Average number of words per generation.
- **Mean Words per Sentence** – Average number of words per sentence.

B.3.2 Constituency Parse Features

- **Mean Depth of Deepest Parse Tree** – Average maximum constituency tree depth per generation.

- **Mean Depth of Parse Trees** – Average constituency tree depth across all sentences/phrases.

- **Mean Word Depth** – Average depth of words within constituency trees.

- **Mean Word Depth Variation** – Average standard deviation of word depths across sentences/phrases.

B.3.3 Dependency Parse Features

- **Mean 90th-Percentile Dependency Head Distances** – For each generation, compute the 90th-percentile of the linear distances between words and their dependency head, then average these values.

- **Mean Maximum Dependency Head Distances** – Average maximum distance from any word to its dependency head per generation.

- **Mean Median Dependency Head Distances** – Average median dependency-head distance per generation.

- **Mean Maximum Dependency Root Distances** – Average maximum distance from any word to the sentence root per generation.

- **Mean Mean Dependency Root Distances** – Average of the mean distances from words to the sentence root per generation.

- **Mean Median Dependency Root Distances** – Average of the median distances from words to the sentence root per generation.

B.3.4 Domain Classification Features

- **% Generated Academic-like Text** – Percentage of generations classified as academic-like.

- **% Generated Books-like Text** – Percentage of generations classified as books-like.

- **% Generated Code-like Text** – Percentage of generations classified as code-like.

- **% Generated Reference-like Text** – Percentage of generations classified as reference-like.

- **% Generated Specialized Text** – Percentage of generations classified as specialized (e.g., music scores, chess PGNs, biomedical data).

- **% Generated Web-like Text** – Percentage of generations classified as web-like.

B.3.5 Classifier and Language Metrics

- **Mean Educational Classifier Score** – Average score assigned by the educational classifier.

- **% Generated English Text** – Average percentage of text generated in English.

B.3.6 Lexical Diversity and Entropy Metrics

- **Mean Bigram Entropy** – Average entropy computed on bigrams across generations.

- **Type-Token Ratio** – Average ratio of unique tokens to total tokens.

- **Unique Tokens** – Average number of unique tokens per generation.

B.3.7 Lexical and Stylistic Features

- **Content-Function Ratio** – Ratio of content words (nouns, verbs, adjectives, adverbs) to function words.

- **Question Words Ratio** – Ratio of question-related words (e.g. how, what, why, when, where, who, which, whose) per 100k words.

- **Imperative Words Ratio** – Ratio of imperative words (e.g. do, make, consider, take, use, ensure, check, build, apply, run, create, find, go, try, turn, start, stop, put, keep, leave, get, move) per 100k words.

- **Conjunctions Ratio** – Ratio of conjunction words (e.g. and, but, or, so, because, although, however, therefore, yet) per 100k words.

- **Instruction Words Ratio** – Ratio of instruction-oriented phrases (e.g. “Question:”, “Answer:”, “Instruction:”, “User:”, “Assistant:”, “Q:”, “A:”) per 100k words.

- **Numbers Ratio** – Ratio of numerical tokens in the generated text.

Task Name	# Models Evaluated
Commonsense Reasoning / NLI	
ANLI	82
HellaSwag	92
Winogrande	92
XNLI	82
Math / Logic	
GSM8K	92
LogiQA2	82
MathQA	82
General Knowledge	
ARC Challenge	92
Lambada	92
MMLU	92
Other	
TruthfulQA	92
HumanEval	91

Table 4: Number of models evaluated for each benchmark task. Note that some models encountered technical errors when being loaded or in lm-eval-harness. The number of models will continue to be updated.

C List of all evaluations and settings

Although we ideally would evaluate the full cross-product of models and tasks, we found that due to some models being incompatible with LM Evaluation Harness and compute constraints we could not evaluate all 92 models on every dataset. We list in Table 4 the number of evaluations we currently have for each benchmark and will continue to fill out evaluations in the database.

D Task Deviations from Kaplan-style Scaling Laws

In Table 5, we document the R^2 value for a fitted power law on the performance of each model.

E Free-generation Domain Classification

We classify model generations into top-level domains with GPT-4o-mini. We found that this multi-stage prompt Listing 1, Listing 2 had reasonable precision on a sample of Dolma by domain (Soldaini et al., 2024), so use it to classify free-generations.

Benchmark	R^2
gsm8k	0.85
arc challenge	0.82
hellaswag	0.80
winogrande	0.80
mmlu 5-shot	0.80
mmlu 0-shot	0.74
mathqa	0.70
ANLI	0.61
humaneval	0.61
lambada	0.55
LogiQA2	0.50
XNLI	0.41
truthfulqa	0.29

Table 5: Overview of R^2 values by benchmark.

F Domain Classifier Validation

To validate the reliability of the 4o-mini based classifier, we had an author of this paper annotate 300 selected samples from three pretraining datasets (the Pile, the SmolLM corpus, and RefinedWeb) according to the same annotation standards used in Appendix E. Samples annotated as "unknown" or "incoherent" by either the model or the human annotator were excluded, as these samples are not included in computing the domain mix.

After filtering, we analyzed 258 text samples and found that the human annotator and the model had an 85.8% absolute agreement, and a Cohen's κ of 0.746, indicating high agreement between human classifications and the model's classifications.

G Free-generation Validation

To validate our free-generation approach as a proxy for pretraining data composition, we analyzed correlations between the free-generation features of models and their pretraining data.

Namely, for models trained on three open pretraining datasets (the Pile, the SmolLM corpus, and Refinedweb), we compared the features of their free-generations to features produced by the same taggers and the LM-based classifier (Appendix E) on a randomly sampled 1M document subset of the pretraining corpora. Due to costs, for the domain classification 5k examples from the 1M were used per corpus. The 1M documents were uniformly sampled with reservoir sampling.

Additionally, we calculated two holistic model-wise correlations, which measure how well each

Domain	Pearson r	p-value
web	0.917	7.55e-12
reference	0.832	3.99e-08
academic	0.824	7.21e-08
code	0.679	7.14e-05
books	0.374	5.02e-02

Table 6: Domain correlations between pretraining data and free-generations. Web content shows the strongest Pearson correlation, suggesting models most faithfully reproduce web distribution patterns, while books content shows the weakest relationship.

Feature	Pearson r	p-value
conjunctions_ratio	0.690	4.80e-05
question_words_ratio	0.554	2.21e-03
numbers_ratio	0.449	1.66e-02
imperative_verbs_ratio	0.443	1.83e-02
char_len	0.214	2.75e-01

Table 7: Linguistic feature correlations between pretraining data and free-generations. Connective elements like conjunctions show stronger correlations, while structural features like character length are less preserved in model generations.

model's complete generation profile matches its training data:

1. **Domain-level correlations:** For each domain category (web, code, academic, books, reference), we computed the correlation between the percentage of that domain in the model's documented pretraining data and the percentage of free-generations classified into that category.

2. **Feature-level correlations:** For linguistic features (conjunctions ratio, imperative verbs ratio, question words ratio, numbers ratio, character length), we assessed how well each feature in free-generations correlates with corresponding features in pretraining data.

These model-wise correlations are notably higher than individual feature correlations, indicating that models preserve the overall "fingerprint" of their training data better than any single dimension alone.

H Free-generation Examples

Llama-2-70B, web-like: Gwen Stefani Couldn't Recognize Her First Song at an Event with Blake Shelton What's the first song you ever wrote and or recorded? And how would you feel if you never heard it again until you were Facetiming with hundreds of thousands of people while attending a big country music event? That's exactly what happened to Gwen Stefani, when she was honored at the Country Radio Seminar event this week. In Nashville, . One can only imagine that when you've had as crazy a career as she's had, it may be a chore to actually remember all of the milestones over the years, but just imagine also doing it on the spot, for everyone to see. Well, as it turns out, the doubly talented Dolly Parton was set to introduce Gwen as "an artist who broke barriers" when she cited the lyrics from one of Stefani's first-ever "country" songs… "year 3,000." The diplomatic "The Voice" coach Blake Shelton interjected at the video she was showing at that moment, asking "was there other work done ....

phi-2, books-like: The sun glistened brightly in the evening sky, casting warm rays across the quaint little kitchen where a pair of diligent parents were preparing a scrumptious meal. The air was filled with anticipation as they embarked on a culinary adventure, their hearts filled with love and joy.

With their sleeves rolled up, the parents deftly wielded an assortment of sharp knives, each one an extension of their skill and precision. Like artists with their brushes, they delicately maneuvered through the ingredients, dancing with knives as their partners. The blades moved with a swiftness that defied the physics of everyday movements, gliding through the air with grace and purpose.

...

Qwen2-72B, reference (instruct)-like: Children: Family ticket (at most 2 adults and 2 children) $23.00 (1) Mr Black wants to see the film with his wife and two sons (they're both over 9), how much should he pay?____ A. $23. B. $46. C. $48. D. $49. (2)Mr Brown wants to see the film with his 5-year-old twin sons, how much should he pay____. A. $18. B. $23. C. $28.5 D. $32 (3)Lisa can't see snow in this film.____ A. True B. False (4). Jane is a maths teacher and she has two daughters (she's 12 and her sister is 8). They want to see the film. How much should she pay?____ A. $28.5 B. $27 C. $33 D. $26.5 (5) Mike is sixteen. Today is Saturday and he wants to see the film with his 40-year-old parents. They will pay____ for the tickets. A. $22.

OPT-6.7B, web-like: I don't think this is a question. It's a statement. I say get a dog, the world is better with dogs.

I said to myself the exact same thing. I don't know why you were down-voted.

because reddit is a bunch of people who hate dogs.

Codegen-16B-mono, code-like:

```python
from typing import Optional

from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()


class Item(BaseModel):
    name: str
    price: float
    is_offer: Optional[bool] = None


@app.get("/")
def read_root():
    return {"Hello": "World␣from␣FastAPI"}


@app.get("/items/{item_id}")
def read_item(item_id: int, q: Optional[str] = None):
    return {"item_id": item_id, "q": q}


@app.put("/items/{item_id}")
def update_item(item_id: int, item: Item):
    return {"item_name": item.name, "item_id": item_id}
```

Pythia-410M, academic-like (patent): This invention relates to new and improved methods and apparatus for removing solid waste material from the waste stream of a power station where the solid waste material is intended for disposal after completion of the power plant.

Various attempts have previously been made to remove solid waste from the waste stream of plants. This is true, for example, to the discharge of sludge which is generally collected into a sludge tank and washed out of the plant in a washing tank which is generally connected to an open drain outlet of the plant. This prior art is discussed by U.S. Pat. No. 3,623,579 which issued to G. R. Clark and described a method for treating the waste stream to remove solid waste by flocculating and flocculating and agitating the solids in a tank to break bonds between the solid particles.

Furthermore, an apparatus was described by U.S. Pat. No. 4,016,823 which describes a method in which liquid sewage is removed from the waste stream and from the sewage treatment plant where the solid waste being removed is to be treated to produce ammonia-purified water for use in bathing baths or soaps and where the sewage from the wastewater treatment plant is removed to the sewage processing plant where this sewage is mixed with water or treated as a fertilizer.

...

I XGBoost Settings

For the inner grid search, the maximum depth of trees was in $[2, 3, 5]$, while the learning rate was in $[0.01, 0.1, 0.3]$ and the number of trees was in $[50, 100]$.

J Selected Features by Task

In Table 8, we show the selected features per benchmark.

K LightGBM Results

The LightGBM version of Table 2 can be found in Table 9.

L SHAP Plots for remaining benchmarks

SHAP plots for the remaining benchmarks can be found in Figure 7 – Figure 15. Please note that lower scores are better for Brier score tasks (ANLI, XNLI, MathQA, LogiQA2)

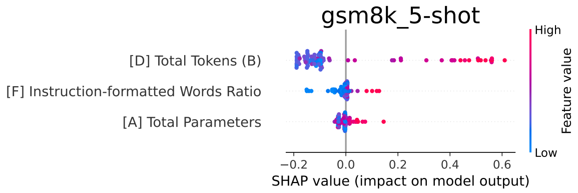


Figure 7: SHAP values for GSM8k

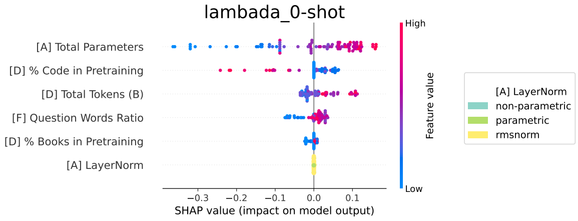


Figure 8: SHAP values for Lambada

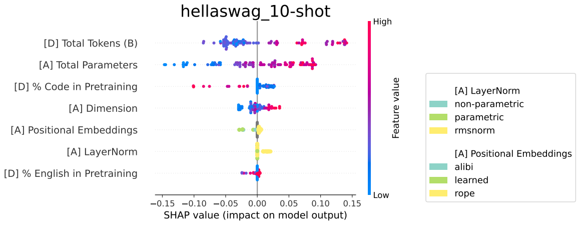


Figure 9: SHAP values for Hellaswag

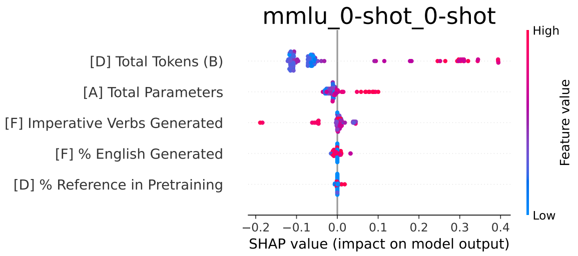


Figure 10: SHAP values for MMLU 0-shot

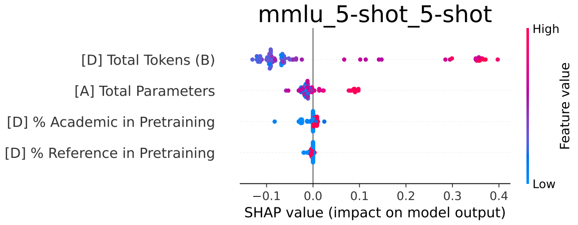


Figure 11: SHAP values for MMLU 5-shot

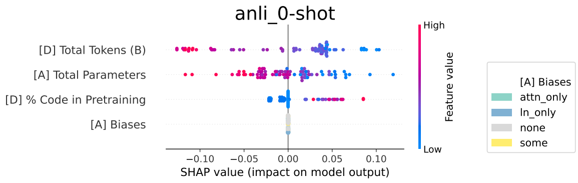


Figure 12: SHAP values for ANLI

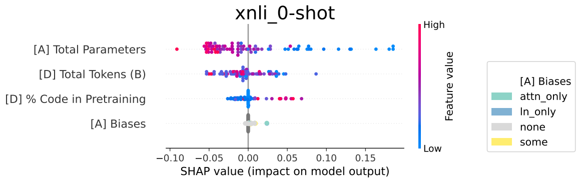


Figure 13: SHAP values for XNLI

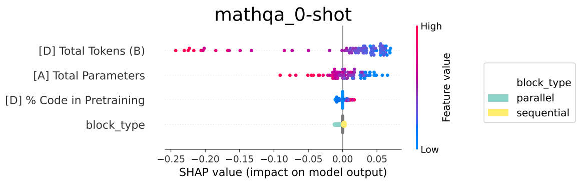


Figure 14: SHAP values for MathQA

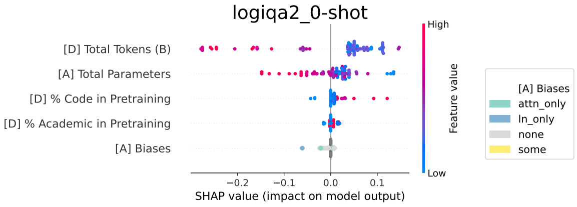


Figure 15: SHAP values for LogiQA2

M Details on confirmatory pretraining runs

M.1 Training

For our confirmatory experiments, we trained 460M parameter Llama-2 architecture models from scratch using the Megatron-Deepspeed library. We capped training tokens to 10B, while using a cosine learning rate schedule set to a length of 100B tokens (meaning that each checkpoint is approximately 10% through a "full" pretraining run). Training took place on one node per checkpoint, with 8 H100 GPUs. Each checkpoint took roughly 6 hours to train.

For our data mixes, we constructed various mixes by using the subsets of the Dolma v1 dataset. In the web vs. other experiments, we fixed the relative percentages of all other data sources while varying the web percentage.

The training configuration is as follows:

```
training:
  num_layers: 14
  num_attention_heads: 12
  seq_length: 2048
  num_kv_heads: 12
  hidden_size: 1536
  ffn_hidden_size: 4128
  tune_steps: 1000
  lr: 0.00015
  min_lr: 1.0e-5
  weight_decay: 1e-2
  grad_clip: 1.0
  lr_warmup_steps: 100
  save_interval: 2000
  eval_interval: 2000
  train_epochs: 1
  tp: 1
  micro_batch_size: 16
  global_batch_size: 512
  seed: 42
```

Aside from data mix, all experiments used identical hyperparameters to ensure fair comparison.

M.2 Evaluation

To assess the association of different data mixes with model performance, we evaluated our models on the following tasks:

1. **Natural language inference:** Lambada, winogrande, arc challenge
2. **Code generation:** Humaneval
3. **Math:** GSM8K
4. **Factuality:** TruthfulQA

Note that we do not select the full evaluation set due to time constraints. As LM eval harness does not implement perplexity/loss based evaluations for all tasks, we manually convert multiple-choice tasks to loss-based metrics, and mask out the prompt or question when calculating loss for all tasks.

M.3 Conversion to Loss-Based Metrics

To ensure consistent evaluation across different tasks and models, we converted various benchmark datasets to loss-based metrics. This approach allows for more direct comparison between models and clearer interpretation of improvements. Here's how we implemented loss calculations for each dataset type:

Multiple Choice Tasks (ARC Challenge, Winogrande, HellaSwag, TruthfulQA): For these datasets, we calculated two primary loss-based metrics:

- **Average Loss:** We computed the negative normalized log probability of the correct answer. For each question, we formatted the input as "Question + Answer Choice", then calculated the sequence log probability normalized by token length for each choice. The negative log probability of the correct answer was used as the loss.
- **Margin-based Loss:** For TruthfulQA specifically, we calculated a margin between truthful and non-truthful answers. This was computed as the negative of the difference between the best truthful answer's log probability and the best non-truthful answer's log probability. A lower loss indicates better differentiation between truthful and non-truthful information.

Generation Tasks (GSM8K, HumanEval, Lambada): For generation tasks, we calculated:

- **Answer Loss:** We computed the cross-entropy loss on the solution tokens only. Note that for Lambada, this is only the last word.

All log probabilities were normalized by sequence length as well.

M.4 Full Results

Exact loss values for the code vs. natural language mixes and the web vs. other mixes can be found respectively in Table 10 and Table 11.

N Publication bias correction

Data from PET-PEESE on architectural variations can be found in Table 12.

Listing 1: Multistage classification prompt.

```
<PROMPT 1>. [SYSTEM] You are a system tasked with classifying documents. First, determine if this
    document is relatively coherent. These documents are generated by language models, so they may
    not make sense. Classify a document as incoherent ONLY if it shows extreme repetition, code
    mixes in a way that does not make sense (such as different languages referencing entirely
    different subjects), or if it is mostly gibberish. Don’t worry about logic errors or factual
    inconsistencies. If multiple documents are mixed into one, classify it as incoherent. Respond
    ONLY with "incoherent" if the document is incoherent, otherwise respond with "not_incoherent"
[USER] Please classify the document as incoherent or not_incoherent.\nDocument: {document}

If not incoherent...
<PROMPT 2>. [SYSTEM] Determine if this document contains programming code. Look for:
1. Programming language keywords (def, class, import, etc)
2. Code blocks (marked with backticks, indentation patterns)
3. Stack Overflow-style Q&A about programming
4. File extensions (.py, .js, etc)
5. Documentation about code/config files

Respond ONLY with:
- "code" if ANY of these are present
- "not_code" otherwise
[USER] Please classify the document as code or not_code.\nDocument: {document}

If not code...
<PROMPT 3> [SYSTEM] For documents WITHOUT programming code, determine if this is web content. Web
    content includes news articles, social media and online forums, blog posts, shopping websites,
    and other general websites. This includes a wide variety of content, and anything that looks
    like it may be a web article at all should be included. Look for:
1. URLs or hyperlinks
2. Social media formatting (@mentions, #hashtags)
3. "Click here" or UI elements
4. Comment threads or forum posts
5. Shopping/e-commerce language
6. Bylines or author names
7. Descriptions of products or product features

Respond ONLY with:
- "web" if ANY of these are present
- "not_web" otherwise

[USER] Please classify the document as web or not_web.\nDocument: {document}

If not web...
<PROMPT 4> [SYSTEM] For documents WITHOUT programming code, determine if this is academic or
    patent-related content. Academic content consists of research papers and snippets of research
    in both sciences and humanities, as well as patent applications. Student essays or assignments
    should also be included in this category. Look for:
1. Citations or references
2. Latex formatting such as equations or tables
3. Formal academic language, not aimed at educating a general audience
4. Technical jargon or domain-specific terminology
5. Patent numbers or legal language (but not court documents, only patents)

Do NOT classify as academic if the document:
- Only uses occasional technical terms
- Is a popular science article or description of a scientific study, rather than the study itself
- Is educational but aimed at a general audience

Respond ONLY with:
- "academic" if ANY of these are present
- "not_academic" otherwise
[USER] Please classify the document as academic or not_academic.\nDocument: {document}
```

Listing 2: Multistage classification prompt (contd).

```
If not academic...
<PROMPT 5> [SYSTEM] For documents WITHOUT programming code, determine if this is a book, reference
     material (including media content), or a specific dataset. Books include literary works,
     fiction, and narrative nonfiction. Reference material includes wikipedia, dictionaries,
     textbooks and textbook like content, and encyclopedias. Please note that reference should also
     include instruction or human preference datasets for language model training. Media content
     includes podcasts, subtitles, and other media-related text. Specific datasets are unique and
     not covered by the other categories, such as biomedical datasets or molecules, chess PGNs or
     specific data formats not covered by any other category. Look for:

For the books category:
1. Chapter headings or book titles
2. Fictional character names or dialogue
3. For literary nonfiction, look for a more narrative and less didactic tone
4. Extended narrative prose or dialogue
Do NOT classify as books if the document:
- Only has a single dialogue snippet
- Could be a web article
- Is primarily informational or educational (use reference instead)

For the reference category:
1. Definitions or explanations of terms
2. Encyclopedic formatting
3. Textbook-like language
4. Explanations or examples meant to educate a reader
5. Chat formatting like 'User:/Assistant:' or similar tokens
6. Court documents or legal language (NOT patents)
7. Wikipedia headers such as 'references' or 'external links'

For the media category (should be classified as reference):
1. Audio or video timestamps
2. Subtitles or captions

For the specific datasets category:
1. Unique names or identifiers
2. Dataset-specific formatting
3. Data or metadata descriptions

If this seems to be a web document (social media, news, blogs, forums, shopping), you can also back
     off to the 'web' category.

Respond ONLY with:
- "books" if the document is a book
- "reference" if the document is reference material
- "specific_datasets" if the document is a specific dataset
- "web" if the document is web content
- "unknown" if none of these are present

[USER] Please classify the document as books, reference, media, specific_datasets, or
     unknown.\nDocument: {document}"
```

Table 8: Greedily-selected features per benchmark.

Benchmark	Selected Features
arc challenge (25-shot)	total params, pretraining summary total tokens billions, question words ratio, layer norm type, dimension, pretraining summary percentage code
gsm8k (5-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage reference, edu classifier std, pretraining summary percentage books
hellaswag (10-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage code, pretraining summary percentage reference, positional embeddings, pretraining summary percentage academic
mmlu 0-shot (0-shot)	total params, pretraining summary total tokens billions, layer norm type, activation, pretraining summary percentage code
truthfulqa (0-shot)	total params, pretraining summary total tokens billions, domain web pct mean, dep parse dep root dist max mean, pretraining summary percentage english, entropy mean, layer norm type
winogrande (5-shot)	total params, pretraining summary total tokens billions, question words ratio, layer norm type, pct english mean, positional embeddings, pretraining summary percentage books, pretraining summary percentage code, block type
anli (0-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage code, pretraining summary percentage web, pretraining summary percentage books, positional embeddings
logiqa2 (0-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage web, domain reference pct mean, dep parse dep root dist mean std, dep parse dep root dist median std
mathqa (5-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage books, num heads
xnli (0-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage web
lambada (0-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage code, block type
mmlu 5-shot (5-shot)	total params, pretraining summary total tokens billions, sequence length, biases, num heads, dimension, edu classifier mean, pretraining summary percentage academic
humaneval (0-shot)	total params, pretraining summary total tokens billions, pretraining summary percentage code, layer norm type, pretraining summary percentage english, biases

Benchmark	**Setting**	**Baseline MAE**	**Scaling Laws MAE**	**All Features MAE**
		Accuracy		
Arc Challenge	25-shot	13.23%	4.91%	3.61%
GSM8k	5-shot	15.65%	11.03%	5.78%
Hellaswag	10-shot	12.26%	4.29%	3.14%
Humaneval	0-shot	11.79%	8.61%	6.80%
Lambada	0-shot	16.89%	9.60%	6.39%
MMLU (0-shot)	0-shot	11.98%	9.12%	4.21%
MMLU (5-shot)	5-shot	12.25%	8.39%	3.05%
TruthfulQA	0-shot	3.72%	3.40%	2.61%
Winogrande	5-shot	10.14%	3.99%	3.09%
		Brier score (×100)		
XNLI	0-shot	7.22	4.70	4.32
ANLI	0-shot	9.48	6.14	6.05
MathQA	0-shot	7.57	3.89	2.95
LogiQA2	0-shot	12.74	8.77	8.29

Table 9: MAE comparison of Scaling Laws and All Features predictors versus Baseline. Note that a significance test was not carried out for LGBM, so this reflects results for one run, though a hyperparameter search is still carried out over the same values as in Appendix I for both predictors. Brier scores are scaled ×100 for comparability. Both predictors here use LGBM.

Data Mix	**Lambada**	**Humaneval**	**Winogrande**	**Arc challenge**	**GSM8k**
NL 70/Code 30	3.426	**1.331**	3.967	3.967	2.295
NL 75/Code 25	3.426	**1.331**	3.966	3.651	2.295
NL 80/Code 20	3.419	1.350	3.989	3.685	**2.293**
NL 90/Code 10	**3.406**	1.533	**3.944**	**3.620**	2.300

Table 10: Loss-based evaluation metrics across different data mix ratios

Data Mix	**Margin Loss**	**Accuracy (%)**
Web 30/Other 70	0.2363	**28.86**
Web 50/Other 50	0.**2342**	28.40
Web 90/Other 10	0.2462	28.03

Table 11: TruthfulQA evaluation metrics across different web/other ratios

Table 12: Bias-corrected architecture effects on accuracy (defaults-only; PET-PEESE). Positive values mean the feature level outperforms the rest of the pool within a task. Effects are in percentage points (pp); CIs are 95%.

Feature	Level	k	Chosen	Effect (pp) [95% CI]	Egger p
Positional embeddings	alibi	6	PEESE	−2.0 [−7.8, +3.9]	0.013
Positional embeddings	learned	6	PET	+5.4 [−7.2, +17.9]	0.240
Positional embeddings	ROPE	6	PEESE	+1.5 [+0.9, +2.2]	0.076
LayerNorm type	non-parametric	6	PEESE	−3.4 [−8.6, +1.8]	0.031
LayerNorm type	parametric	6	PEESE	−1.3 [−2.7, −0.0]	0.021
LayerNorm type	RMSNorm	6	PET	+2.3 [+0.9, +3.8]	0.133
Attention variant	full	6	PEESE	+1.1 [+0.2, +2.0]	0.014
Attention variant	GQA	6	PET	+3.4 [+0.6, +6.3]	0.290
Attention variant	local,full	6	PET	+1.3 [−1.8, +4.3]	0.358
Attention variant	MQA	6	PEESE	0.0 [0.0, 0.0]†	0.038