Smoky Quartz: A Transparent Bilingual Large Language Model

Quzhe Huang[*], **Chen Zhang**[*], **Mingxu Tao**[*],
Jiuheng Lin, **Songfang Huang**, **Yansong Feng**
Peking University
{zhangch,thomastao,huangquzhe,czb-peking}@pku.edu.cn
linjiuheng@stu.pku.edu.cn
✉ fengyansong@pku.edu.cn

Abstract

Recent advancements in large-scale models have been remarkable; however, a significant challenge persists in their lack of transparency, particularly concerning the training data utilized. This obscurity raises concerns about potential data leaks and forms a barrier to comprehensively understanding the models' capabilities and limitations, as it remains uncertain whether these capabilities are attributed to specific datasets used during training. Addressing this gap, our study introduces Smoky Quartz, a transparent bilingual (Chinese-English) model. We meticulously curated 1.1 trillion tokens of bilingual data from open-source corpora and trained a 7B model from scratch. The performance of our model on various downstream tasks is on par with other open-source models trained on similar data volumes, positioning Smoky Quartz as an invaluable asset for in-depth research and analysis. In a bid to further aid research in long-text applications, we also release a checkpoint capable of handling 128K context lengths, whose context length is expanded via interpolation. All the data and training details are disclosed and we anticipate that Smoky Quartz could contribute to the community's understanding of the relation between LLMs's performance and training data.

1 Introduction

In the past year, significant advancements have been made in the field of large language models (Touvron et al., 2023; OpenAI, 2023; GeminiTeam, 2023), but a persistent concern has been the lack of transparency, particularly regarding the training data (Bommasani et al., 2023). Initial models like ChatGPT were entirely opaque, leaving users in the dark about their inner workings (OpenAI, 2022). Subsequent open-source models like LLaMA have somewhat alleviated this issue (Touvron et al., 2023). Currently, open-source models like Mistral (Jiang et al., 2023), Baichuan (Yang et al., 2023), and DeepSeek (DeepSeek, 2023) are nearing the capabilities of closed-source counterparts like GPT-4 (OpenAI, 2023), marking a significant success for the open-source community. However, we observe that these open-source models, despite revealing their architecture and weights, fall short in disclosing their training data, a crucial component of large models.

The lack of data transparency hinders the analysis and understanding of models. Users are concerned about data leakage and are unable to accurately measure a model's capabilities through downstream tasks, as the model might have been pre-trained on corresponding training or even test data (Li and Flanigan, 2023). For instance, Skywork's report (Wei et al., 2023) indicated that some models might have used downstream task data during training, resulting in anomalous performances on general evaluation tasks like MMLU. A more significant issue is the difficulty in pinpointing the source of a model's capabilities or flaws. When a model solves a challenging problem, we cannot ascertain whether it has encountered similar issues during training or has developed specific capabilities to address the problem. Similarly, when a model exhibits abnormal behavior, it's challenging to determine whether this is due to bad cases in the training data. In essence, without access to the training data, models remain somewhat opaque to us, limiting our understanding of them.

To further unveil this *black box* and assist the open-source community in better comprehending the behavior and origins of large model capabilities, we propose a transparent 7B bilingual model, Smoky Quartz. We utilized a mainstream decoder-only framework with RoPE (Su et al., 2024) for positional encoding, aligning with the majority of current open-source models. We collected and cleaned about 1.1T tokens of bilingual corpora, almost all of which came from open-source datasets. Less

[*] Equal Contribution.

than 10% of the data was additionally collected by us, and even this portion has open-source alternatives available. We will provide detailed information on the sources and cleaning recipes of the training data in this report. Although trained almost exclusively on open-source data, our model's performance on downstream tasks is comparable to other open-source models like Baichuan2 (Yang et al., 2023) and Skywork (Wei et al., 2023) when trained on a similar volume of data. This indicates the robustness of our training process and the efficacy of our model, making it a suitable candidate for analyzing the sources of model capabilities or defects.

An interesting observation is that our model outperforms existing open-source models in long-context tasks, possibly due to our consistent use of 8K context length from scratch. We have also open-sourced a checkpoint that can accept 128K tokens as input, hoping it will be helpful in better understanding the model's performance on long texts and its relationship with the training data."

Our contributions are as follows:

1. We have released a transparent 7B bilingual model and detailed all the training data, aiding future research in analyzing the sources of large model capabilities or defects from a data perspective.

2. We demonstrated that using nearly only open-source data can achieve comparable results to existing mainstream models under similar data volumes.

3. We provide a detailed process for Chinese data cleaning, assisting subsequent researchers in better data handling.

4. We also released a checkpoint capable of handling 128K context length, facilitating researchers in exploring the origins of long-text capabilities.

2 Pretraining

2.1 Architecture

Considering the outstanding performance of LLaMA 2 (Touvron et al., 2023), we choose the same model architecture as LLaMA to validate our entire pipeline to train a model from scratch. Specifically, SmokyQuartz-7B has 32 layers, with a hidden size of 4096 and 32 attention heads for each layer. Different from LLaMA 2, our model has a vocabulary size of 68K, including English, Chinese, and tokens specifically designed for code. Additionally, our model is trained with a sequence length of 8K, instead of 4K.

2.2 Infrastructure

To train our model, we use a cluster of 8 NVIDIA-A800 nodes, a total of 64 A800-80G SXM GPUs. Our training framework is based on Megatron-LM (Shoeybi et al., 2019), which is designed to support the training of extremely large language models. We chose it because the efficiency is much higher than other frameworks, such as Deepspeed.

We use Flash Attention V2 (Dao, 2023) to optimize the GPU memory usage and accelerate the training speed. We find Flash Attention V2 is very useful when training with long contexts. With the 8K context length, the Flash Attention V2 could bring about 10% improvements in training speed compared with the Flash Attention V1 (Dao et al., 2022).

We adopt pipeline parallel (PP) instead of tensor parallel or sequence parallel, as the pipeline parallel methods introduce minimal communication overhead compared with the other two methods. With PP=4, we achieve the throughput of 3271 tokens per GPU per second and a model flop utilization of 44.0%. The whole training process takes about 70 days.

2.3 Data

The diversity and quality of pre-training data are vital to successful pre-training (Computer, 2023; Penedo et al., 2023). As the first step of training the model, we collect a corpus from a wide range of domains and conduct strict filtering and deduplication to ensure the high quality of the data.

2.3.1 Data Sources

Our pre-training data consists of three parts: English, code, and Chinese. We describe the data sources of these parts respectively. The ratio between English, code and Chinese data is 4.5:0.5:5.

English Data For English, we use the RedPajama corpus (Computer, 2023), whose domain distribution is similar to that of LLaMA including CommonCrawl, C4, Github, Books, Arxiv, Wikipedia, and StackExchange. Considering that RedPajama is already carefully cleaned, we do not apply further treatments to it.

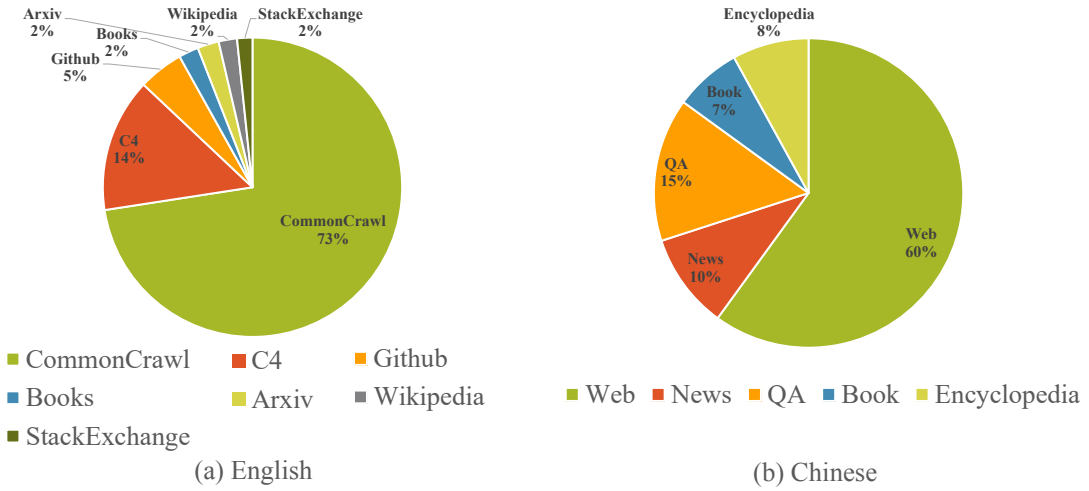

Figure 1: Distribution of the training corpus of Smoky Quartz.

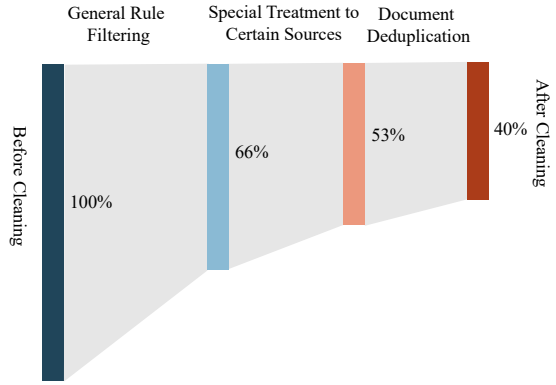

Figure 2: Illustration of the data cleaning procedure.

Code Data For code, we use the StarCoder Dataset (Li et al., 2023b), which contains code in 86 programming languages. We do not use the split of GitHub issues in the dataset. We remove the markers of GitHub stars at the beginning of each file.

Chinese Data For Chinese, we construct a collection of Chinese corpus from multiple public resources. The distribution of our Chinese corpus is shown in Figure 1. Web data makes up 60% of our training corpus, where we use Wudao (Yuan et al., 2021) and the web text split of WanJuan 1.0 (He et al., 2023). The next largest source is question-answering (QA) pairs collected from Chinese QA forums. For news, we mainly use the Chinese News split of WanJuan 1.0. In terms of encyclopedias, we use the Chinese articles in Wikipedia and Baidu Baike. In addition, we use the textbook and exam splits of WanJuan 1.0, together with a small amount of in-house books.

2.3.2 Data Cleaning

To further improve the quality of our pre-training data, we design an effective and efficient pipeline for data cleaning. The pipeline includes three steps: filtering by general rules, special treatments to texts from certain sources, and deduplication of documents. We show the change in data sizes after each cleaning step in Figure 2.

In the first step, we adopt a series of heuristic rules to filter undesired low-quality documents, following Penedo et al. (2023). These rules can comprehensively evaluate the quality of a document from multiple aspects, such as repetition, document length, and symbol-to-word ratio. In this way, we filtered many empty documents, repetitive texts, advertisements, garbled texts, non-Chinese documents, etc.

Afterward, we deal with the documents from certain sources with special treatments. We design specific rules considering the distinct characteristics of the texts from different sources. For example, we fix the format problems in the books, filter answers with a small number of *likes*, and remove the inline advertisements in the web pages.

Finally, we combine multiple deduplication strategies to avoid substantial overlaps between documents. We first remove the documents that are from different sources but share the same URLs. We then adopt both exact and fuzzy deduplication. For exact deduplication, we discard the pages with the same SHA256 hash values. For fuzzy deduplication, we apply the MinHash algorithm (Broder, 1997): for each page, we compute the minhash values and measure their approximate similarities with other pages, removing pairs whose minhash

values are the same in at least one bucket.

2.4 Tokenizer

We tokenize the pre-training data with byte-pair encoding (BPE) algorithm (Sennrich et al., 2015), which is implemented by SentencePiece (Kudo and Richardson, 2018). We train the tokenizer with texts sampled from our pre-training corpus. Considering that the number of tokens might affect the compression rate and the model's representation ability across languages, we evenly distribute the number of Chinese and English tokens. To improve the model performance on math and code problems, we also additionally train a tokenizer with the GitHub and arXiv splits of RedPajama corpus (Computer, 2023). Notably, we then select a subset of high-frequency tokens from the code tokenizer and integrate them into the text tokenizer. Figure 3 illustrates the compression rates under different quantities of merged code tokens. The tokenizer of Smoky Quartz has about 68K tokens, consisting of around 32K English tokens, 32K Chinese tokens, and 4K code tokens.

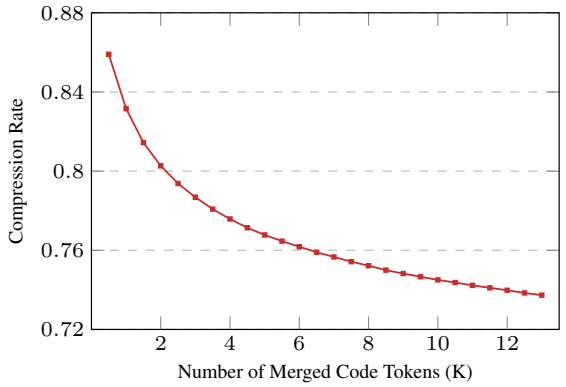

Figure 3: The compression rates for GitHub and arXiv corpus under different quantities of merged code tokens.

To process texts in diverse languages with distinct characteristics, we introduce the following constraint rules when designing the tokenizer:

- We split all numbers into individual digits.
- For Chinese tokens, they should not contain any whitespace characters. For each other token, either it is composed entirely of whitespaces, or it should have at most one whitespace as a prefix.
- We set the maximum length to 8 for Chinese tokens, while it is set to 16 for others.
- When training the tokenizer with plain texts, we remove all tokens containing multiple punctuation characters. However, when training the tokenizer with code data, this constraint is not enforced.

2.5 Hyperparameter

We use the AdamW (Loshchilov and Hutter, 2017) to optimize our model, with β_1 and β_2 are set to 0.9 and 0.95 respectively. The weight decay is set to 0.1 and we clip the grad norm to 1.0. The maximum learning rate is set to 3e-5 at the beginning but we reset it to 2e-5 during the training as we found the learning rate might be too large and the training loss didn't decrease. The batch size is set to 4M tokens. And we use bfloat16 to train our model.

3 Experiments

3.1 Performance on Popular Benchmarks

To assess the capabilities of our model, we conduct evaluations on widely used benchmarks, like MMLU (Hendrycks et al., 2020), CMMLU (Li et al., 2023a), CEVAL (Huang et al., 2023), and BBH (Suzgun et al., 2022).

Table 1 showcases the performance of our model compared to other open-source models, including their performance on these tasks with a comparable amount of training data to ours. From the table, we can draw the following conclusions:

1. When using a similar amount of training data (approximately 1T tokens), the performance of our model is comparable to that of other open-source models, such as Baichuan2-7B (Yang et al., 2023) and Skywork-13B (Wei et al., 2023). This indicates that our training process is sound and that our current model is sufficiently powerful to analyze the relationship between model performance and training data.

2. There are performance gaps between our model and those trained with larger data sets. However, we believe that, like Baichuan-7B and Skywork-13B, our model's performance will significantly improve with increased training, reaching a level comparable to other open-source models.

3.2 Data Leakage Test

A possible issue in the pre-training of LLMs is data leakage. To ensure the fairness and objectivity

	MMLU (5-shot)	CMMLU (5-shot)	CEVAL (5-shot)	BBH (3-shot)
Trained with 1T data				
LLaMA-7B	35.1	26.8	27.1	32.4
Baichuan-7B	42.3	44.0	42.8	32.5
ChatGLM2-6B	47.9	-	51.7	33.7
Baichuan2-7B (1T)	∼48	∼49	∼48	-
Skywork-13B (1T)	∼43	-	∼38	-
Ours-7B-Base	45.2	46.4	44.1	32.3
More Training Data				
Baichuan2-7B	54.2	57.1	54.0	41.6
ChatGLM3-6B	61.4	67.5	69.0	66.1
Skywork-13B	62.1	61.8	60.6	-
Yi-7B	63.2	75.5	72.0	42.8

Table 1: Performance of widely used benchmarks with short input texts. Except for our own, the results of all the other models are reported in their GitHub projects. The results of Baichuan2-7B (1T) and Skywork-13B (1T) are inferred from the figures in their projects.

of the evaluation datasets, we do not specifically construct data to enhance the model for particular tasks. Similar to Wei et al. (2023), we evaluate the language modeling (LM) loss on the samples of MMLU (Hendrycks et al., 2020) and CMMLU (Li et al., 2023a).

The results are shown in Table 2. For MMLU and CMMLU, the losses on the non-test and test splits are almost identical. Moreover, we find that the loss on the benchmark-style data is notably higher than that on our general-domain training corpus. These results indicate that the benchmarks are out-of-domain for our model.

	MMLU	CMMLU
Non-test split	2.12	2.07
Test split	2.15	2.08
General domain	1.95	
Δ_{split}	0.03	0.01
Δ_{domain}	0.17	0.12

Table 2: Average loss of SmokyQuartz-7B on the non-test and test sets of MMLU and CMMLU. Each sample for loss calculation is a concatenation of question and answer. "General domain" denotes the loss on our general-domain training corpus. Δ_{split} denotes the loss difference between the non-test split and the test split of a benchmark. Δ_{domain} denotes the loss difference between the benchmark and our general-domain training corpus.

3.3 Long Context Experiments

To adapt to more complex inputs, recent open-source models have started to accept longer input lengths, such as ChatGLM3 and Yi. To contribute to community research on the correlation between model performance and training data in long-text scenarios, we also release a checkpoint that can handle up to 128K in context length. This model surpasses ChatGLM3 and Yi in the LongBench benchmark and demonstrates commendable performance in a QA task whose input is longer than 100K tokens.

Our model is trained with an 8K context length from the beginning, which is different from many other open-source models that are initially trained with 2K or 4K context lengths and later expanded to 8K or beyond. To help the model process longer contexts, we adopt the Adjusted Base Frequency (ABF) method (Xiong et al., 2023) to further extend the input length of our model. Specifically, we modify the frequency base of the position embedding RoPE from 10,000 to 500,000 and continue training on texts with a context length of 32K tokens. During the continual training process, we adjust the proportion of training data based on the text length, sampling more long texts like books and academic papers. Although the model is only trained with a context length of 32K tokens, we found that our model demonstrates good extrapolation capabilities, which means our model could handle inputs with up to 128K tokens. This finding of good extrapolation of ABF is consistent with other concurrent works (Liu et al., 2023).

	Avg	SQA	MQA	Sum	FS	Code
Base Models						
LLaMA2-7B-4K	17.0	11.9	5.2	0.2	19.8	48.1
Qwen-7B-8K	25.9	14.5	12.6	8.9	30.3	63.2
Baichuan2-7B-4K	26.2	13.3	20.8	9.7	29.3	57.7
Yi-6B-200K	29.3	23.0	13.2	8.5	38.0	63.0
Ours-7B-Base	**40.4**	50.1	40.0	8.4	38.0	65.8
Chat Models						
ChatGLM3-6B-32K	43.1	59.5	42.0	15.7	42.0	56.1
Ours-7B-Chat	**43.8**	54.9	41.8	16.4	40.1	65.9

Table 3: Performance of different open-source models on Longbench-ZH, which are reimplemented by us based on the official code provided by LongBench. SQA, MQA, SUM and FS represent Single-Document QA, Multi-Document QA, Summarization, and Few-Shot respectively. We do not consider the synthetic task because almost all models are unable to handle this task.

As shown in Table 3, after further extend-

	32K-64K	64K-128K
ChatGLM3-6B-32K	2.2	-
Yi-6B-200K	23.0	21.3
Ours-7B-Base	**30.6**	**26.8**

Table 4: Peformance on two extremely long subsets of Narrative-QA whose inputs have 32,000 to 64,000 tokens and 64,000 to 128,000 tokens.

ing the model's available context length, its performance on LongBench improved significantly. SmokyQuartz-128K-Base achieves an average score of 40.4, surpassing Yi-200K, which is also designed for long texts. Since many tasks in LongBench are in a zero-shot format, it may be challenging for a base model to follow the task instructions. Thus, we also train a chat version and we observe that the SmokyQuartz-128K-Chat outperforms ChatGLM3-32K, which is one of the best models in long text tasks.

Although LongBench is widely used for evaluating the capability of handling long texts, we notice that the average length of inputs in LongBench is actually less than 32,000 tokens. To further explore the models' ability to understand extremely long texts, we select two subsets from Narrative QA, whose input length is from [32000, 64000] and [64001, 128000]. Table 4 shows the performance of SmokyQuartz and other models that perform well on LongBench. From the table, we can observe that SmokyQuartz performs best when the input length is longer than 32,000 tokens, while Yi-200K can also achieve good performance on extremely long texts. However, we observe that ChatGLM3-32K fails in this scenario, although it is also continually trained with 32,000 tokens. We speculate that this may be attributed to the better extrapolation capabilities of ABF compared to linear interpolation adopted by ChatGLM3-32K.

4 Conclusion

In this technique report, we provide a transparent bilingual large language model, SmokyQuartz. Our model performs comparably to models trained with an equivalent amount of data in common benchmarks such as MMLU and CMMU. Moreover, in long-text tasks, the performance of our extended model, SmokyQuartz-128K, even surpasses existing open-source models like ChatGLM3-32K and Yi-200K. We have not only made the model's weights publicly available, but more importantly, we have also detailed the data used in training. We hope our work will assist the community in better understanding the remarkable capabilities of large models from a data perspective.

References

Rishi Bommasani, Kevin Klyman, Shayne Longpre, Sayash Kapoor, Nestor Maslej, Betty Xiong, Daniel Zhang, and Percy Liang. 2023. The foundation model transparency index. *arXiv preprint arXiv:2310.12941*.

A.Z. Broder. 1997. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pages 21–29.

Together Computer. 2023. Redpajama: An open source recipe to reproduce llama training dataset.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

DeepSeek. 2023. Deepseek llm: Let there be answers. https://github.com/deepseek-ai/DeepSeek-LLM.

GeminiTeam. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Conghui He, Zhenjiang Jin, Chao Xu, Jiantao Qiu, Bin Wang, Wei Li, Hang Yan, Jiaqi Wang, and Dahua Lin. 2023. Wanjuan: A comprehensive multimodal dataset for advancing english and chinese large models.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, et al. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *arXiv preprint arXiv:2305.08322*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Changmao Li and Jeffrey Flanigan. 2023. Task contamination: Language models may not be few-shot anymore. *arXiv preprint arXiv:2312.16337*.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023a. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2023b. Starcoder: may the source be with you!

Xiaoran Liu, Hang Yan, Shuo Zhang, Chenxin An, Xipeng Qiu, and Dahua Lin. 2023. Scaling laws of rope-based extrapolation. *arXiv preprint arXiv:2310.05209*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

OpenAI. 2022. Chatgpt. `https://chat.openai.com`.

OpenAI. 2023. Gpt-4 technical report.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *Computer Science*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, et al. 2023. Skywork: A more open bilingual foundation model. *arXiv preprint arXiv:2310.19341*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.