Moxin-7B Technical Report

Pu Zhao[1], **Xuan Shen**[1], **Zhenglun Kong**[2], **Yixin Shen**[3], **Sung-En Chang**[1], **Timothy Rupprecht**[1], **Lei Lu**[1], **Enfu Nan**[1], **Changdi Yang**[1], **Yumei He**[4], **Xingchen Xu**[5], **Yong Huang**[6], **Wei Wang**[6], **Yue Chen**[6], **Yong He**[6], **Yanzhi Wang**[1,7]

[1]Northeastern University, [2]Harvard University,
[3]Cornell University, [4]Tulane University,
[5]University of Washington, [6]Futurewei Technologies, [7]AIBAO LLC

Abstract

Recently, Large Language Models (LLMs) have undergone a significant transformation, marked by a rapid rise in both their popularity and capabilities. Leading this evolution are proprietary LLMs like GPT-4 and GPT-o1, which have captured widespread attention in the AI community for their power and versatility. Simultaneously, open-source LLMs, such as LLaMA and Mistral, have made great contributions to the ever-increasing popularity of LLMs due to the ease to customize and deploy the models across various applications. Although LLMs offer unprecedented opportunities for research and innovation, its commercialization has raised concerns about transparency, reproducibility, and safety. Many open LLM models lack the necessary components (such as training code and data) for full understanding and reproducibility, and some use restrictive licenses whilst claiming to be "open-source", which may hinder further innovations on LLMs. To mitigate this issue, we follow the Model Openness Framework (MOF), a ranked classification system that rates machine learning models based on their completeness and openness, following principles of open science, open source, open data, and open access. We present a truly open source LLM Moxin 7B and release pre-training code and configurations, training and fine-tuning data, and intermediate and final checkpoints, aiming to make continuous commitments to fully open-source LLMs. Our model can achieve superior performance in zero-shot evaluation compared with popular 7B models, and perform competitively in few-shot evaluation.

Homepage: *https://github.com/OminiX-ai/OminiX-LLM*

Base model: *https://huggingface.co/OminiX-ai/moxin-7b*

Chat model: *https://huggingface.co/OminiX-ai/moxin-chat-7b*

1 Introduction

The field of natural language processing has witnessed the most exciting discoveries of the last ten years with the emergence of large language models. At the forefront of this evolution are proprietary LLMs such as GPT-4 [1], Mistral [2] and Gemini [3], which have captured the attention of the AI community due to their power and versatility. Meanwhile, the recent emergence of openly accessible yet highly capable LLMs such as LLaMA [4], Falcon [5], and Mistral [2] allow researchers and practitioners to easily obtain, customize, and deploy LLMs in more diverse environments and for more diverse use cases. The trends in the research have people eagerly asking about what's next and some suggest "a general intelligence" is right around the corner.

Despite the growing influence and accessibility of open-source LLMs, a notable trend has been to restrict visibility and access to their training, fine-tuning, and evaluation processes, including crucial components such as their training code and data. This practice limits the ability of the broader AI research community to study, replicate, and innovate upon advanced LLMs. This hampers research by limiting the amount of scientists who can make contributions to the field, especially when compared to the smaller models of the previous ten years in other deep learning fields. To fully harness the power of these models, more must be done to democratize this research by putting it into the hands of more researchers and making the datasets the models train on fully open source. A more transparent approach to sharing not just the final model but also training details and artifacts is crucial for fostering a more inclusive and collaborative research environment.

Generative AI (GAI) offers unprecedented opportunities for research and innovation, but its commercialization has raised concerns about transparency, reproducibility, and safety. Many open GAI models lack the necessary components for full understanding and reproducibility, and some use restrictive licenses whilst claiming to be "open-source". To address these concerns, we follow the Model Openness Framework (MOF), a ranked classification system that rates machine learning models based on their completeness and openness, following principles of open science, open source, open data, and open access. By promoting transparency and reproducibility, the MOF combats "open-washing" practices and establishes completeness and openness as primary criteria alongside the core tenets of responsible AI. Wide adoption of the MOF will foster a more open AI ecosystem, benefiting research, innovation, and adoption of state-of-the-art models. Follwoing MOF, we present an open source LLM Moxin 7B and release pre-training code and configurations, training and fine-tuning data, and intermediate and final checkpoints, aiming to make continuous commitments to fully open-source LLMs. Our base model can achieve superior performance in zero-shot evaluation compared with popular 7B models, and perform competitively in few-shot evaluation. Our chat model can outperform 7B baselines such as Llama2-7B-chat. Our Github link is *https://github.com/OminiX-ai/OminiX-LLM*.

2 Related Work

2.1 Models, Tokenizers, and Training

Models. The most state-of-the-art large language models (LLMs) tend to be larger, approaching, or even surpassing 100 billion parameters [1], [3], [4]. Bringing these technologies into the hands of more people is possible now with the development of smaller model sizes under 20 billion parameters, and even moreso with models around 7 billion parameters [2], [4], [6]–[9]. The LLaMA 3 foundational model set is the only one above with a smaller 7B model [4]. Recent improvements to efficiency have included utilization of MAMBA based architectures as seen in the new Jamba foundational set of language models [8], [9].

Tokenizers. Today many models use Byte-Pair Encoding [10] as implemented by OpenAI in their tiktoken tokenizer [11]. For a wider application to languages that may not handle some tokens (such as spaces) the same way as romantic languages do, options such as SentiencePiece [12] as implemented in XLNet [13]. Hugging Face has a good summary of state-of-the-art tokenizers with examples. [14] Tokenizers are not only limited to a text modality; many of these foundational model sets now include multi-modal models capable of working with documents, videos, pictures, and in some cases audio [15]–[18].

Training. Various training strategems can also be deployed to help smaller models punch above it weightclass. We have seen great success in the application of Mixture of Experts training like with Mixtral [19].

2.2 Data curation methods

To collect large datasets for training LMs [20], researchers typically resort to web crawls, which can contain undesirable content that can be improved via curation. Most data curation efforts focus on methods for improving model performance [20]–[23], including filtering by language [24]–[26], heuristic-based filtering [21], [27], [28], quality filtering [29]–[31], data deduplication [32], [33] and mixing [34]–[36].

2.3 Open-source datasets

As the scale of LMs has increased over the past years [1], [4], [37], [38], the community has curated larger datasets to match. Early works include the C4 dataset with 160 billion (B) tokens and The Pile [28] with 300B tokens. More recently, RefinedWeb [21] contains 600B tokens, Dolma 3 trillion (T) tokens [39], FineWeb 15T tokens [40], and RedPajama-v2 30T tokens [41]. There are also large domain-specific datasets, such as the code-focused StackV2 with 900B tokens [42], as well as high-quality filtered subsets such as FineWeb-Edu with 1.3T tokens [40].

3 Model Training

3.1 Model Architecture

We extend the Mistral model architecture [2]. Mistral 7B can deliver high performance while maintaining an efficient inference speed. It outperforms multiple 7B or larger models across various evaluation benchmarks, such as surpassing the LLaMA 34B model [43] in mathematics and code generation.

Mistral leverages grouped-query attention (GQA) [44], and sliding window attention (SWA) [45]. By reducing the memory requirement during decoding, GQA can allow for higher batch sizes and thus higher throughput. It also achieves significant acceleration for the inference speed, which plays a key role in real-time applications. Besides, SWA can address long sequences effectively without incurring significant computational overhead. These techniques lead to significant improvements for the model performance and efficiency.

Table 1: Parameter setting.

Parameter	Value
n_layers	36
dim	4096
head_dim	128
hidden_dim	14336
n_heads	32
n_kv_heads	8

We extend the Mistral architecture. The original Mistral model has 32 blocks. We extend the model with 36 blocks. GQA divides query heads into a number of groups, each of which shares a single key head and value head. It interpolates between multi-query attention (MQA) and multi-head attention (MHA) in LLMs and strikes a balance between the speed of MQA and the quality of MHA, providing a favorable trade-off. Our model adopts Rolling Buffer Cache which limit the cache size using a rolling buffer cache with a fixed attention span. Thus, the cache memory usage does not become too large for long sequence data.

3.2 Training Data

The data is critical in the LLM pre-training. It needs to address multiple challenges such as handling sensitive data, covering comprehensive knowledge, and achieving higher efficiency with better quality. In this section, we provide more details for preparing textual data in general domain, and coding date related to programming languages.

3.2.1 Text Data

We use a mix of data from SlimPajama [46] and DCLM-BASELINE [36] as our text training data.

LLaMA [47] demonstrates training smaller models on more than 1T tokens. The LLaMA data collection methodology was quickly replicated and released as the RedPajama 1.2T token open-source dataset. After some investigation, it is discovered that some corpora contained a large percentage of duplicates. The deduplication guidelines in RedPajama only operated within each data source, while the duplicates between them are less explored. To improve data quality and training efficiency, SlimPajama was developed to produce a cleaned and extensively deduplicated version of the RedPajama.

To produce SlimPajama, it first removed short, low quality documents from RedPajama. After removing punctuation, space symbols, newlines and tabs, it filtered out documents with less than 200 characters. These documents typically contain only meta data and no useful information. In total this removed 1.86% of documents from RedPajama. Next it removes duplicated data. Training on deduplicated data can improve the training efficiency and reduce memorization with a less chance to generate text memorized from the training data [21], [32], [48]–[50]. CommonCrawl and GitHub

contains more significant duplication [21] with a similar duplication rate in each corpus. To perform deduplication, the MinHashLSH [51] is adopted with a Jaccard similarity threshold of 0.8. Document signatures are constructed on top of pre-processed lower-cased 13-grams. Pre-processing includes removing punctuation, consecutive spaces, newlines and tabs. Deduplication was performed both within and between data sources. In total, by pruning 49.6% of bytes from RedPajama, the 627B token SlimPajama dataset can be obtained.

The DCLM-BASELINE dataset [36] is obtained from CommonCrawl, with key dataset construction steps including text extraction, deduplication, and model-based filtering. DCLM-BASELINE use resiliparse to extract text from CommonCrawl. Web-crawled datasets often contain many duplicate or near-duplicate data. For deduplication, DCLM-BASELINE uses MinHash [52], as part of a suffix array pipeline [32], [53], and near-duplicate Bloom filtering, which modifies an exact document and paragraph deduplication scheme [39]. Recent literature [39], [54], [55] indicates that using learnable models as quality filters leads to downstream improvements. For DCLM-BASELINE and the remaining experiments, DCLM-BASELINE uses fastText OH-2.5 + ELI5 classifier score to keep the top 10% of documents.

3.2.2 Coding Data

Programming is crucial for LLMs to support various downstream tasks, such as code completion from natural language descriptions, documentation generation for individual functions, and auto-completion of code snippets. Furthermore, as code is generally better structured and organized than natural language, training on code data may improve the LLM reasoning capabilities [56]. We use part of the-stack-dedup dataset [57] during the pretraining.

The Stack contains over 6TB of permissively-licensed source code files covering 358 programming languages, which was created for the responsible development of LLMs for Code. It serves for code-generating AI systems to enable the synthesis of programs from natural language descriptions as well as other from code snippets.

To create the Stack, 220.92M active GitHub repository names were collected from the event archives published between 2015 and 2022 on GHArchive. Only 137.36M of these repositories were public and accessible on GitHub. 51.76B files were downloaded from the public repositories on GitHub. 5.28B files were unique. The uncompressed size of all stored files is 92.36TB. Near-deduplication was implemented in the pre-processing pipeline on top of exact deduplication. To find near-duplicates, MinHash with 256 permutations of all documents was computed in linear time. Locality Sensitive Hashing was used to find the clusters of duplicates. Jaccard Similarities were computed inside these clusters to remove any false positives and with a similarity threshold of 0.85. Roughly 40% of permissively licensed files were (near-)duplicates.

3.2.3 Capability Enhancement

Capabilities such as reasoning, mathematical problem-solving, and knowledge memorizing are key abilities expected from large language models. However, in the pre-training process, high-quality capability-related data is sparsely distributed in the entire corpus, which makes it hard for models to be proficient at these mentioned capabilities. Previous works, such as Qwen [6], GLM-130B [58], Nemotron-4 [59], have tried to incorporate instruction-based or high quality data during the pre-training stage to enhance these abilities. In our model, we collect open-source data from huggingface including the training data of various evaluation datasets, such as MMLU [60] and HellaSwag [61]. It is not recommended to train on the training data of evaluation datasets due to data pollution issues and our training is just experimental.

3.3 Training Configuration

The total number of tokens used for pre-training the 7B model is over 1.5T, and the pre-training process consists of three phases. In the first phase, we use pre-training corpora with the context length of 2k. In the second phase, we use pre-training corpora with the context length of 4k. In the third phase, we utilize the capability-specific enhancement data. We provide the model performance with only the first two phases and also with all three phases to validate the performance of the third phase.

We use Colossal-AI [62] as our training framework. Colossal-AI is a unified deep learning system that provides the fullest set of acceleration techniques for the AI community. With its modular design, ColossalAI allows for free combination of these techniques to achieve the best training speedup. Optimized parallelism and heterogeneous training methods are provided in Colossal-AI. These methods achieve better system performance than the baseline systems. They are provided for the user via friendly APIs with minimum code changes.

During training, AdamW [63] with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 1e^{-8}$ and weight decay = 0.1 is used to optimize the model. We use the cosine learning rate decay and the learning rate decays to 10% of its maximum. Learning Rate is set to $2e^{-6}$.

3.4 Alignment

After the pre-training stage with a base model, we further finetune the LLM to guide LLM as a helpful and harmless AI assistant. Our Alignment stage mainly uses supervised fine-tuning (SFT). During SFT, we fine-tune the model to follow diverse human instructions by high-quality instruction data. We use the Tulu v2 dataset [64] for instruction tuning. The dataset consists of a mix of FLAN, Open Assistant 1, ShareGPT, GPT4-Alpaca, LIMA and so on.

3.5 Long-Context

Our model leverages grouped-query attention (GQA) [44], sliding window attention (SWA) [45], and Rolling Buffer Cache. GQA reduces the memory requirement during decoding, allowing for higher batch sizes hence higher throughput.

Besides, SWA can handle longer sequences more effectively at a reduced computational cost, thereby alleviating a common limitation in LLMs. SWA exploits the stacked layers of a transformer to attend information beyond the window size W. At the last layer, with SWA, using a window size of $W = 4096$, we have a theoretical attention span of approximately $14K$ tokens or above. In practice, for a sequence length of $16K$ and $W = 4096$, changes made to FlashAttention and xFormers yield a $2\times$ speed improvement over a vanilla attention baseline.

Like Mistral [2], our model also adopts Rolling Buffer Cache which limit our cache size using a rolling buffer cache with a fixed attention span. The cache has a fixed size of W, and the keys and values for the timestep i are stored in position i mod W of the cache. As a result, when the position i is larger than W, past values in the cache are overwritten, and the size of the cache stops increasing. On a sequence length of $32k$ tokens, this reduces the cache memory usage by $8\times$, without impacting the model quality.

With the above techniques, our model can support $32K$ context length with fast inference and low memory cost.

4 Evaluation

We compare our model with popular LLMs, and re-run all benchmarks with the same evaluation pipeline for fair comparison. We use lm-evaluation-harness [65] and opencompass [66] for evaluation. Lm-evaluation-harness provides a unified framework to test generative language models on a large number of different evaluation tasks. It supports over 60 standard academic benchmarks for LLMs, with hundreds of subtasks and variants implemented. It support for models loaded via transformers (including quantization via AutoGPTQ [67]), GPT-NeoX [68], and Megatron-DeepSpeed [69], with a flexible tokenization-agnostic interface. Lm-evaluation-harness is the backend for Hugging Face's popular Open LLM Leaderboard, and is used internally by dozens of organizations including NVIDIA, Cohere, BigScience, BigCode, Nous Research, and Mosaic ML. OpenCompass performs an in-depth and holistic assessment of large language models, leveraging an extensive array of rigorously curated benchmarks across eight fundamental dimensions: language comprehension, knowledge precision, logical deduction, creative ideation, mathematical problem-solving, programming proficiency, extended text analysis, and intelligent agent engagement.

4.1 Evaluation Tasks

We evaluate the performance on various tasks below.

- AI2 Reasoning Challenge - a set of grade-school science questions.
- HellaSwag [61] - a test of commonsense natural language inference, which is easy for humans (95%) but challenging for SOTA models. It consists of 70,000 multiple-choice questions. Each question presents a scenario and four possible outcomes, asking to select the most reasonable conclusion.
- MMLU [70] - a test to measure a text model's multitask accuracy. The test covers 57 tasks including elementary mathematics, US history, computer science, law, and more.
- Winogrande [71] - an adversarial and difficult Winograd benchmark at scale, for commonsense reasoning. It contains 44,000 multiple-choice questions with two options each. It requires the model to choose the appropriate entity word for the pronoun in the descriptive text based on the scenario.

4.2 Performance

Our model before finetuning on the training data of the evaluation datasets is named Moxin-7B-original. After further partially finetuning Moxin-7B-original on the training data of the evaluation datasets, we can further obtain the model Moxin-7B-finetuned.

4.2.1 Zero-Shot Evaluation

We report the result of base models for zero-shot evaluation in Table 2. The tasks are listed below. After training with the training data of evaluation tasks, our Moxin-7B-finetuned can achieve superior performance compared with SOTA baselines. Consequently, our models emerge as an excellent candidate for real-world applications.

- AI2 Reasoning Challenge (0-shot)
- AI2 Reasoning Easy (0-shot)
- HellaSwag (0-shot)
- PIQA (0-shot)
- Winogrande (0-shot)

Table 2: Performance comparison for various models in zero-shot evaluation.

Models	HellaSwag	WinoGrade	PIQA	ARC-E	ARC-C	Ave
Mistral - 7B	80.39	73.4	82.15	78.28	52.22	73.29
LLaMA 2 - 7B	75.99	69.06	79.11	74.54	46.42	69.02
LLaMA 2 - 13B	79.37	72.22	80.52	77.4	49.06	71.71
LLaMA 3.1 - 8B	78.92	74.19	81.12	81.06	53.67	73.79
gemma - 7b	80.45	73.72	80.9	79.97	54.1	73.83
Qwen v2 - 7B	78.9	72.38	79.98	74.71	50.09	71.21
internlm2.5 - 7b	79.14	77.9	80.52	76.16	51.37	73.02
Baichuan2 - 7B	72.25	67.17	77.26	72.98	42.15	66.36
Yi-1.5-9B	77.86	73.01	80.74	79.04	55.03	73.14
deepseek - 7b	76.13	69.77	79.76	71.04	44.8	68.3
Moxin - 7B - original	72.06	66.31	78.07	71.47	48.15	67.21
Moxin - 7B - finetune	80.03	75.17	82.24	81.12	58.64	75.44

4.2.2 Few-Shot Evaluation

We report the result of base models in Table 3. The tasks and their few-show settings are listed below. Thanks to its rigorous and high-quality training corpus, our model demonstrates a remarkable competitive edge in tasks that involve language understanding and knowledge application. Our Moxin-7B-original achieves superior performance than LLaMA2-7B. After training with the training

data of evaluation tasks, our Moxin-7B-finetuned can achieve competitive performance compared with SOTA baselines. Consequently, our models emerge as an excellent choice for a multitude of real-world applications where the reliance on robust language comprehension and extensive knowledge is paramount.

- AI2 Reasoning Challenge (25-shot)
- HellaSwag (10-shot)
- MMLU (5-shot)
- Winogrande (5-shot)

Table 3: Performance comparison for various models in few-shot evaluation.

model	ARC-C	hellaswag	mmlu	WinoGrade	Ave
Mistral - 7B	57.59	83.25	62.42	78.77	70.51
LLaMA 3.1 - 8B	54.61	81.95	65.16	77.35	69.77
LLaMA 3 - 8B	55.46	82.09	65.29	77.82	70.17
LLaMA 2 - 7B	49.74	78.94	45.89	74.27	62.21
Qwen 2 - 7B	57.68	80.76	70.42	77.43	71.57
gemma - 7b	56.48	82.31	63.02	78.3	70.03
internlm2.5 - 7b	54.78	79.7	68.17	80.9	70.89
Baichuan2 - 7B	47.87	73.89	54.13	70.8	61.67
Yi-1.5-9B	58.36	80.36	69.54	77.53	71.48
Moxin - 7B - original	53.75	75.46	59.43	70.32	64.74
Moxin - 7B - finetuned	59.47	83.08	60.97	78.69	70.55

4.3 Alignment Evaluation

We evaluate the alignment performance on MTBench [72]. It is a two-round conversation dataset with 80 questions. It covers eight dimensions (reasoning, roleplay, math, coding, writing, humanities, STEM, and information extraction) with 10 questions for each dimension. The model needs to answer the first question and then refine its previous response following additional specific instructions. We use GPT-4 as a judge model to provide scores (between 1-10) for the quality of responses. Our Moxin-7B-chat achieves superior performance on MTbench compared with baselines as shown in Table 4.

Table 4: Performance for various chat models.

Model	MTbench
Moxin-7B-chat	**6.42**
Llama 2 13B Chat	6.65
Vicuna 13B	6.57
Llama 2 7B Chat	6.27
Vicuna 7B	6.17
Alpaca 13B	4.53

5 Conclusion

Many open LLM models lack the necessary components (such as training code and data) for full understanding and reproducibility, which may hinder further innovations on LLMs. To mitigate this issue, following MOF, we present a truly open source LLM Moxin 7B and release pre-training code and configurations, training and fine-tuning data, and intermediate and final checkpoints, aiming to make continuous commitments to fully open-source LLMs. Our Moxin achieves superior performance compared with SOTA baselines.

References

[1] J. Achiam, S. Adler, S. Agarwal, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[2] A. Q. Jiang, A. Sablayrolles, A. Mensch, *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.

[3] G. Team, R. Anil, S. Borgeaud, *et al.*, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[4] A. Dubey, A. Jauhri, A. Pandey, *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[5] T. Prest, P.-A. Fouque, J. Hoffstein, *et al.*, "Falcon," *Post-Quantum Cryptography Project of NIST*, 2020.

[6] J. Bai, S. Bai, Y. Chu, *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.

[7] A. Yang, B. Yang, B. Hui, *et al.*, "Qwen2 technical report," *arXiv preprint arXiv:2407.10671*, 2024.

[8] O. Lieber, B. Lenz, H. Bata, *et al.*, "Jamba: A hybrid transformer-mamba language model," *arXiv preprint arXiv:2403.19887*, 2024.

[9] J. Team, B. Lenz, A. Arazi, *et al.*, "Jamba-1.5: Hybrid transformer-mamba models at scale," *arXiv preprint arXiv:2408.12570*, 2024.

[10] R. Sennrich, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[11] O. Team, *Tiktoken*, 2022. [Online]. Available: https://github.com/openai/tiktoken.

[12] T. Kudo, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[13] Z. Yang, "Xlnet: Generalized autoregressive pretraining for language understanding," *arXiv preprint arXiv:1906.08237*, 2019.

[14] H. F. Team, "Summary of the tokenizers," 2024. [Online]. Available: https://github.com/huggingface/transformers/blob/main/docs/source/en/tokenizer_summary.md.

[15] M. Reid, N. Savinov, D. Teplyashin, *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024.

[16] M. Maaz, H. Rasheed, S. Khan, and F. S. Khan, "Video-chatgpt: Towards detailed video understanding via large vision and language models," *arXiv preprint arXiv:2306.05424*, 2023.

[17] H. Zhang, X. Li, and L. Bing, "Video-llama: An instruction-tuned audio-visual language model for video understanding," *arXiv preprint arXiv:2306.02858*, 2023.

[18] D. Zhang, Y. Yu, C. Li, *et al.*, "Mm-llms: Recent advances in multimodal large language models," *arXiv preprint arXiv:2401.13601*, 2024.

[19] A. Q. Jiang, A. Sablayrolles, A. Roux, *et al.*, "Mixtral of experts," *arXiv preprint arXiv:2401.04088*, 2024.

[20] B. Mann, N. Ryder, M. Subbiah, *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, vol. 1, 2020.

[21] G. Penedo, Q. Malartic, D. Hesslow, *et al.*, "The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only," *arXiv preprint arXiv:2306.01116*, 2023.

[22] J. W. Rae, S. Borgeaud, T. Cai, *et al.*, "Scaling language models: Methods, analysis & insights from training gopher," *arXiv preprint arXiv:2112.11446*, 2021.

[23] G. Wenzek, M.-A. Lachaux, A. Conneau, *et al.*, "Ccnet: Extracting high quality monolingual datasets from web crawl data," *arXiv preprint arXiv:1911.00359*, 2019.

[24] L. Xue, "Mt5: A massively multilingual pre-trained text-to-text transformer," *arXiv preprint arXiv:2010.11934*, 2020.

[25] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[26] A. Conneau and G. Lample, "Cross-lingual language model pretraining," *Advances in neural information processing systems*, vol. 32, 2019.

[27] M. Chen, J. Tworek, H. Jun, *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021.

[28] L. Gao, S. Biderman, S. Black, *et al.*, "The pile: An 800gb dataset of diverse text for language modeling," *arXiv preprint arXiv:2101.00027*, 2020.

[29] N. Du, Y. Huang, A. M. Dai, *et al.*, "Glam: Efficient scaling of language models with mixture-of-experts," in *International Conference on Machine Learning*, PMLR, 2022, pp. 5547–5569.

[30] S. Longpre, G. Yauney, E. Reif, *et al.*, "A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity," *arXiv preprint arXiv:2305.13169*, 2023.

[31] N. Sachdeva, B. Coleman, W.-C. Kang, *et al.*, "How to train data-efficient llms," *arXiv preprint arXiv:2402.09668*, 2024.

[32] K. Lee, D. Ippolito, A. Nystrom, *et al.*, "Deduplicating training data makes language models better," *arXiv preprint arXiv:2107.06499*, 2021.

[33] A. Agarwal, H. S. Koppula, K. P. Leela, *et al.*, "Url normalization for de-duplication of web pages," in *Proceedings of the 18th ACM conference on information and knowledge management*, 2009, pp. 1987–1990.

[34] A. Albalak, L. Pan, C. Raffel, and W. Y. Wang, "Efficient online data mixing for language model pre-training," in *R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023.

[35] Z. Shen, T. Tao, L. Ma, *et al.*, "Slimpajama-dc: Understanding data combinations for llm training," *arXiv preprint arXiv:2309.10818*, 2023.

[36] J. Li, A. Fang, G. Smyrnis, *et al.*, "Datacomp-lm: In search of the next generation of training sets for language models," *arXiv preprint arXiv:2406.11794*, 2024.

[37] A. Chowdhery, S. Narang, J. Devlin, *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[38] G. Team, T. Mesnard, C. Hardin, *et al.*, "Gemma: Open models based on gemini research and technology," *arXiv preprint arXiv:2403.08295*, 2024.

[39] L. Soldaini, R. Kinney, A. Bhagia, *et al.*, "Dolma: An open corpus of three trillion tokens for language model pretraining research," *arXiv preprint arXiv:2402.00159*, 2024.

[40] G. Penedo, H. Kydlíček, A. Lozhkov, *et al.*, "The fineweb datasets: Decanting the web for the finest text data at scale," *arXiv preprint arXiv:2406.17557*, 2024.

[41] M. Ostendorff, P. O. Suarez, L. F. Lage, and G. Rehm, "Llm-datasets: An open framework for pretraining datasets of large language models,"

[42] A. Lozhkov, R. Li, L. B. Allal, *et al.*, "Starcoder 2 and the stack v2: The next generation," *arXiv preprint arXiv:2402.19173*, 2024.

[43] B. Roziere, J. Gehring, F. Gloeckle, *et al.*, "Code llama: Open foundation models for code," *arXiv preprint arXiv:2308.12950*, 2023.

[44] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, "Gqa: Training generalized multi-query transformer models from multi-head checkpoints," *arXiv preprint arXiv:2305.13245*, 2023.

[45] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[46] D. Soboleva, F. Al-Khateeb, R. Myers, J. R. Steeves, J. Hestness, and N. Dey, *SlimPajama: A 627B token cleaned and deduplicated version of RedPajama*, https://cerebras.ai/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama, 2023. [Online]. Available: https://huggingface.co/datasets/cerebras/SlimPajama-627B.

[47] H. Touvron, T. Lavril, G. Izacard, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[48] A. Abbas, K. Tirumala, D. Simig, S. Ganguli, and A. S. Morcos, "Semdedup: Data-efficient learning at web-scale through semantic deduplication," *arXiv preprint arXiv:2303.09540*, 2023.

[49] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.

[50] *Large-scale near-deduplication behind bigcode*, 2023. [Online]. Available: https://huggingface.co/blog/dedup.

[51] J. Leskovec, A. Rajaraman, and J. Ullman, *Mining of massive datasets, cambridge university press, cambridge*, 2014.

[52] A. Z. Broder, "On the resemblance and containment of documents," in *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, IEEE, 1997, pp. 21–29.

[53] *Fineweb*, 2024. [Online]. Available: https://huggingface.co/datasets/HuggingFaceFW/fineweb.

[54] D. Brandfonbrener, H. Zhang, A. Kirsch, J. R. Schwarz, and S. Kakade, "Color-filter: Conditional loss reduction filtering for targeted language model pre-training," *arXiv preprint arXiv:2406.10670*, 2024.

[55] A. Fang, A. M. Jose, A. Jain, L. Schmidt, A. Toshev, and V. Shankar, "Data filtering networks," *arXiv preprint arXiv:2309.17425*, 2023.

[56] D. Groeneveld, I. Beltagy, P. Walsh, *et al.*, "Olmo: Accelerating the science of language models," *arXiv preprint arXiv:2402.00838*, 2024.

[57] D. Kocetkov, R. Li, L. B. Allal, *et al.*, "The stack: 3 tb of permissively licensed source code," *arXiv preprint arXiv:2211.15533*, 2022.

[58] A. Zeng, X. Liu, Z. Du, *et al.*, "GLM-130b: An open bilingual pre-trained model," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=-AwOrrrPUF.

[59] J. Parmar, S. Prabhumoye, J. Jennings, *et al.*, "Nemotron-4 15b technical report," *arXiv preprint arXiv:2402.16819*, 2024.

[60] D. Hendrycks, C. Burns, S. Basart, *et al.*, *Measuring massive multitask language understanding*, 2021. arXiv: 2009.03300 [cs.CY]. [Online]. Available: https://arxiv.org/abs/2009.03300.

[61] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?" *arXiv preprint arXiv:1905.07830*, 2019.

[62] S. Li, H. Liu, Z. Bian, *et al.*, "Colossal-ai: A unified deep learning system for large-scale parallel training," in *Proceedings of the 52nd International Conference on Parallel Processing*, 2023, pp. 766–775.

[63] I. Loshchilov, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[64] H. Ivison, Y. Wang, V. Pyatkin, *et al.*, "Camels in a changing climate: Enhancing lm adaptation with tulu 2," *arXiv preprint arXiv:2311.10702*, 2023.

[65] L. E. H. Team, *Lm evaluation harness*, Accessed: Summer 2024, 2024. [Online]. Available: https://github.com/EleutherAI/lm-evaluation-harness.

[66] O. C. Team, *Open compass*, Accessed: Summer 2024, 2024. [Online]. Available: https://github.com/open-compass/opencompass.

[67] A. Team, *Autogptq: An user-friendly llms quantization package*, Accessed: Spring 2024, 2024. [Online]. Available: https://github.com/AutoGPTQ/AutoGPTQ.

[68] S. Black, S. Biderman, E. Hallahan, *et al.*, "Gpt-neox-20b: An open-source autoregressive language model," *arXiv preprint arXiv:2204.06745*, 2022.

[69] S. L. Song, B. Kruft, M. Zhang, *et al.*, "Deepspeed4science initiative: Enabling large-scale scientific discovery through sophisticated ai system technologies," *arXiv preprint arXiv:2310.04610*, 2023.

[70] T. van der Weij, F. Hofstätter, O. Jaffe, S. F. Brown, and F. R. Ward, "Ai sandbagging: Language models can strategically underperform on evaluations," *arXiv preprint arXiv:2406.07358*, 2024.

[71] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, "Winogrande: An adversarial winograd schema challenge at scale," *Communications of the ACM*, vol. 64, no. 9, pp. 99–106, 2021.

[72] L. Zheng, W.-L. Chiang, Y. Sheng, *et al.*, "Judging llm-as-a-judge with mt-bench and chatbot arena," *Advances in Neural Information Processing Systems*, vol. 36, pp. 46 595–46 623, 2023.