TO CODE, OR NOT TO CODE? EXPLORING IMPACT OF CODE IN PRE-TRAINING

Viraat Aryabumi, Yixuan Su, Raymond Ma, Adrien Morisot, Ivan Zhang,
Acyr Locatelli, Marzieh Fadaee, Ahmet Üstün, Sara Hooker
{viraat,ahmetustun,sarahooker}@cohere.com

ABSTRACT

Including code in the pre-training data mixture, even for models not specifically designed for code, has become a common practice in LLMs pre-training. While there has been anecdotal consensus among practitioners that code data plays a vital role in general LLMs' performance, there is only limited work analyzing the precise impact of code on non-code tasks. In this work, we **systematically** investigate the impact of code data on general performance. We ask "*what is the impact of code data used in pre-training on a large variety of downstream tasks beyond code generation*". We conduct extensive ablations and evaluate across a broad range of natural language reasoning tasks, world knowledge tasks, code benchmarks, and LLM-as-a-judge win-rates for models with sizes ranging from 470M to 2.8B parameters. Across settings, we find a consistent results that code is a critical building block for generalization far beyond coding tasks and improvements to code quality have an outsized impact across all tasks. In particular, compared to text-only pre-training, the addition of code results in up to relative increase of 8.2% in natural language (NL) reasoning, 4.2% in world knowledge, 6.6% improvement in generative win-rates, and a 12x boost in code performance respectively. Our work suggests investments in code quality and preserving code during pre-training have positive impacts.

1 INTRODUCTION

The role of data has taken on critical significance in recent breakthroughs. State-of-the-art models highlight the importance of the pre-training data mixture, diversity of data sources (Brown et al., 2020; Longpre et al., 2023; Singh et al., 2024) combined with compute availability as key drivers on performance (Dubey et al., 2024; Üstün et al., 2024; Team et al., 2023; Aryabumi et al., 2024). A critical question is *what properties of data impart the best general performance?*

Perhaps surprisingly, code is often included in pre-training even if a model is not explicitly intended to generate high-quality code. Code datasets differ significantly in terms of structure and textural characteristics from high-quality web datasets (Wikimedia; Raffel et al., 2019). Despite this, several previous generations of LLMs like PaLM (Chowdhery et al., 2022), Gopher (Rae et al., 2022) and Bloom (Workshop et al., 2023) that were not explicitly intended to support code generation, included code data together with high-quality natural language data in their pre-training mixture.

In current state-of-the-art models, it is an accepted norm to not only include code data but further increase the proportion – for instance, Llama 3 (Dubey et al., 2024) has four times more code data in proportion (17%), of its pre-training mixture than Llama 2 (4.5%) (Touvron et al., 2023). While there has been consensus anecdotally among practitioners that code data plays a vital role in LLMs' performance, there has been only limited work analyzing the precise impact of code on non-code tasks. Prior work shows particular side benefits of the inclusion of code data, such as impact on scaling in limited data regime (Muennighoff et al., 2023a), entity tracking capabilities (Kim et al., 2024), and mathematical reasoning (Razeghi et al.). However, there has been no exhaustive study to date that **systematically** investigates the impact of code data on general performance. In this work, we ask "*what is the impact of code data used in pre-training on a large variety of downstream tasks beyond code generation?*".

We embark on an exhaustive set of large-scale controlled pre-training experiments. This includes a consideration of where in the training process adding code is beneficial, code proportions, the role of scaling, and the quality and properties of code added. While a costly endeavor to perform these ablations in a rigorous way, we find consistent and valuable results that code provides critical improvements to non-code performance. In particular, compared to text-only pre-training, for our best variant, the addition of code results in relative increase of 8.2% in natural language (NL) reasoning , 4.2% in world knowledge, 6.6% improvement in generative win-rates, and a 12x boost in code performance respectively. Further performing cooldown with code, improves NL reasoning by 3.7%, World knowledge by 6.8%, and code by 20%, relative to cooldown without code and leads to a 4.1% additional win-rate increase.

Here, several factors matter including getting the proportion of code correct, improving the quality of code by including synthetic code and code adjacent data such as commits, and leveraging code across multiple stages of training including during cooldown. Our results suggest code is a critical building block for generalization far beyond coding tasks and improvements to code quality have an outsized impact on performance. We conduct an extensive evaluation on a broad range of benchmarks, which cover world knowledge tasks, natural language reasoning, and code generation, as well as LLM-as-a-judge win-rates. Across experiments on models ranging from 470 million to 2.8 billion parameter models, we find the following detailed results:

1. **Code provides critical improvements to non-code performance.** Initialization with code pre-trained models results in improved performance for natural language tasks. In particular, compared to text-only pre-training, for our best variant, the addition of code results in a relative increase of 8.2% in NL reasoning, 4.2% in world knowledge, 6.6% improvement in generative win-rates, and a 12x boost in code performance respectively.
2. **Code quality and properties matter.** Using markup-style programming languages, code-adjacent datasets such as GitHub commits and synthetically generated code improves the performance in pre-training. In particular, training on a higher quality synthetically generated code dataset results in a 9% and 44% increase in natural language reasoning and code performance, respectively, compared to web-based code data in pre-training. Additionally, continual pre-training from a code model that includes synthetic data results in 1.9% and 41% relative increases in natural language reasoning and code performance respectively, compared to initialization from a code model that does not include code data.
3. **Code in cooldown enables further improvement across all tasks.** Including code data in pre-training cooldown, where high-quality datasets are up-weighted, leads to an increase of 3.6% in NL reasoning, 10.1% in world knowledge, and 20% in code performance relative to no cooldown. More significantly, cooldown with code beats the baseline (no cooldown) by 52.3% win-rates, where win-rates are 4.1% higher compared to cooldown without code.

2 METHODOLOGY

We describe the details of our Pre-training Data (§ 2.1), Evaluation (§ 2.2), Training and Model details (§ 2.3). Figure 1 shows the high-level experimental framework. Precise details for each experiment and their results are presented in Section 3.

2.1 PRE-TRAINING DATA

In this section, we describe the details of our pre-training and cooldown datasets. We aim to evaluate the role of code in pre-training, following current state-of-art practices. Hence, we consider pre-training runs that consist of two phases: **1) continued pretraining** and **2) cooldown**. Continued pre-training refers to training a model that is initialized from a pre-trained model and trained for a fixed token budget. Cooldown (Team et al., 2023; Parmar et al., 2024) involves up-weighting high-quality datasets and annealing the learning rate for a relatively small number of tokens during the final stages of training. This up-weighting of high-quality datasets for a smaller amount of steps at the end of training can significantly boost model quality.

Text dataset. We use the SlimPajama pre-training corpus (Soboleva et al., 2023) as our source of natural language text data. SlimPajama is a de-duplicated, quality-filtered, multi-corpora, open-source dataset based on RedPajama-1.2T (Computer, 2023). SlimPajama consists of documents

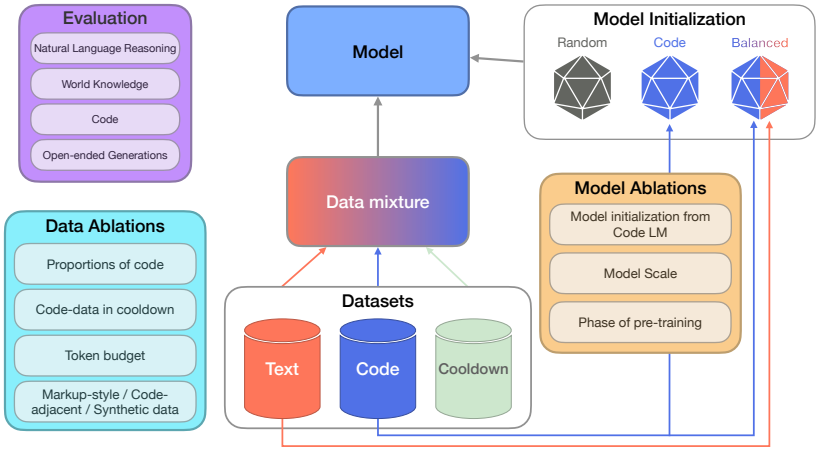


Figure 1: **Overview of our experimental framework**: We exhaustively evaluate the impact of code by varying: 1) the proportion of code in pre-training, 2) code quality and properties, 3) model initialization, 4) model scale, and 5) stage of training at which code is introduced. We evaluate the resulting model on a wide-ranging set of tasks, including natural language reasoning, world knowledge, code, and open-ended generations.

from CommonCrawl, C4, GitHub, Books, ArXiv, Wikipedia, and StackExchange. We filter out all documents from GitHub and StackExchange to remove code and code-adjacent data sources and ensure this is a text-only source. SlimPajama has a total of 627B tokens. After removing all code sources, this results in our text pre-training corpus with a total of 503B tokens.

Code datasets. To explore the impact of different properties of code data, we use multiple sources of code in our experiments:

- WEB-BASED CODE DATA: For our main source of code data, we start with the Stack dataset (Kocetkov et al., 2022) that was used to train StarCoder (Li et al., 2023a). The Stack consists of permissively licensed code data scraped from GitHub. We apply quality filters[1] and restrict to the top 25 programming languages based on document count [2]. After all filtering steps, the size of the code-only and markup subset is 139B tokens.
- MARKDOWN DATA We also separately process markup-style languages such as Markdown, CSS, and HTML.[2] After all filtering steps, the size of this markup subset is 180B tokens.
- SYNTHETIC CODE DATA: To ablate the quality of the code dataset, we use a proprietary synthetically generated code dataset that consists of Python programming problems that have been formally verified. We treat this as a high-quality source of code data (See the details in § 3.4). The final synthetic dataset consists of 3.2B code tokens.
- CODE ADJACENT DATA: Finally, to explore different properties of code data, we include a version of the code data which includes auxiliary data such as GitHub commits, jupyter notebooks, StackExchange threads. For GitHub commits, and jupyter notebooks we use the datasets provided as part of the Stack (Kocetkov et al., 2022). We use the version of StackExchange that is part of SlimPajama (Soboleva et al., 2023). In total we have 21.4B tokens of code-adjacent data.

Pre-training cooldown datasets. Cooldown involves up-weighting higher quality datasets for the final steps of pre-training and has been found to improve performance on downstream tasks (Parmar et al., 2024; Team et al., 2023), in particular to impart instruction-following capabilities. We choose a cooldown mixture comprising high-quality text, math, code, and instruct-style text datasets.

[1] See Appendix C.1 for details about quality filters
[2] Refer to Appendix C.2, C.3 for the full list of programming and markup-style languages included

2.2 EVALUATION

Our goal is to systematically understand the impact of code on general performance, which requires a broad evaluation suite that extends to a large variety of downstream tasks beyond code generation. To achieve this, we evaluate models on benchmarks that are reasonable proxies for model ability on **1)** world knowledge, **2)** natural language reasoning, and **3)** code performance. In addition, we report win-rates as evaluated by an LLM-as-a-judge. Table 1 (Appendix A) shows the full evaluation suite and their respective grouping, along with the metric used.

For **World knowledge**, we use benchmarks to measure knowledge memorization, retrieval, and question answering capability given context. We include Natural Questions Open (Kwiatkowski et al., 2019), and TriviaQA (Joshi et al., 2017) as the datasets. **Natural language reasoning** suite consists of 11 benchmarks that involve natural language based reasoning such as Question Answering, natural language inference (NLI), sentence completion, co-reference resolution, and general intelligence. We include the full list of the constituent benchmarks with references in Table 1. Finally, while our main focus is general performance, we also want to measure any changes to code generation performance. For **Code** benchmarks, we focus on the function completion task where we use HumanEval-Python (Chen et al., 2022) and MBPP (Austin et al., 2021).

We evaluate performance at two scales: 470M to 2.8B parameter models. At 470M scale, model capabilities are limited, thus to ensure fair comparisons, we only compare benchmarks for which all models achieve scores above random similar to Muennighoff et al. (2023a); Lozhkov et al. (2024).

LLM-as-a-judge win-rates. In addition to task-specific discriminative performance, to allow for a more holistic view across all performance measures, we also evaluate generative performance using LLM-as-a-judge win-rates. This is particularly valuable given recent work that has shown that as performance on open-ended generations improves, there is deterioration in traditional academic tasks (Üstün et al., 2024; Ouyang et al., 2022; Iyer et al., 2022; Muennighoff et al., 2023c). The use of LLMs-as-a-Judge benchmarks (Fu et al., 2023; Liu et al., 2023; Chiang & yi Lee, 2023; Shimabucoro et al., 2024) has gained traction as an alternative to performing human evaluation, which tends to be laborious and expensive (Wang et al., 2023; Boubdir et al., 2023). LLMs as evaluators compare two completions based upon detailed prompts and are reasonable proxies aligned with human preference (Üstün et al., 2024; Dubois et al., 2024).

We use the Dolly-200 English dataset (Üstün et al., 2024; Singh et al., 2024), which consists of 200 hand-picked examples from the Dolly-15K dataset (Conover et al., 2023). These prompts are open-ended and capture general-purpose non-code use cases making them a valuable proxy for how code impacts more fluid and often open-ended tasks. For our win-rate evaluations, we use Command-R+[3] as the LLM judge. Details about the prompt are provided in Appendix D.

2.3 TRAINING AND MODEL DETAILS

We use 470M and 2.8B parameters decoder-only auto-regressive Transformer models (Radford et al., 2019) that are trained with a standard language modeling objective. We use parallel attention layers, (Chowdhery et al., 2022; Wang & Komatsuzaki, 2021), SwiGLU activation (Shazeer, 2020), no biases in dense layers, and a byte-pair-encoding tokenizer with a vocabulary size of 256K. All models are pre-trained using AdamW (Loshchilov & Hutter, 2019) with a max sequence length of 8192, batch size of 512 and a cosine LR schedule with a warmup of 1325 steps.

Infrastructure. We use TPU v5e chips (Jouppi et al., 2017) for training and evaluation. All models are trained using Jax (Bradbury et al., 2018) framework. We pre-train 64 models in total. This is an enormous endeavour given the scale and computational resources required. Each pre-training run for 200B tokens takes 4736 TPU-chip hours for 470M and 13824 TPU-chip-hours for 2.8B parameter models. Each cooldown run for 40B tokens takes 1024 TPU-chip hours for 470M models.

3 RESULTS AND DISCUSSION

In this section, we will report descriptions and results for each experimental variants. We systematically investigate, **(1)** initializing an LLM with code pre-trained models (§ 3.1), and **(2)** the impact of

[3]https://huggingface.co/CohereForAI/c4ai-command-r-plus

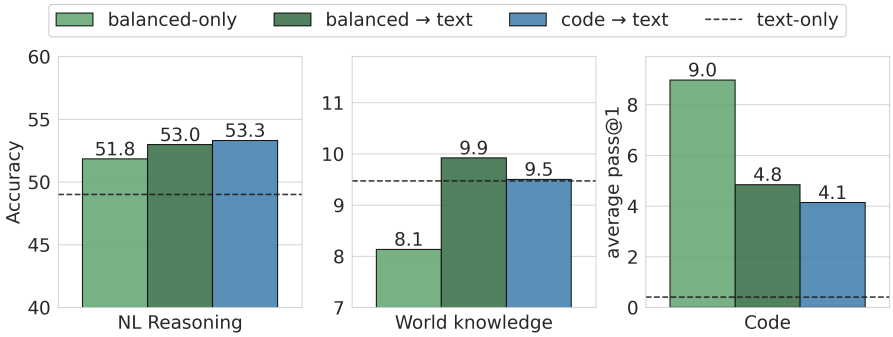


Figure 2: **Impact of initialization using code pre-trained models**: Initializing model training with code pre-trained models improves reasoning and code generation compared to text-only models, where the improvement is the most when continued pre-training with high percentage text (Balanced→Text, Code→Text). Note that these variants are designed to isolate the role of initialization, so do not include cooldown.

model scale (§ 3.2), **(3)** varying proportion of code in pre-training data (§ 3.3), **(4)** quality and properties of the code data (§ 3.4), **(5)** code data in pre-training cooldown (§ 3.5). Finally, we compare the resulting pre-training recipes (§ 3.6). Figure 1 shows the key levers of our experimental design.

3.1 Initializing an LLM with Code Pre-trained Models

We explore different initializations of pre-trained models to understand if using an LM with a large portion of code data as initialization improves the performance. These key ablations, along with their token counts, are summarized in Table 2. We briefly describe below:

- **Text LM** (text-only baseline): Pre-trained model *from scratch* using glorot-normal initialization (Glorot et al., 2011) on the text-only data for 400B tokens.
- **Balanced LM** (balanced-only): A model is trained with an equal ratio of code and text data (50% text and 50% code) in pre-training for 400B tokens.
- **Balance-initialized Text LM** (balanced → text): This model is initialized with a balanced LM (50% text and 50% code) and further pre-trained using text data for 200B tokens.
- **Code-initialized Text LM** (code → text): Different from other variants, this model is initialized with a code-LM which is pre-trained on a code dataset for 200B tokens. The code dataset contains a mixture of 80% code data and 20% markup-style code data. We then continually pre-train this model on text for another 200B tokens.[4]

Natural Language Reasoning As seen in Figure 2, initializing with 100% code pre-trained model (`code→text`) has the best performance for NL Reasoning benchmarks, and is closely followed by the `balanced→text` model. The `code→text` model and `balanced→text` model beat the `text-only` baseline on NL reasoning tasks by 8.8% and 8.2% relative improvement respectively. The `balanced-only` model improves upon the baseline by 3.2%. This shows that initialization from a pre-trained model with a mix of code has a strong positive effect on NL reasoning tasks. Further using a text mix with a small percentage of code for continual pre-training results in the best performance as evidenced by both the `code→text` and `balanced→text` models.

World Knowledge For World Knowledge tasks, we see that the `balanced→text` model has the best performance over all other variants, beating the `code→text` by 21% and `text-only` by 4.1% relative improvement. This suggests that performance on world knowledge tasks depends on a more balanced data mixture for initialization and a larger proportion of text in the continual pre-training stage. Overall, code data is still beneficial compared to text-only pre-training for world knowledge tasks.

[4]We use a 10% of code in text mixture data during continual pre-training of code-initialized models (`balanced→text`, `code→text`) to avoid a full distribution shift and maintain the benefits of code.

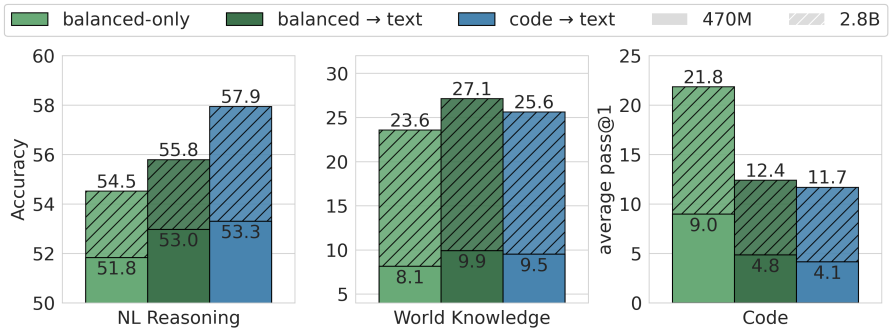


Figure 3: **Impact of model scale on different tasks.** We observe that scale provides pronounced gains across tasks of up-to 2.7x increase, however the overall trend remains the same across scales showing consistency of findings across model sizes.

Trade-offs between NL tasks and code generation For code generation, `balanced-only` achieves the best performance, where we see a 46.7% and 54.5% relative improvement over `balanced→text` and `code→text`. This is expected as `balanced-only` includes 50% code throughout pre-training. However, this model trades off better code generation with lower performance in NL tasks. `code→text` and `balanced→text` achieves 2.9% and 2.3% relative increase in NL reasoning, and 17.3% and 22.2% relative increase in world knowledge respectively compared to `balanced-only`.

Generative quality win-rates comparison Additionally, we compare the generative performance of each code variant (`code→text` and `balanced-only`) against the `text-only` model. We report win-rates and observe that the presence of code has a strong positive impact on generation quality. Both `code→text` and `balanced-only`) models beat the `text-only` variant by a 6.6% difference in win-loss rates. We again note that Dolly-200-English evaluation set we use for win-rate calculation is curated to reflect open ended questions and is a non-code evaluation. This confirms that code data in the pre-training mix does not *only* improves reasoning but also helps the model produce better quality generations.[5]

3.2 Impact of Scale

To understand if the findings of Section 3.1 transfer to larger models, we train 2.8B parameters models with the same token budget following the same model variants at 470M scale. Figure 3 shows the results of 2.8B models in comparison with 470M results.

Comparison between 2.8B and 470M models Scaling model size to 2.8B enables higher performance for all model variants in all task categories, compared to 470M results. In terms of average performance across NL reasoning and world knowledge, `balanced→text` model benefits from scaling-up by a 33.1% increase relative to the same model with 470M size. The improvement for `code→text` and `balanced-only` are 31.7% and 30% relative increase.

We find that the improvements in NL reasoning are relatively modest with 5.3%, 9.2%, and 5.2% relative gains for `balanced→text`, `code→text`, and `balanced-only` respectively. However, world knowledge and code performance nearly triples for all the model variants. In particular, 2.8B `balanced→text` results increase by 2.7x in world knowledge and 2.5x in code evaluation compared to 470M.

Trends between model variants in 2.8B Notably, in terms of initialization with code pre-trained models, the same trends seen in 470M parameter scale hold at 2.8B models. `code→text` and `balanced→text` models improve over `balanced` models by 6.9% and 6.1% relative gain, however, fall significantly behind in code generation performance with 43.1% and 46.3% relative drop. These results show that the trade-off between NL tasks and code generation increases with the model size. Overall our experiments scaling to a larger size shows that our results hold and are consistent with the trends we observe at 470M parameter ablations.

[5]We include the extended Win-rates for these experiments in Appendix E.

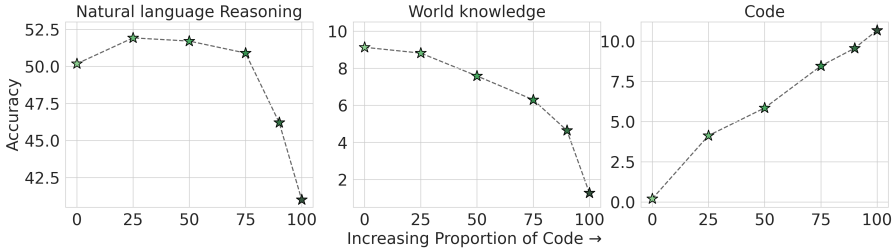


Figure 4: **Impact of the proportion of code in pre-training for different tasks**: We observe that as the code proportion of pre-training increases, the performance on code tasks increases linearly. In contrast, there is more sensitivity for NL reasoning and World knowledge tasks and an optimal range of code proportions where benefits are most tangible. Model size is 470M parameters and trained for 200B tokens.

3.3 Code Data Proportion in Pre-training

In these experiments, we ablate the proportions of code data in the pre-training mixture to understand what is the optimal amount of code to maximize performance on non-code tasks. Here, we focus on the first phase of pre-training with random initialization. We train six models for 200B tokens with increasing code proportions: 0%, 25%, 50%, 75%, 90%, and 100%. The remaining proportion is filled with text data. For each variant, we train a new model independently in order to carefully ablate the impact of varying code proportions.

Natural Language Reasoning and World Knowledge For NL Reasoning, as the amount of code increases, in Figure 4 we see an increase in performance compared to a text-only (0% code) model. The best performance is from a model with 25% code and 75% text, with a 3.4% relative improvement over a model with 0% code. While performance is maintained up to 75% code, it starts to rapidly erode at higher proportions with a sharp relative drop of 18.3% when the model is trained on 100% code compared to a model with no code.

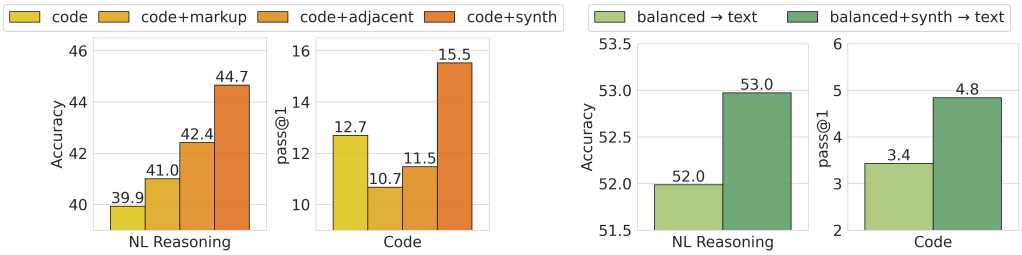
For World Knowledge tasks, we see an inverse relationship with increasing the amount of code. As seen in Figure 4 middle inset, there is a slight relative drop of 3.4% at 25% code and this relative drop worsens to 31% at 75% code compared to the no-code model. The fully code model (100% code) is unable to perform in world knowledge task (86% drop relative to text-only) as there are no data sources to acquire the required knowledge in the pre-training mix.

Performance on Code For code evaluation, there is a linear increase in performance as the amount of code increases, with the best model being a code-only model. As observable in Figure 4 right inset, the 100% code leads to a 2.6x increase in the code benchmarks compared to the 25% code model. As expected, for the model with 0% code, the average pass@1 score drops to 0.

3.4 Influence of Code Quality and Properties on General Performance

In this section, we investigate the properties of code data by varying its quality and composition. We study this firstly **(a)** from the perspective of training *from scratch*, as we want to isolate the exact effects of different properties of code data. Secondly **(b)**, we incorporate the best variant of the code data (high-quality synthetic code), in our continual pre-training experiments to see if the impact of the code quality transfer. We report performance on NL reasoning and Code tasks.

We study the effect of the following properties: **(1)** Markup-style Data: we separate markup-style programming languages (§ 2.1) from the rest of web-based code (Appendix C.3). We replace 20% of code-only tokens with markup-style tokens. **(2)** Code Adjacent Data: Instead of using purely web-based code data, we replaced 15% percentage of code tokens with code-adjacent datasets - GitHub issues (5%), StackExchange (5%) and Jupyter Notebooks (5%), resulting in a `code-adjacent` model. **(3)** Code Quality: To control the quality of the code, we replaced 10% of existing code tokens with a synthetically generated high-quality code dataset. The remaining proportions of web-based code data are kept the same, resulting in a `code-synth` model.



(a) Code-only Pre-training

(b) Continual Pre-training

Figure 5: **Impact of using different properties of code data**: **(a)** As the most impactful code data source, synthetically generated high-quality code improves NL reasoning and code performance for code pre-training. **(b)** These improvements with synthetically generated high-quality code data also transfer the continual pre-training setting. All models are of size 470M parameters.

Code-only pre-training We compare the above variants to a model that is trained only on web-based code data (`code`) from the stack dataset (Kocetkov et al., 2022), which forms our baseline model. All the variants are pre-trained using the same amount of tokens (200B) for fair comparison.

In Figure 5a, we evaluate the impact of code quality and code composition. We observe that across all variants, including diverse code sources and also synthetic code leads to gains in natural language performance relative to `code`, however, only synthetically generated code improves the code benchmarks. We relate this to our code evaluation where we measure performance in *python*, thus different programming languages or code-adjacent data slightly decrease the results. Here, `code+markup` and `code+adjacent` leads to 2.8% and 6.3% relative improvement in NL reasoning compared to **code** (web-code-only), but cause 15.7% and 9.4% drop in code evaluation.

Our synthetic code data (`code+synth`) is the most impactful ablation. It is particularly impressive given its relatively small share of the overall dataset. Despite a small weighting of 10%, the inclusion of synthetic data leads to relative improvements of 9% on NL reasoning, and 44.9% on code benchmarks compared to the baseline of web-code-only. We note that the lifts observed for synthetic data are even more impressive given the limited amount of synthetic data available compared to code-adjacent data (3.2B tokens vs 21.4B tokens) or code+markup data (3.2B tokens vs 40B tokens), and the weighting during pre-training allocation (10% vs 15% vs 20% for synthetic data, code-adjacent, code-markup respectively). This suggests a key future lever of improvement is increasing the proportion of such high-quality code data sources.

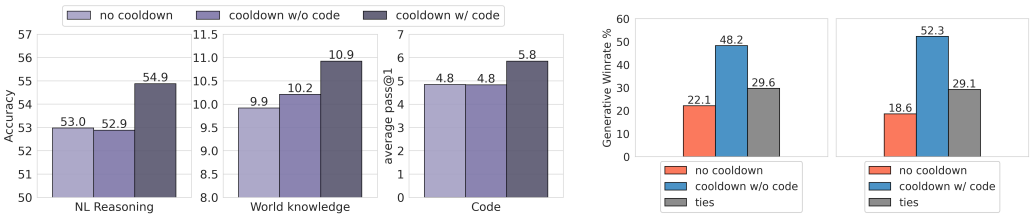
Continual pre-training Here, based on the findings from code-only pre-training, we incorporated `code+synth` into our best continual pre-training variant (`balanced+synth→text`). We compare this against the same variant without synthetic code data (`balanced→text`) to evaluate the benefits of synthetic data. We use the same amount of code and text tokens in these experiments.

As shown in Figure 5b, `balanced+synth→text` achieves 2% and 35% relative improvement over `balanced→text` in NL reasoning and code, respectively. This further confirms that even a small percentage of a high-quality code data, not only improves performance in code pre-training but also increases code and non-code performance after continual pre-training with text data.

3.5 Code in Pre-training Cooldown

In this section, we evaluate the impact of code at the final stage of pre-training. Here, we consider cooldown, where we up-weight high-quality text, math, code, and instruct-style datasets. We change the learning rate schedule from cosine-based to linear annealing with a final learning rate of $1e-6$. We evaluate the impact of code in cooldown by comparing 3 models: a pre-trained model before cooldown, cooldown without code data, and cooldown with 20% code data. For our pre-trained model, we use `balanced→text` as it is our best pre-trained variant. We preserve the same token budget across variants – 40B tokens which is 10% of the token budget of the pre-trained model.

Impact of code used during cooldown in different tasks In Figure 6a, we evaluate the impact of code in cooldown on model performance in NL Reasoning, world knowledge, and code benchmarks.



(a) Downstream tasks (b) Generative win-rates

Figure 6: **Impact of code data in pre-training cooldown**: Including code data in the cooldown phase improves downstream relative to cooldown with no code. All cooldown variants benefit for downstream tasks and especially generative quality. We find the largest gains from cooldown with code, with the highest win-rates of 52.3 % over a model with no cooldown.

Across tasks, we find that a cooldown with code data is most beneficial with 3.6%, 10.1%, and 20% in NL reasoning, world knowledge, and code relative to the model without cooldown.

In contrast, we find that cooldown without code does not provide any increases for both NL reasoning and Code, while providing a relative improvement of 3.1% in World Knowledge tasks compared to no cooldown, showing the critical role of code data in also cooldown phase of pre-training.

Generative win-rates after cooldown As expected, cooldown has a significant impact on generative performance as measured by win-rates (seen in Figure 6b). This is because we up-weight high-quality data sources in pre-training mix including instruction-style datasets such as Dolly v2 (Conover et al., 2023). Both cooldown variants (`cooldown w/o code`, `cooldown w/ code`) beat `no-cooldown` model by large win-rates (48.2% and 52.3%) as seen in Figure 6b. Comparing the cooldown variants, including code leads an additional 4.1% generative win-rates against `no-cooldown` compared to cooldown without code.

3.6 Comparing Pre-Training Recipes

Considering all our experiments, we summarize our findings and recommend recipes for pre-training with code data. Table 2 (Appendix B) shows the different variants along with pre-training phases.

Best recipe for natural language tasks As seen in Sections 3.1, 3.3, and 3.5, including code in all phases of pre-training provides improvements across all task categories. When looking at the final recipes, we find that `balanced→text` model followed by cooldown that includes code data corresponds to the best overall performance in natural language tasks considering NL reasoning, world knowledge, and generative performance. Notably this model achieves the highest generative win-rates with 37.7% vs 33.7 against `text-only` as shown in Figure 7.

Best recipe for code performance Among complete recipes shown in Table 2, `balanced-only` provides the best performance in code benchmarks. This model achieves 20% relative gain compared to second best `code→text` and 55% relative gain compared to `balanced→text`. However, `balanced-only` falls behind in natural language performance by 2.5% relative difference and 5.0% win-rate difference (vs `text-only`), both compared to `balanced→text`.

Including code in all phases of pre-training is beneficial across our three task categories and generative performance. Our recommendation for the best overall performance is to include a balanced mixture of code and text data during pre-training from scratch (§ 3.3), use a relatively lower code percentage during continual pre-training (§ 3.1), and include code data into cooldown mixture. Further, we recommend including high-quality code data during all phases of pre-training (§ 3.4).

4 Related Work

Understanding the impact of pre-training mixes Several works have studied the effects of data age, quality, toxicity and domain of pre-training data (Longpre et al., 2023; Üstün et al., 2024). Several works have looked at the impact of filtering (Raffel et al., 2020; Rae et al., 2021; Penedo et al., 2023), de-duping (Zhang et al., 2022) and data pruning (Lozhkov et al., 2024; Marion et al., 2023;

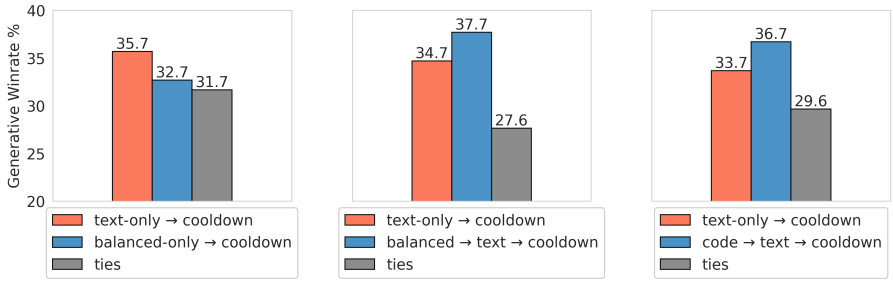


Figure 7: Generative performance as measured by win-rates for variants with full-cooldown.

Chimoto et al., 2024; Boubdir et al., 2023). Furthermore, several works have considered the role of synthetic data at improving performance (Shimabucoro et al., 2024; Dang et al., 2024; Aakanksha et al., 2024) and helping bridge the gap in performance between open weights and proprietary models (Gunasekar et al., 2023; Li et al., 2023b). In contrast to our work which focuses explicitly on understanding the role of code, these studies focus on characteristics of training data as a whole.

Understanding the role of code Including code in the pre-training data mixture, even for models not specifically designed for code, has been a common practice in LLMs pre-training (Dubey et al., 2024; Gemini-Team et al., 2024; Groeneveld et al., 2024). In addition to serving the popular use case in code completion and generation (Chen et al., 2021), previous studies suggest that the addition of code improves the performance of LLMs on various NLP tasks, such as entity linking (Kim et al., 2024) and commonsense reasoning (Madaan et al., 2022b)), mathematical reasoning tasks (Liang et al., 2022; Madaan et al., 2022a; Gao et al., 2023; Shao et al., 2024), and general reasoning capabilities (Muennighoff et al., 2023a; Fu & Khot, 2022; Ma et al., 2023). Muennighoff et al. (2023b) demonstrated Python code data can be used to improve pretraining performance. They focused on a low-resource pre-training regime with limited data and an evaluation set-up limited to perplexity evaluations. Zhang et al. (2024) investigated the impact of code on LLMs' internal reasoning capability across various tasks and model families. They only focus on the effect of code in the supervised fine-tuning stage (SFT) primarily measuring the impact on reasoning. Zhu et al. (2024) report the performance of their DeepSeek-Coder-V2 models on General Natural Language benchmarks. They compare chat and instruct models, and do not investigate different phases of pre-training and properties of code.

To the best of our knowledge, this work is the first study that presents a thorough investigation of the impact of code in pre-training on non-code tasks. Our experiment spans several axes and a exhaustive evaluation suite, with costly ablations at scale including model initialization strategies, different proportions and properties of code data, and model scales.

5 Conclusion

We perform a first-of-its-kind systematic study to answer "*what is the impact of code data used in pre-training on a large variety of downstream tasks beyond code generation*". We focus, not just on code performance but on downstream natural language performance, as well as generative quality using LLM-as-a-judge win-rates. We conduct ablations that look at initialization, proportions of code, quality and properties of code, and role of code in pre-training cooldown. We find across all scales of experiments that code provides critical improvements to performance on non-code tasks. Compared to text-only pre-training, for our best variant, the addition of code results in relative increase of 8.2% in natural language (NL) reasoning, 4.2% in world knowledge, 6.6% improvement in generative win-rates, and a 12x boost in code performance respectively. Further performing cooldown with code, improves 3.6%, 10.1%, and 20% in NL reasoning, world knowledge, and code relative to the model before cooldown and leads 52.3% generative win-rates. Finally, we find that adding a small amount of high-quality synthetic data can have an outsized impact on both NL reasoning (9% relative increase) and code performance (44.9% relative increase).

REFERENCES

Aakanksha, Arash Ahmadian, Beyza Ermis, Seraphina Goldfarb-Tarrant, Julia Kreutzer, Marzieh Fadaee, and Sara Hooker. The multilingual alignment prism: Aligning global and local preferences to reduce harm, 2024. URL https://arxiv.org/abs/2406.18682.

Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Jon Ander Campos, Yi Chern Tan, Kelly Marchisio, Max Bartolo, Sebastian Ruder, Acyr Locatelli, Julia Kreutzer, Nick Frosst, Aidan Gomez, Phil Blunsom, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. Aya 23: Open weight releases to further multilingual progress, 2024. URL https://arxiv.org/abs/2405.15032.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Meriem Boubdir, Edward Kim, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. Which prompts make the difference? data prioritization for efficient human llm evaluation, 2023. URL https://arxiv.org/abs/2310.14424.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Zixiang Chen, Yihe Deng, Yue Wu, Quanquan Gu, and Yuanzhi Li. Towards understanding the mixture-of-experts layer in deep learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=MaYzugDmQV.

Cheng-Han Chiang and Hung yi Lee. Can large language models be an alternative to human evaluations?, 2023.

Everlyn Asiko Chimoto, Jay Gala, Orevaoghene Ahia, Julia Kreutzer, Bruce A. Bassett, and Sara Hooker. Critical learning periods: Leveraging early training dynamics for efficient data pruning, 2024. URL https://arxiv.org/abs/2405.19462.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2174–2184, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1241. URL https://aclanthology.org/D18-1241.

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

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Together Computer. Redpajama: an open dataset for training large language models, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

John Dang, Arash Ahmadian, Kelly Marchisio, Julia Kreutzer, Ahmet Üstün, and Sara Hooker. Rlhf can speak many languages: Unlocking multilingual preference optimization for llms, 2024. URL https://arxiv.org/abs/2407.02552.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse, 2019.

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

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

Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2024. URL https://arxiv.org/abs/2305.14387.

Hao Fu, Yao; Peng and Tushar Khot. How does gpt obtain its ability? tracing emergent abilities of language models to their sources. *Yao Fu's Notion*, Dec 2022. URL https://yaofu.notion.site/b9a57ac0fcf74f30a1ab9e3e36fa1dc1?pvs=25.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire, 2023.

Yifu Gao, Yongquan He, Zhigang Kan, Yi Han, Linbo Qiao, and Dongsheng Li. Learning joint structural and temporal contextualized knowledge embeddings for temporal knowledge graph completion. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 417–430, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.findings-acl.28.

Gemini-Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Güra, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rrustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayana Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault

Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, Alex Kaskasoli, Sébastien M. R. Arnold, Vijay Vasudevan, Shubham Agrawal, Jason Riesa, Dmitry Lepikhin, Richard Tanburn, Srivatsan Srinivasan, Hyeontaek Lim, Sarah Hodkinson, Pranav Shyam, Johan Ferret, Steven Hand, Ankush Garg, Tom Le Paine, Jian Li, Yujia Li, Minh Giang, Alexander Neitz, Zaheer Abbas, Sarah York, Machel Reid, Elizabeth Cole, Aakanksha Chowdhery, Dipanjan Das, Dominika Rogozińska, Vitaliy Nikolaev, Pablo Sprechmann, Zachary Nado, Lukas Zilka, Flavien Prost, Luheng He, Marianne Monteiro, Gaurav Mishra, Chris Welty, Josh Newlan, Dawei Jia, Miltiadis Allamanis, Clara Huiyi Hu, Raoul de Liedekerke, Justin Gilmer, Carl Saroufim, Shruti Rijhwani, Shaobo Hou, Disha Shrivastava, Anirudh Baddepudi, Alex Goldin, Adnan Ozturel, Albin Cassirer, Yunhan Xu, Daniel Sohn, Devendra Sachan, Reinald Kim Amplayo, Craig Swanson, Dessie Petrova, Shashi Narayan, Arthur Guez, Siddhartha Brahma, Jessica Landon, Miteyan Patel, Ruizhe Zhao, Kevin Villela, Luyu Wang, Wenhao Jia, Matthew Rahtz, Mai Giménez, Legg Yeung, James Keeling, Petko Georgiev, Diana Mincu, Boxi Wu, Salem Haykal, Rachel Saputro, Kiran Vodrahalli, James Qin, Zeynep Cankara, Abhanshu Sharma, Nick Fernando, Will Hawkins, Behnam Neyshabur, Solomon Kim, Adrian Hutter, Priyanka Agrawal, Alex Castro-Ros, George van den Driessche, Tao Wang, Fan Yang, Shuo yiin Chang, Paul Komarek, Ross McIlroy, Mario Lučić, Guodong Zhang, Wael Farhan, Michael Sharman, Paul Natsev, Paul Michel, Yamini Bansal, Siyuan Qiao, Kris Cao, Siamak Shakeri, Christina Butterfield, Justin Chung, Paul Kishan Rubenstein, Shivani Agrawal, Arthur Mensch, Kedar Soparkar, Karel Lenc, Timothy Chung, Aedan Pope, Loren Maggiore, Jackie Kay, Priya Jhakra, Shibo Wang, Joshua Maynez, Mary Phuong, Taylor Tobin, Andrea Tacchetti, Maja Trebacz, Kevin Robinson, Yash Katariya, Sebastian Riedel, Paige Bailey, Kefan Xiao, Nimesh Ghelani, Lora Aroyo, Ambrose Slone, Neil Houlsby, Xuehan Xiong, Zhen Yang, Elena Gribovskaya, Jonas Adler, Mateo Wirth, Lisa Lee, Music Li, Thais Kagohara, Jay Pavagadhi, Sophie Bridgers, Anna Bortsova, Sanjay Ghemawat, Zafarali Ahmed, Tianqi Liu, Richard Powell, Vijay Bolina, Mariko Iinuma, Polina Zablotskaia, James Besley, Da-Woon Chung, Timothy Dozat, Ramona Comanescu, Xiance Si, Jeremy Greer, Guolong Su, Martin Polacek, Raphaël Lopez Kaufman, Simon Tokumine, Hexiang Hu, Elena Buchatskaya, Yingjie Miao, Mohamed Elhawaty, Aditya Siddhant, Nenad Tomasev, Jinwei Xing, Christina Greer, Helen Miller, Shereen Ashraf, Aurko Roy, Zizhao Zhang, Ada Ma, Angelos Filos, Milos Besta, Rory Blevins, Ted Klimenko, Chih-Kuan Yeh, Soravit Changpinyo, Jiaqi Mu, Oscar Chang, Mantas Pajarskas, Carrie Muir, Vered Cohen, Charline Le Lan, Krishna Haridasan, Amit Marathe, Steven Hansen, Sholto Douglas, Rajkumar Samuel, Mingqiu Wang, Sophia Austin, Chang Lan, Jiepu Jiang, Justin Chiu, Jaime Alonso Lorenzo, Lars Lowe Sjösund, Sébastien Cevey, Zach Gleicher, Thi Avrahami, Anudhyan Boral, Hansa Srinivasan, Vittorio Selo, Rhys May, Konstantinos Aisopos, Léonard Hussenot, Livio Baldini Soares, Kate Baumli, Michael B. Chang, Adrià Recasens, Ben Caine, Alexander Pritzel, Filip Pavetic, Fabio Pardo, Anita Gergely, Justin Frye, Vinay Ramasesh, Dan Horgan, Kartikeya Badola, Nora Kassner, Subhrajit Roy, Ethan Dyer, Víctor Campos Campos, Alex Tomala, Yunhao Tang, Dalia El Badawy, Elspeth White, Basil Mustafa, Oran Lang, Abhishek Jindal, Sharad Vikram, Zhitao Gong, Sergi Caelles, Ross Hemsley, Gregory Thornton, Fangxiaoyu Feng, Wojciech Stokowiec, Ce Zheng, Phoebe Thacker, Çağlar Ünlü, Zhishuai Zhang, Mohammad Saleh, James Svensson, Max Bileschi, Piyush Patil, Ankesh Anand, Roman Ring, Katerina Tsihlas, Arpi Vezer, Marco Selvi, Toby Shevlane, Mikel Rodriguez, Tom Kwiatkowski, Samira Daruki, Keran Rong, Allan Dafoe, Nicholas FitzGerald, Keren Gu-Lemberg, Mina Khan, Lisa Anne Hendricks, Marie Pellat, Vladimir Feinberg, James Cobon-Kerr, Tara Sainath, Maribeth Rauh, Sayed Hadi Hashemi, Richard Ives, Yana Hasson, Eric Noland, Yuan Cao, Nathan Byrd, Le Hou, Qingze Wang, Thibault Sottiaux, Michela Paganini, Jean-Baptiste Lespiau, Alexandre Moufarek, Samer Hassan, Kaushik Shivakumar, Joost van Amersfoort, Amol Mandhane, Pratik Joshi, Anirudh Goyal, Matthew Tung, Andrew Brock, Hannah Sheahan, Vedant Misra, Cheng Li, Nemanja Rakićević, Mostafa Dehghani, Fangyu Liu, Sid Mittal, Junhyuk Oh, Seb Noury, Eren Sezener, Fantine Huot, Matthew Lamm, Nicola De Cao, Charlie Chen, Sidharth Mudgal, Romina Stella, Kevin Brooks, Gautam Vasudevan, Chenxi Liu, Mainak Chain, Nivedita Melinkeri, Aaron Cohen, Venus Wang, Kristie Seymore, Sergey Zubkov, Rahul Goel, Summer Yue, Sai Krishnakumaran, Brian Albert, Nate Hurley, Motoki Sano, Anhad Mohananey, Jonah Joughin, Egor Filonov, Tomasz Kepa, Yomna Eldawy, Jiawern Lim, Rahul Rishi, Shirin Badiezadegan, Taylor Bos, Jerry Chang, Sanil Jain, Sri Gayatri Sundara Padmanabhan, Subha Puttagunta, Kalpesh Krishna, Leslie Baker, Norbert Kalb, Vamsi Bedapudi, Adam Kurzrok, Shuntong Lei, Anthony Yu, Oren Litvin, Xiang Zhou, Zhichun Wu, Sam Sobell, Andrea Siciliano, Alan Papir, Robby Neale, Jonas Bragagnolo, Tej Toor, Tina Chen, Valentin Anklin, Feiran Wang, Richie Feng, Milad Gholami, Kevin Ling, Lijuan

Liu, Jules Walter, Hamid Moghaddam, Arun Kishore, Jakub Adamek, Tyler Mercado, Jonathan Mallinson, Siddhinita Wandekar, Stephen Cagle, Eran Ofek, Guillermo Garrido, Clemens Lombriser, Maksim Mukha, Botu Sun, Hafeezul Rahman Mohammad, Josip Matak, Yadi Qian, Vikas Peswani, Pawel Janus, Quan Yuan, Leif Schelin, Oana David, Ankur Garg, Yifan He, Oleksii Duzhyi, Anton Älgmyr, Timothée Lottaz, Qi Li, Vikas Yadav, Luyao Xu, Alex Chinien, Rakesh Shivanna, Aleksandr Chuklin, Josie Li, Carrie Spadine, Travis Wolfe, Kareem Mohamed, Subhabrata Das, Zihang Dai, Kyle He, Daniel von Dincklage, Shyam Upadhyay, Akanksha Maurya, Luyan Chi, Sebastian Krause, Khalid Salama, Pam G Rabinovitch, Pavan Kumar Reddy M, Aarush Selvan, Mikhail Dektiarev, Golnaz Ghiasi, Erdem Guven, Himanshu Gupta, Boyi Liu, Deepak Sharma, Idan Heimlich Shtacher, Shachi Paul, Oscar Akerlund, François-Xavier Aubet, Terry Huang, Chen Zhu, Eric Zhu, Elico Teixeira, Matthew Fritze, Francesco Bertolini, Liana-Eleonora Marinescu, Martin Bölle, Dominik Paulus, Khyatti Gupta, Tejasi Latkar, Max Chang, Jason Sanders, Roopa Wilson, Xuewei Wu, Yi-Xuan Tan, Lam Nguyen Thiet, Tulsee Doshi, Sid Lall, Swaroop Mishra, Wanming Chen, Thang Luong, Seth Benjamin, Jasmine Lee, Ewa Andrejczuk, Dominik Rabiej, Vipul Ranjan, Krzysztof Styrc, Pengcheng Yin, Jon Simon, Malcolm Rose Harriott, Mudit Bansal, Alexei Robsky, Geoff Bacon, David Greene, Daniil Mirylenka, Chen Zhou, Obaid Sarvana, Abhimanyu Goyal, Samuel Andermatt, Patrick Siegler, Ben Horn, Assaf Israel, Francesco Pongetti, Chih-Wei "Louis" Chen, Marco Selvatici, Pedro Silva, Kathie Wang, Jackson Tolins, Kelvin Guu, Roey Yogev, Xiaochen Cai, Alessandro Agostini, Maulik Shah, Hung Nguyen, Noah Ó Donnaile, Sébastien Pereira, Linda Friso, Adam Stambler, Adam Kurzrok, Chenkai Kuang, Yan Romanikhin, Mark Geller, ZJ Yan, Kane Jang, Cheng-Chun Lee, Wojciech Fica, Eric Malmi, Qijun Tan, Dan Banica, Daniel Balle, Ryan Pham, Yanping Huang, Diana Avram, Hongzhi Shi, Jasjot Singh, Chris Hidey, Niharika Ahuja, Pranab Saxena, Dan Dooley, Srividya Pranavi Potharaju, Eileen O'Neill, Anand Gokulchandran, Ryan Foley, Kai Zhao, Mike Dusenberry, Yuan Liu, Pulkit Mehta, Ragha Kotikalapudi, Chalence Safranek-Shrader, Andrew Goodman, Joshua Kessinger, Eran Globen, Prateek Kolhar, Chris Gorgolewski, Ali Ibrahim, Yang Song, Ali Eichenbaum, Thomas Brovelli, Sahitya Potluri, Preethi Lahoti, Cip Baetu, Ali Ghorbani, Charles Chen, Andy Crawford, Shalini Pal, Mukund Sridhar, Petru Gurita, Asier Mujika, Igor Petrovski, Pierre-Louis Cedoz, Chenmei Li, Shiyuan Chen, Niccolò Dal Santo, Siddharth Goyal, Jitesh Punjabi, Karthik Kappaganthu, Chester Kwak, Pallavi LV, Sarmishta Velury, Himadri Choudhury, Jamie Hall, Premal Shah, Ricardo Figueira, Matt Thomas, Minjie Lu, Ting Zhou, Chintu Kumar, Thomas Jurdi, Sharat Chikkerur, Yenai Ma, Adams Yu, Soo Kwak, Victor Ähdel, Sujeevan Rajayogam, Travis Choma, Fei Liu, Aditya Barua, Colin Ji, Ji Ho Park, Vincent Hellendoorn, Alex Bailey, Taylan Bilal, Huanjie Zhou, Mehrdad Khatir, Charles Sutton, Wojciech Rzadkowski, Fiona Macintosh, Konstantin Shagin, Paul Medina, Chen Liang, Jinjing Zhou, Pararth Shah, Yingying Bi, Attila Dankovics, Shipra Banga, Sabine Lehmann, Marissa Bredesen, Zifan Lin, John Eric Hoffmann, Jonathan Lai, Raynald Chung, Kai Yang, Nihal Balani, Arthur Bražinskas, Andrei Sozanschi, Matthew Hayes, Héctor Fernández Alcalde, Peter Makarov, Will Chen, Antonio Stella, Liselotte Snijders, Michael Mandl, Ante Kärrman, Paweł Nowak, Xinyi Wu, Alex Dyck, Krishnan Vaidyanathan, Raghavender R, Jessica Mallet, Mitch Rudominer, Eric Johnston, Sushil Mittal, Akhil Udathu, Janara Christensen, Vishal Verma, Zach Irving, Andreas Santucci, Gamaleldin Elsayed, Elnaz Davoodi, Marin Georgiev, Ian Tenney, Nan Hua, Geoffrey Cideron, Edouard Leurent, Mahmoud Alnahlawi, Ionut Georgescu, Nan Wei, Ivy Zheng, Dylan Scandinaro, Heinrich Jiang, Jasper Snoek, Mukund Sundararajan, Xuezhi Wang, Zack Ontiveros, Itay Karo, Jeremy Cole, Vinu Rajashekhar, Lara Tumeh, Eyal Ben-David, Rishub Jain, Jonathan Uesato, Romina Datta, Oskar Bunyan, Shimu Wu, John Zhang, Piotr Stanczyk, Ye Zhang, David Steiner, Subhajit Naskar, Michael Azzam, Matthew Johnson, Adam Paszke, Chung-Cheng Chiu, Jaume Sanchez Elias, Afroz Mohiuddin, Faizan Muhammad, Jin Miao, Andrew Lee, Nino Vieillard, Jane Park, Jiageng Zhang, Jeff Stanway, Drew Garmon, Abhijit Karmarkar, Zhe Dong, Jong Lee, Aviral Kumar, Luowei Zhou, Jonathan Evens, William Isaac, Geoffrey Irving, Edward Loper, Michael Fink, Isha Arkatkar, Nanxin Chen, Izhak Shafran, Ivan Petrychenko, Zhe Chen, Johnson Jia, Anselm Levskaya, Zhenkai Zhu, Peter Grabowski, Yu Mao, Alberto Magni, Kaisheng Yao, Javier Snaider, Norman Casagrande, Evan Palmer, Paul Suganthan, Alfonso Castaño, Irene Giannoumis, Wooyeol Kim, Mikołaj Rybiński, Ashwin Sreevatsa, Jennifer Prendki, David Soergel, Adrian Goedeckemeyer, Willi Gierke, Mohsen Jafari, Meenu Gaba, Jeremy Wiesner, Diana Gage Wright, Yawen Wei, Harsha Vashisht, Yana Kulizhskaya, Jay Hoover, Maigo Le, Lu Li, Chimezie Iwuanyanwu, Lu Liu, Kevin Ramirez, Andrey Khorlin, Albert Cui, Tian LIN, Marcus Wu, Ricardo Aguilar, Keith Pallo, Abhishek Chakladar, Ginger Perng, Elena Allica Abellan, Mingyang Zhang, Ishita Dasgupta, Nate Kushman, Ivo Penchev, Alena Repina, Xihui Wu, Tom

van der Weide, Priya Ponnapalli, Caroline Kaplan, Jiri Simsa, Shuangfeng Li, Olivier Dousse, Fan Yang, Jeff Piper, Nathan Ie, Rama Pasumarthi, Nathan Lintz, Anitha Vijayakumar, Daniel Andor, Pedro Valenzuela, Minnie Lui, Cosmin Paduraru, Daiyi Peng, Katherine Lee, Shuyuan Zhang, Somer Greene, Duc Dung Nguyen, Paula Kurylowicz, Cassidy Hardin, Lucas Dixon, Lili Janzer, Kiam Choo, Ziqiang Feng, Biao Zhang, Achintya Singhal, Dayou Du, Dan McKinnon, Natasha Antropova, Tolga Bolukbasi, Orgad Keller, David Reid, Daniel Finchelstein, Maria Abi Raad, Remi Crocker, Peter Hawkins, Robert Dadashi, Colin Gaffney, Ken Franko, Anna Bulanova, Rémi Leblond, Shirley Chung, Harry Askham, Luis C. Cobo, Kelvin Xu, Felix Fischer, Jun Xu, Christina Sorokin, Chris Alberti, Chu-Cheng Lin, Colin Evans, Alek Dimitriev, Hannah Forbes, Dylan Banarse, Zora Tung, Mark Omernick, Colton Bishop, Rachel Sterneck, Rohan Jain, Jiawei Xia, Ehsan Amid, Francesco Piccinno, Xingyu Wang, Praseem Banzal, Daniel J. Mankowitz, Alex Polozov, Victoria Krakovna, Sasha Brown, MohammadHossein Bateni, Dennis Duan, Vlad Firoiu, Meghana Thotakuri, Tom Natan, Matthieu Geist, Ser tan Girgin, Hui Li, Jiayu Ye, Ofir Roval, Reiko Tojo, Michael Kwong, James Lee-Thorp, Christopher Yew, Danila Sinopalnikov, Sabela Ramos, John Mellor, Abhishek Sharma, Kathy Wu, David Miller, Nicolas Sonnerat, Denis Vnukov, Rory Greig, Jennifer Beattie, Emily Caveness, Libin Bai, Julian Eisenschlos, Alex Korchemniy, Tomy Tsai, Mimi Jasarevic, Weize Kong, Phuong Dao, Zeyu Zheng, Frederick Liu, Fan Yang, Rui Zhu, Tian Huey Teh, Jason Sanmiya, Evgeny Gladchenko, Nejc Trdin, Daniel Toyama, Evan Rosen, Sasan Tavakkol, Linting Xue, Chen Elkind, Oliver Woodman, John Carpenter, George Papamakarios, Rupert Kemp, Sushant Kafle, Tanya Grunina, Rishika Sinha, Alice Talbert, Diane Wu, Denese Owusu-Afriyie, Cosmo Du, Chloe Thornton, Jordi Pont-Tuset, Pradyumna Narayana, Jing Li, Saaber Fatehi, John Wieting, Omar Ajmeri, Benigno Uria, Yeongil Ko, Laura Knight, Amélie Héliou, Ning Niu, Shane Gu, Chenxi Pang, Yeqing Li, Nir Levine, Ariel Stolovich, Rebeca Santamaria-Fernandez, Sonam Goenka, Wenny Yustalim, Robin Strudel, Ali Elqursh, Charlie Deck, Hyo Lee, Zonglin Li, Kyle Levin, Raphael Hoffmann, Dan Holtmann-Rice, Olivier Bachem, Sho Arora, Christy Koh, Soheil Hassas Yeganeh, Siim Põder, Mukarram Tariq, Yanhua Sun, Lucian Ionita, Mojtaba Seyedhosseini, Pouya Tafti, Zhiyu Liu, Anmol Gulati, Jasmine Liu, Xinyu Ye, Bart Chrzaszcz, Lily Wang, Nikhil Sethi, Tianrun Li, Ben Brown, Shreya Singh, Wei Fan, Aaron Parisi, Joe Stanton, Vinod Koverkathu, Christopher A. Choquette-Choo, Yunjie Li, TJ Lu, Abe Ittycheriah, Prakash Shroff, Mani Varadarajan, Sanaz Bahargam, Rob Willoughby, David Gaddy, Guillaume Desjardins, Marco Cornero, Brona Robenek, Bhavishya Mittal, Ben Albrecht, Ashish Shenoy, Fedor Moiseev, Henrik Jacobsson, Alireza Ghaffarkhah, Morgane Rivière, Alanna Walton, Clément Crepy, Alicia Parrish, Zongwei Zhou, Clement Farabet, Carey Radebaugh, Praveen Srinivasan, Claudia van der Salm, Andreas Fidjeland, Salvatore Scellato, Eri Latorre-Chimoto, Hanna Klimczak-Plucińska, David Bridson, Dario de Cesare, Tom Hudson, Piermaria Mendolicchio, Lexi Walker, Alex Morris, Matthew Mauger, Alexey Guseynov, Alison Reid, Seth Odoom, Lucia Loher, Victor Cotruta, Madhavi Yenugula, Dominik Grewe, Anastasia Petrushkina, Tom Duerig, Antonio Sanchez, Steve Yadlowsky, Amy Shen, Amir Globerson, Lynette Webb, Sahil Dua, Dong Li, Surya Bhupatiraju, Dan Hurt, Haroon Qureshi, Ananth Agarwal, Tomer Shani, Matan Eyal, Anuj Khare, Shreyas Rammohan Belle, Lei Wang, Chetan Tekur, Mihir Sanjay Kale, Jinliang Wei, Ruoxin Sang, Brennan Saeta, Tyler Liechty, Yi Sun, Yao Zhao, Stephan Lee, Pandu Nayak, Doug Fritz, Manish Reddy Vuyyuru, John Aslanides, Nidhi Vyas, Martin Wicke, Xiao Ma, Evgenii Eltyshev, Nina Martin, Hardie Cate, James Manyika, Keyvan Amiri, Yelin Kim, Xi Xiong, Kai Kang, Florian Luisier, Nilesh Tripuraneni, David Madras, Mandy Guo, Austin Waters, Oliver Wang, Joshua Ainslie, Jason Baldridge, Han Zhang, Garima Pruthi, Jakob Bauer, Feng Yang, Riham Mansour, Jason Gelman, Yang Xu, George Polovets, Ji Liu, Honglong Cai, Warren Chen, XiangHai Sheng, Emily Xue, Sherjil Ozair, Christof Angermueller, Xiaowei Li, Anoop Sinha, Weiren Wang, Julia Wiesinger, Emmanouil Koukoumidis, Yuan Tian, Anand Iyer, Madhu Gurumurthy, Mark Goldenson, Parashar Shah, MK Blake, Hongkun Yu, Anthony Urbanowicz, Jennimaria Palomaki, Chrisantha Fernando, Ken Durden, Harsh Mehta, Nikola Momchev, Elahe Rahimtoroghi, Maria Georgaki, Amit Raul, Sebastian Ruder, Morgan Redshaw, Jinhyuk Lee, Denny Zhou, Komal Jalan, Dinghua Li, Blake Hechtman, Parker Schuh, Milad Nasr, Kieran Milan, Vladimir Mikulik, Juliana Franco, Tim Green, Nam Nguyen, Joe Kelley, Aroma Mahendru, Andrea Hu, Joshua Howland, Ben Vargas, Jeffrey Hui, Kshitij Bansal, Vikram Rao, Rakesh Ghiya, Emma Wang, Ke Ye, Jean Michel Sarr, Melanie Moranski Preston, Madeleine Elish, Steve Li, Aakash Kaku, Jigar Gupta, Ice Pasupat, Da-Cheng Juan, Milan Someswar, Tejvi M., Xinyun Chen, Aida Amini, Alex Fabrikant, Eric Chu, Xuanyi Dong, Amruta Muthal, Senaka Buthpitiya, Sarthak Jauhari, Nan Hua, Urvashi Khandelwal, Ayal Hitron, Jie Ren, Larissa Rinaldi, Shahar Drath, Avigail Dabush,

Nan-Jiang Jiang, Harshal Godhia, Uli Sachs, Anthony Chen, Yicheng Fan, Hagai Taitelbaum, Hila Noga, Zhuyun Dai, James Wang, Chen Liang, Jenny Hamer, Chun-Sung Ferng, Chenel Elkind, Aviel Atias, Paulina Lee, Vít Listík, Mathias Carlen, Jan van de Kerkhof, Marcin Pikus, Krunoslav Zaher, Paul Müller, Sasha Zykova, Richard Stefanec, Vitaly Gatsko, Christoph Hirnschall, Ashwin Sethi, Xingyu Federico Xu, Chetan Ahuja, Beth Tsai, Anca Stefanoiu, Bo Feng, Keshav Dhandhania, Manish Katyal, Akshay Gupta, Atharva Parulekar, Divya Pitta, Jing Zhao, Vivaan Bhatia, Yashodha Bhavnani, Omar Alhadlaq, Xiaolin Li, Peter Danenberg, Dennis Tu, Alex Pine, Vera Filippova, Abhipso Ghosh, Ben Limonchik, Bhargava Urala, Chaitanya Krishna Lanka, Derik Clive, Yi Sun, Edward Li, Hao Wu, Kevin Hongtongsak, Ianna Li, Kalind Thakkar, Kuanysh Omarov, Kushal Majmundar, Michael Alverson, Michael Kucharski, Mohak Patel, Mudit Jain, Maksim Zabelin, Paolo Pelagatti, Rohan Kohli, Saurabh Kumar, Joseph Kim, Swetha Sankar, Vineet Shah, Lakshmi Ramachandruni, Xiangkai Zeng, Ben Bariach, Laura Weidinger, Amar Subramanya, Sissie Hsiao, Demis Hassabis, Koray Kavukcuoglu, Adam Sadovsky, Quoc Le, Trevor Strohman, Yonghui Wu, Slav Petrov, Jeffrey Dean, and Oriol Vinyals. Gemini: A family of highly capable multimodal models, 2024.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323, 2011.

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

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. *arXiv preprint arXiv:2212.12017*, 2022.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147.

Norman P. Jouppi, Cliff Young, Nishant Patil, David A. Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon. In-Datacenter Performance Analysis of a Tensor Processing Unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture, ISCA*, 2017.

Najoung Kim, Sebastian Schuster, and Shubham Toshniwal. Code pretraining improves entity tracking abilities of language models. *arXiv preprint arXiv:2405.21068*, 2024.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. The stack: 3 tb of permissively licensed source code. *Preprint*, 2022.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019. doi: 10.1162/tacl_a_00276. URL https://doi.org/10.1162/tacl_a_00276.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1612. URL https://www.aclweb.org/anthology/P19-1612.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023a.

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

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment, 2023.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. *arXiv*, abs/2305.13169, 2023.

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

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu, May 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu.

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

Aman Madaan, Dheeraj Rajagopal, Niket Tandon, Yiming Yang, and Antoine Bosselut. Conditional set generation using seq2seq models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 4874–4896, Abu Dhabi, United Arab Emirates, December 2022a. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.324.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language models of code are few-shot commonsense learners. *arXiv preprint arXiv:2210.07128*, 2022b.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. When less is more: Investigating data pruning for pretraining llms at scale, 2023. URL https://arxiv.org/abs/2309.04564.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1098. URL https://aclanthology.org/N16-1098.

Niklas Muennighoff, Alexander M Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. *arXiv preprint arXiv:2305.16264*, 2023a.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models, 2023b. URL https://arxiv.org/abs/2305.16264.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. Crosslingual generalization through multitask finetuning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15991–16111, Toronto, Canada, July 2023c. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.891. URL https://aclanthology.org/2023.acl-long.891.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

Jupinder Parmar, Shrimai Prabhumoye, Joseph Jennings, Mostofa Patwary, Sandeep Subramanian, Dan Su, Chen Zhu, Deepak Narayanan, Aastha Jhunjhunwala, Ayush Dattagupta, Vibhu Jawa, Jiwei Liu, Ameya Mahabaleshwarkar, Osvald Nitski, Annika Brundyn, James Maki, Miguel Martinez, Jiaxuan You, John Kamalu, Patrick LeGresley, Denys Fridman, Jared Casper, Ashwath Aithal, Oleksii Kuchaiev, Mohammad Shoeybi, Jonathan Cohen, and Bryan Catanzaro. Nemotron-4 15b technical report, 2024. URL https://arxiv.org/abs/2402.16819.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis & insights from training gopher, 2022. URL https://arxiv.org/abs/2112.11446.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, abs/1910.10683, 2020.

Yasaman Razeghi, Hamish Ivison, Sameer Singh, and Yanai Elazar. Backtracking mathematical reasoning of language models to the pretraining data. In *The Second Tiny Papers Track at ICLR 2024*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *arXiv*, abs/1904.09728, 2019.

Minjoon Seo, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. Phrase-indexed question answering: A new challenge for scalable document comprehension. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 559–564, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1052. URL https://aclanthology.org/D18-1052.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Luísa Shimabucoro, Sebastian Ruder, Julia Kreutzer, Marzieh Fadaee, and Sara Hooker. Llm see, llm do: Guiding data generation to target non-differentiable objectives, 2024. URL https://arxiv.org/abs/2407.01490.

Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, Mike Zhang, Ramith Hettiarachchi, Joseph Wilson, Marina Machado, Luisa Souza Moura, Dominik Krzemiński, Hakimeh Fadaei, Irem Ergün, Ifeoma Okoh, Aisha Alaagib, Oshan Mudannayake, Zaid Alyafeai, Vu Minh Chien, Sebastian Ruder, Surya Guthikonda, Emad A. Alghamdi, Sebastian Gehrmann, Niklas Muennighoff, Max Bartolo, Julia Kreutzer, Ahmet Üstün, Marzieh Fadaee, and Sara Hooker. Aya dataset: An open-access collection for multilingual instruction tuning. *arXiv preprint arXiv:2402.06619*, 2024.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama, 2023. URL https://huggingface.co/datasets/cerebras/SlimPajama-627B.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *arXiv*, abs/2307.09288, 2023.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems, 2020. URL https://arxiv.org/abs/1905.00537.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13484–13508, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.acl-long.754.

Johannes Welbl, Nelson F Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. pp. 94–106, September 2017. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413.

Wikimedia. Wikimedia downloads. URL https://dumps.wikimedia.org.

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 Haji-Hosseini, 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. Bloom: A 176b-parameter open-access multilingual language model, 2023. URL https://arxiv.org/abs/2211.05100.

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

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

Xinlu Zhang, Zhiyu Zoey Chen, Xi Ye, Xianjun Yang, Lichang Chen, William Yang Wang, and Linda Ruth Petzold. Unveiling the impact of coding data instruction fine-tuning on large language models reasoning. *arXiv preprint arXiv:2405.20535*, 2024.

Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*, 2024.

Ahmet Üstün, Viraat Aryabumi, Zheng-Xin Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. Aya model: An instruction finetuned open-access multilingual language model, 2024.

ETHICS STATEMENT AND LIMITATIONS

While we systematically study the impact of code data on downstream natural language tasks, we do not study its impact on safety and bias. Additionally, given the nature of pre-training and the number of ablations we have conducted we were limited by the scale of larger model sizes due to prohibitive compute costs.

REPRODUCIBILITY

We provide details about our data mixture (Section 2.1), data filtering (Appendix C.1, C.2, C.3), evaluation (Section 2.2, Appendix A) and training (Section 2.3) setups. We believe these details provide a clear picture on how to obtain our data setup, model ablations and evaluation results.

A EVALUATION DETAILS

We briefly describe the details of our evaluation benchmarks and the composite datasets used for each category below:

1. **World knowledge.** These benchmarks aim to measure world knowledge, testing knowledge memorization, retrieval, and question answering capability given context. We include Natural Questions Open (Kwiatkowski et al., 2019), and TriviaQA (Joshi et al., 2017) as the datasets. We report the average exact match scores for both these benchmarks.
2. **Natural language reasoning.** The Natural language (NL) reasoning suite consists of 11 benchmarks that involve natural language based reasoning such as Question Answering (Clark et al., 2019; Seo et al., 2018; Welbl et al., 2017; Sap et al., 2019; Choi et al., 2018), natural language inference (NLI) (Wang et al., 2020; de Marneffe et al., 2019; Wang et al., 2020), sentence completion (Mostafazadeh et al., 2016; Zellers et al., 2019), co-reference resolution (Sakaguchi et al., 2019) and general intelligence (Clark et al., 2018). We include a full list of the constituent benchmarks in Table 1. We report the average accuracy scores across all benchmarks.
3. **Code.** While our main focus is general performance, we also want to measure any changes to code generation performance. For code benchmarks, we focus on the function completion task. We evaluate on HumanEval-Python (Chen et al., 2022) and MBPP (Austin et al., 2021). We report the average `pass@1` scores of these benchmarks.

B SUMMARY RESULTS FOR PRE-TRAINING RECIPES

Summary results are shown in Table 2.

C CODE-DATASETS FILTERING

C.1 QUALITY FILTERS

In addition to the deduplication and quality filtering applied on the GitHub scrapes by Starcoder for The Stack dataset (Li et al., 2023a), we apply filters to remove documents with greater than 1000 float numbers, with instances of the string `0x`, that are lists of top-level domains, and with 'generated by' in the first 400 characters

C.2 PROGRAMMING LANGUAGES PRESENT IN WEB-BASED CODE DATASET

Programming languages included in our version of The Stack dataset are present in Table 3

C.3 MARKUP-STYLE PROGRAMMING LANGUAGES PRESENT IN WEB-BASED CODE DATASET

Markup-style languages included in our version of The Stack dataset are in Table 4

Task	Dataset	Metric	
WORLD KNOWLEDGE TASKS			
Question Answering	TriviaQA (Joshi et al., 2017)	0-shot	Acc.
	NaturalQuestionsOpen (Lee et al., 2019)	0-shot	Acc.
NATURAL LANGUAGE REASONING			
Question Answering	BoolQ (Clark et al., 2019)	0-shot	Acc.
	PiQA (Seo et al., 2018)	0-shot	Acc.
	SciQ (Welbl et al., 2017)	0-shot	Acc.
	SocialIQA (Sap et al., 2019)	0-shot	Acc.
	QUAC (Choi et al., 2018)	0-shot	Acc.
Natural Language Inference	SuperGLUE-CB (Wang et al., 2020; de Marneffe et al., 2019)	0-shot	Acc.
	SuperGLUE-COPA (Wang et al., 2020)	0-shot	Acc.
Sentence Completion	StoryCloze (Mostafazadeh et al., 2016)	0-shot	Acc.
	HellaSwag (Zellers et al., 2019)	0-shot	Acc.
Coreference Resolution	Winogrande (Sakaguchi et al., 2019)	0-shot	Acc.
General Intelligence	ARC-Easy (Clark et al., 2018)	0-shot	Acc.
TEXT GENERATION			
Open-Ended Generation	Dolly-200 (English) (Singh et al., 2024)	0-shot	win-rate
CODE GENERATION			
Function completion	HumanEval (Chen et al., 2021)	0-shot	pass@1
	MBPP (Austin et al., 2021)	0-shot	pass@1

Table 1: **Datasets considered for evaluation**: We conduct extensive evaluations across benchmarks detailed above. These provide valuable proxies for performance in natural language reasoning, world knowledge, open ended text generation, and code generation tasks.

Model Variant	Recipe	Token Count		Natural Language			Code	Total Avg.
		Text	Code	Reason.	Know.	Avg.		
TEXT-ONLY	Pre-training	400B	-	49.0	9.5	29.2	0.4	19.6
	Cooldown	+32B	+8B	54.1	11.1	32.6	4.4	23.2
BALANCED-ONLY	Pre-training	200B	200B	51.8	8.1	30.0	9.0	23.0
	Cooldown	+32B	+8B	53.2	11.1	32.1	8.4	24.2
BALANCED → TEXT	Pre-training Init.	100B	100B	52.0	7.4	29.6	7.8	22.4
	Continue Pre-train.	+180B	+20B	53.0	9.9	31.5	4.8	22.6
	Cooldown	+32B	+8B	54.9	10.9	32.9	5.8	23.9
CODE → TEXT	Pre-training Init.	-	200B	44.7	1.5	23.1	15.5	20.6
	Continue Pre-train.	+180B	+20B	53.3	9.5	31.4	4.1	22.3
	Cooldown	+32B	+8B	52.1	10.3	31.2	7.5	23.3

Table 2: **Model variants with the corresponding pre-training recipes**: Pre-training recipes include initial pre-training, continued pre-training, and cooldown phases. Balanced→Text achieves the best NL performance while Balanced-only performs significantly better in code generation.

D LLM JUDGE PROMPT AND PREAMBLE FOR WIN-RATES

Preamble

```
You are a helpful following assistant whose goal is to select the
preferred
(least wrong) output for a given instruction.
```

Prompt

```
Which of the following answers is the best one for the given
instruction.
A good answer should follow these rules:
1) It should have correct reasoning,
2) It should answer the request in the instruction,
3) It should be factually correct and semantically comprehensible,
4) It should be grammatically correct and fluent.

Instruction:  instruction
```

Language Name	Proportion of total code documents
java	15.54
javascript	15.29
php	12.46
python	9.60
c-sharp	8.30
typescript	7.92
c	6.63
cpp	4.91
go	3.49
ruby	2.69
shell	1.82
kotlin	1.76
Swift	1.52
Vue	1.48
rust	1.00
scala	0.94
JSX	0.83
sql	0.74
dart	0.72
makefile	0.53
lua	0.47
haskell	0.45
smalltalk	0.43
tex	0.37
clojure	0.10

Table 3: Programming languages included in our version of The Stack dataset

Language Name	Proportion of total code documents
markdown	54.23
yaml	10.77
json	9.97
html	8.57
css	6.86
SCSS	5.84
restructuredtext	2.26
TOML	1.25
rmarkdown	0.02
Sass	0.22

Table 4: Markup-style languages included in our version of The Stack dataset

```
Answer (A): completion_a

Answer (B): completion_b

FIRST provide a concise comparison of the two answers which
explains
which answer you prefer and why.
SECOND, on a new line, state exactly one of
'Preferred:  Answer (A)' or 'Preferred:  Answer (B)' to indicate
your choice
of preferred response.

Your response should use the format:
Comparison:  <concise comparison and explanation>
Preferred:  <'Answer (A)' or 'Answer (B)'>
```

E GENERATIVE WIN-RATES FOR IMPACT OF INITIALIZATION

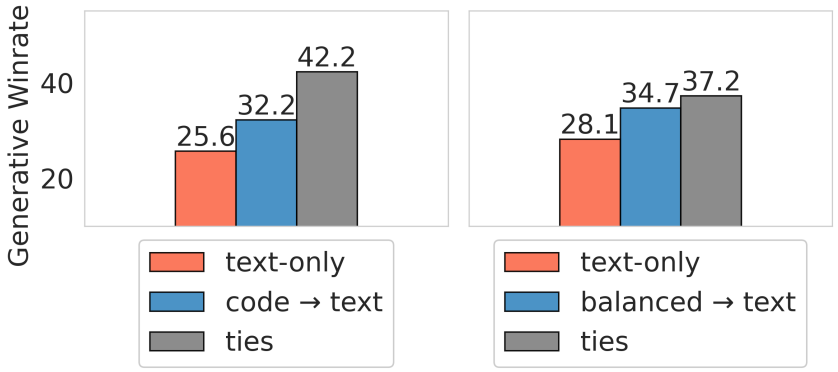


Figure 8: **Impact of initialization on generative quality as judged by LLM-as-a-judge.**

F EVALUATION OF 470M COOLDOWN MODELS ON GSM8K

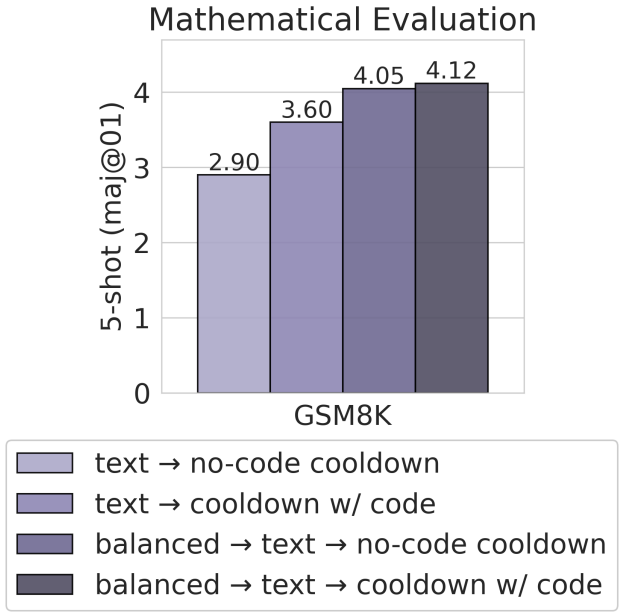


Figure 9: **Evaluation of 470M cooldown models on GSM8K** Including code in any stage of the pre-training improves performance compared to the model where no code has been seen in any of the training stages: pre-training, continual pre-training and cooldown. The most performant model in this comparison has seen code in all stages including cooldown where it leads a significant improvement (from 2.9 to 4.12, +42% relative gain).