FOX-1 TECHNICAL REPORT

Zijian Hu[*1], **Jipeng Zhang**[*2], **Rui Pan**[*3], **Zhaozhuo Xu**[1], **Salman Avestimehr**[1], **Chaoyang He**[†1], **Tong Zhang**[†1,3]
[1]TensorOpera Inc.
[2]The Hong Kong University of Science and Technology
[3]University of Illinois Urbana-Champaign
`{zjh, zhaozhuo, avestimehr, ch}@tensoropera.com`
`jzhanggr@ust.hk ruip4@illinois.edu tongzhang@tongzhang-ml.org`

ABSTRACT

We present Fox-1, a series of small language models (SLMs) consisting of Fox-1-1.6B and Fox-1-1.6B-Instruct-v0.1. These models are pre-trained on 3 trillion tokens of web-scraped document data and fine-tuned with 5 billion tokens of instruction-following and multi-turn conversation data. Aiming to improve the pre-training efficiency, Fox-1-1.6B model introduces a novel 3-stage data curriculum across all the training data with 2K-8K sequence length. In architecture design, Fox-1 features a deeper layer structure, an expanded vocabulary, and utilizes Grouped Query Attention (GQA), offering a performant and efficient architecture compared to other SLMs. Fox-1 achieves better or on-par performance in various benchmarks compared to StableLM-2-1.6B, Gemma-2B, Qwen1.5-1.8B, and OpenELM1.1B, with competitive inference speeds when handling a large number of user requests. The model weights have been released under the Apache 2.0 license, where we aim to promote the democratization of LLMs and make them fully accessible to the whole open-source community.

Base Model: `https://huggingface.co/tensoropera/Fox-1-1.6B`
Chat Model: `https://huggingface.co/tensoropera/Fox-1-1.6B-Instruct-v0.1`

1 Introduction

Recent advances in Large Language Models (LLMs) show that the unified next-token prediction paradigm excels in improving performance across diverse text generation tasks, such as code generation Chen et al. [2021], math word problem solving Wei et al. [2022a], medical question answering Diao et al. [2023]. Furthermore, LLMs Brown et al. [2020], OpenAI [2023] are widely acknowledged for their System 2 capabilities Deng et al. [2023], Saha et al. [2023], Weston and Sukhbaatar [2023] in in-context learning Brown et al. [2020] and chain-of-thought prompting Wei et al. [2022a]. Generally, LLMs are developed by training a decoder-only transformer model on extensive text corpora Brown et al. [2020], Rae et al. [2021]. Currently, most state-of-the-art models achieve high performance through fine-tuning white-box LLMs Touvron et al. [2023a,b], Dubey et al. [2024] or using black-box LLM APIs OpenAI [2023], Ouyang et al. [2022].

To further enhance the capabilities of decoder-only language models, training LLMs focuses more on their scaling laws Kaplan et al. [2020] and compute-optimal training schedules. The key components include model size, the volume of training data, and the corresponding training compute. Several studies Hoffmann et al. [2022] provide guidance on optimally balancing the increase in model size with the amount of training data, consistently showing that large models require extensive datasets and substantial training compute. Given that existing LLMs often reach hundreds of billions of parameters Touvron et al. [2023a,b], Ouyang et al. [2022], Dubey et al. [2024] for better performance, the training costs for these giant models are extremely high, leading to significant financial and environmental impacts. These costs also restrict most academic researchers and resource-limited engineers from participating, and the high deployment costs further hinder the widespread application of these advanced AI technologies.

[*]Equal contributions
[†]Corresponding author

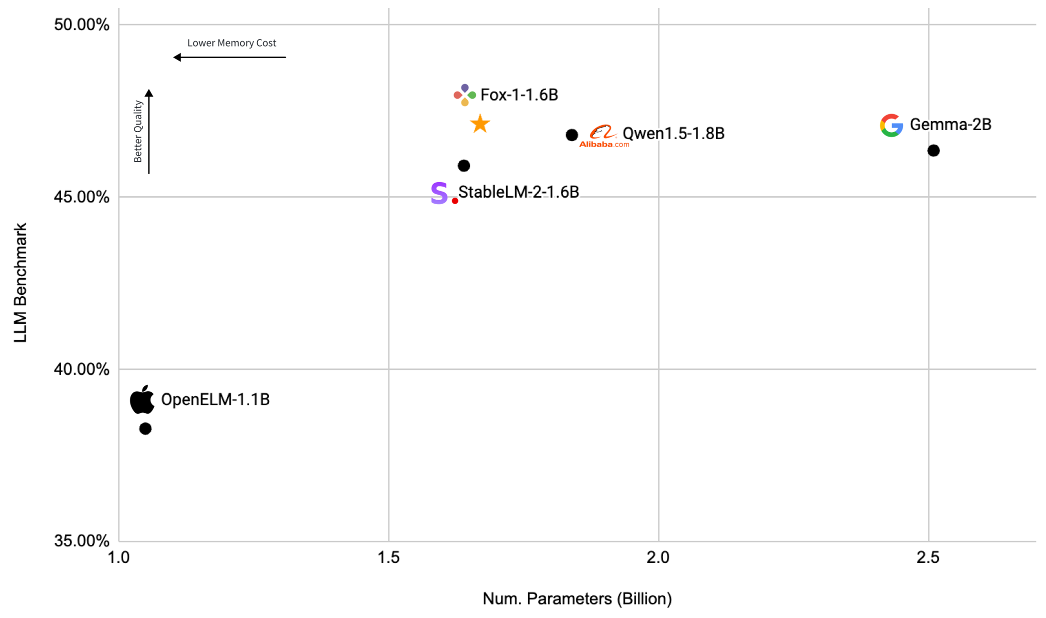

Figure 1: Fox-1-1.6B compare with other SLMs.

Recently, there has been a growing interest in training Small Language Models (SLMs) that achieve performance comparable to models four times their size. Examples of these models include the Phi series Gunasekar et al. [2023a], Li et al. [2023], Abdin et al. [2024], TinyLlama Zhang et al. [2024], OpenELM Mehta et al. [2024], Gemma Team et al. [2024], MiniCPM Hu et al. [2024] and Qwen Bai et al. [2023]. Research on these models explores various aspects of training effective SLMs, such as data paradigms, model architecture, and tokenizers. However, it remains unclear how to further enhance the training curriculum and data organization at each stage.

In this report, we introduce Fox-1, a series of Small Language Models (SLMs) that primarily explore research issues related to training curricula and present a high-performance SLM with extensive investigation into training techniques. We release all our model weights publicly under the Apache 2.0 license, making them accessible on the TensorOpera AI Platform and Hugging Face [3].

2 Pre-Training

Fox-1 pre-training involves curating and filtering a large corpus of open-sourced textual data, searching the model architecture, and developing the training recipe. In this section, we present the details of the above components.

2.1 Data

We utilize 3 trillion textual data to train large language models. To align with our proposed three-stage curriculum pre-training pipeline, we have reorganized the original data into three distinct collections. We have gathered an extensive range of textual data, encompassing unsupervised and instruction-tuning datasets, as well as diverse domains such as code, web content, mathematics, and science documents. To simplify the description of the data combination in the subsequent sections, we first list the dataset collection and then detail the data usage for each stage.

Raw Data Collection We collected datasets from publicly released large and high-quality sources, including Redpajama Computer [2023], SlimPajama Soboleva et al. [2023], Dolma Soldaini et al. [2024], Pile Gao et al. [2020], and Falcon Penedo et al. [2023] datasets. Here are the detailed datasets:

[3]Base model is available at `https://huggingface.co/tensoropera/Fox-1-1.6B` and the instruction-tuned version is available at `https://huggingface.co/tensoropera/Fox-1-1.6B-Instruct-v0.1`

- **Common Crawl.** We selected the Common Crawl (CC) split from all four datasets, which occupies almost 5T of storage. The SlimPajama-CC ranges from 2019 to 2023, Dolma-CC contains all the CC dumps before 2023, Pile-CC includes selected subsets from dumps across 2013 to 2020, and Falcon-CC ranges from 2008 to 2023. Deduplication was performed across all the CC subsets using a bloom filter algorithm Bloom [1970].
- **C4.** As demonstrated in the Llama-1 paper Touvron et al. [2023a], including diverse preprocessed Common Crawl datasets improves LLM performance. We also include the widely-recognized high-quality Common Crawl datasets C4 Raffel et al. [2020].
- **Books.** Following the books collection in the Pile dataset, we include Books3 Presser [2020], BookCorpus2 Zhu et al. [2015], and the Gutenberg Rae et al. [2019] collection in our Books dataset.
- **Wikipedia.** We selected Redpajama's Wiki collection, based on Wikipedia dumps from 2023-03-20, containing text in 20 different languages. This collection is similar to the Wiki collection in Llama.
- **Papers.** RedPajama, Dolma, and Pile all have their own paper collections. Dolma uses the pes2o Soldaini and Lo [2023] dataset from AI2, which only contains abstracts of papers. For the paper collections in RedPajama and Pile, we also performed deduplication using a bloom filter algorithm.
- **Code.** Stack Kocetkov et al. [2022] is a 3.1T dataset containing permissively licensed source code across 30 languages. Since it maintains license permission checking, we include this as our code pre-training data collection.
- **Science Text.** We include PubMed, OpenWebText2 Gokaslan and Cohen [2019], FreeLaw, USPTO Backgrounds, PhilPapers, and NIH Grant Abstracts from the Pile Gao et al. [2020] in our collection.
- **WebQ&A.** The Reddit data in Dolma and StackExchange in SlimPajama are also collected separately to construct a web-based multi-round question answering dataset.
- **Math.** We include Open-Web-Math Paster et al. [2023] and algerbraic-stack in the pre-training collection to enhance the model's mathematical abilities.
- **Instruction.** We collect several instruction or domain-specific datasets under permissive licenses to further increase the quality of the data.
- **RewriteBooks.** We select a subset of high-quality books and rewrite them with the Mixtral-8x22B Jiang et al. [2024] model to construct the RewriteBooks collection.
- **Others.** We got several other datasets widely used for training Language models like Flan Wei et al. [2022b], which is curated as an unification of various NLP taks.

First Stage Data This stage is comparable to the pre-training stage of existing large language models. For this stage, we randomly sampled half of the Common Crawl collection data to complete the first stage of pre-training. By initially using several Common Crawl-based datasets, we aim to create a more balanced distribution for the following training stages.

Dataset	Storage Size	Tokens	Average Seq Length
Common Crawl	7.9TB	1.05T	591.29

Table 1: Dataset statistics for the first stage training of collection. Here, "TB" refers to terabytes for storage statistics, while "T" denotes the number of tokens in the respective training datasets.

Second Stage Data In this stage, we use a smaller scale of training data while enhancing long-context capabilities. To achieve this, we collect data from more diverse domains. This includes another part of the Common Crawl collection, Books, C4, Math, Papers, Wiki, Code, and all subsets from Science Text and WebQ&A.

Third Stage Data In the final stage, we aim to curate an extremely high-quality data collection. As demonstrated in Phi-family models [Gunasekar et al., 2023b, Li et al., 2023], machine-generated datasets are incorporated at this stage. Consequently, we include the Instruction and RewriteBooks subsets in this stage's data collection. We also down-sample a subset with 100B tokens from the second stage data mix for this stage.

Dataset	**Storage Size**	**Num. Tokens**	**Seq. Length Avg.**	**Chunk Length**
Common Crawl	9.93 TB	1.47 T	1269.09	4K
The Stack	1.89 TB	375.98 B	1783.49	4K
C4	853.92 GB	169.75 B	471.69	4K
Papers	785.42 GB	121.18 B	2837.34	4K
Dolma_Reddit	424.61 GB	79.32 B	210.13	4K
PubMed_Central	227.16 GB	45.35 B	7984.65	8K
OpenWebText2	142.76 GB	28.12 B	869.78	4K
Books	124.54 GB	24.91 B	121057	8K
RedPajama StackExchange	102.49 GB	20.15 B	680.03	4K
Wikipedia	101.49 GB	20.03 B	671.32	4K
Freelaw	98.42 GB	19.63 B	3872.02	8K
Open-Web-Math	66.59 GB	13.22 B	2093.74	4K
USPTO_Backgrounds	49.19 GB	9.67 B	869.23	4K
PhilPapers	5.64 GB	1.13 B	17639.44	8K
NIH ExPorter	3.72 GB	720.67 M	405.34	4K
Total	14.35 TB	2.30 T	1105.56	-

Table 2: Dataset statistics for the second stage training of collection. Here, "TB" and "GB" indicate terabytes and gigabytes for storage space statistics, while "T", "B", "M", and "K" denote the number of tokens in the respective training datasets.

Dataset	**Final Size**	**Num. Tokens**	**Ratio**
Instruction	183.87 GB	36.02 B	34.07%
The Stack	94.34 GB	18.79 B	17.78%
flan	29.05 GB	15.47 B	14.63%
algebraic-stack	10.92 GB	12.51 B	11.83%
Papers	30.63 GB	6.06 B	5.73%
Books	7.75 GB	5.04 B	4.77%
Dolma_Reddit	20.14 GB	3.96 B	3.74%
PubMed Central	11.66 GB	2.26 B	2.14%
OpenWebText2	7.35 GB	1.40 B	1.33%
RedPajama_StackExchange	5.36 GB	1.00 B	0.95%
Wikipedia	5.33 GB	997.75 M	0.94%
Freelaw	5.23 GB	977.90 M	0.93%
Open-Web-Math	3.63 GB	657.70 M	0.62%
USPTO_Backgrounds	2.74 GB	479.97 M	0.45%
PhilPapers	601.02 MB	52.96 M	0.05%
NIH ExPorter	499.56 MB	32.65 M	0.03%
Total	**419.07 GB**	**105.71 B**	**100.00%**

Table 3: Dataset statistics for the second stage training of collection. Here, "TB", "GB" and "MB" refer to terabytes, gigabytes and megabytes for storage space statistics, while "B" and "M" denote the number of tokens in the respective training datasets.

2.1.1 Tokenization

We hypothesize that it is easier to correctly predict a few consecutive tokens than many consecutive tokens Le Scao et al. [2023], Team et al. [2024], Tao et al. [2024], Kudo [2018]. Therefore, we believe using a large vocabulary could lead to better downstream performance. Arguably, a large vocabulary typically yields fewer tokens for a given text corpus Le Scao et al. [2023], Team et al. [2024], which could lead to better inference performance.

Although Fox-1 is an English-only SLM, we aim to select a sufficiently general vocabulary to leverage intermediate checkpoints for future multilingual capabilities and domain-specific applications. We selected the tokenizer from Team et al. [2024], which offers a vocabulary size of 256K, making it one of the largest available at the time of this report.

Increasing the size of vocabulary has at least two major benefits. First, the effective context length is implicitly extended given the denser information encoded in each token. For example, a vocabulary with size 26 can only encode one character in [a-z], but a vocabulary with size 26^2 can encode two alphabetical letters at a time, leading to longer representable strings with a fixed length of tokens. Second, a larger vocabulary size reduces the probability of unknown words or phrases, resulting in better downstream task performance in practice.

2.1.2 Model Architecture

Fox-1 employs a decoder-only transformer architecture inspired by Touvron et al. [2023b,a], Dubey et al. [2024], Liu et al. [2024] with 1.6B total parameters while introducing various improvements and redesigns for better performance.

Deeper Network A fundamental trade-off in model architecture design is depth versus width. While wider and shallower networks allow better memorization, deeper and thinner networks present stronger reasoning ability [Cheng et al., 2016, Liu et al., 2024]. In accordance with this principle, Fox-1 chose a deeper architecture than most modern SLMs. Specifically, Fox-1 consists 32 self-attention layers, which is 78% deeper than Gemma-2B (18 layers), 33% deeper than StableLM-2-1.6B (24 layers), and 15% deeper than Qwen2.5-1.5B (28 layers).

Shared Embedding Fox-1 utilizes a large vocabulary of 256,000, resulting in approximately 0.5 billion parameters with an embedding size of 2,048. Larger models typically use separate embedding layers for the input (vocabulary size to embedding size) and output (embedding size to vocabulary size). For Fox-1, the embedding layers alone would require 1 billion parameters. To reduce the total number of parameters, we follow the approach of Zhang et al. [2022], Liu et al. [2024] and share the input and output embedding layer, maximizing weight utilization and reducing the parameter count by 0.5 billion, approximately 30.49% of the final model size.

Pre-normalization Similar to Touvron et al. [2023b], we use RMSNorm [Zhang and Sennrich, 2019] to normalize the input of each transformer layer. RMSNorm is the dominant choice of pre-normalization in modern large language models [Team et al., 2024, Bai et al., 2023, Touvron et al., 2023b], where it demonstrates better efficiency than LayerNorm [Ba, 2016].

Feature	**Fox-1**
Parameters	1.6B
Attention Mechanism	GQA [Ainslie et al., 2023]
Non-linearity	SwiGLU [Shazeer, 2020]
Normalization	RMSNorm [Touvron et al., 2023b]
Vocabulary Size	256K
Context length	8K
Embedding Share	True

Table 4: Fox-1 Model Architecture.

Rotary Positional Embeddings (RoPE) Fox accepts at most 8K-length tokens by default. To improve the generalization performance for a longer context window, the widely adopted RoPE [Su et al., 2024] is employed to facilitate encoding of relative positional dependency between tokens, with θ set to $10,000$.

Grouped Query Attention (GQA) Grouped Query Attention (GQA) [Ainslie et al., 2023] divides the query heads of multihead attention layers into groups of shared key-value heads. We use Grouped Query Attention (GQA) [Ainslie et al., 2023] with 4 key-value heads and 16 attention heads to improve training and inference speed and reduce memory usage.

2.1.3 Training

The pre-training of long sequences is known to be challenging given the training inefficiency incurred by the quadratic complexity of the attention mechanism [Vaswani, 2017]. To mitigate this problem, a 3-stage curriculum learning strategy is introduced in the pre-training of Fox, where the chunk length of the training sample is gradually increased from 2K to 8K to ensure the long-context ability of Fox at a small cost.

Specifically, stage 1 comprises $\sim 39\%$ total data samples in the whole pre-training process, where the 1.05T-token dataset is chunked into 2K-length samples, with a batch size 2M tokens. We use 2,000 steps of linear warm-up for

Stages	Dataset Size	Chunk length	Batch Size
Stage 1	1.05T	2K	2M Tokens
Stage 2	1.58T	4K-8K	4M Tokens
Stage 3	0.06T	4K-8K	4M Tokens

Table 5: Fox-1 Pre-training Sequence Length Curriculum: In this context, "T" represents the total number of tokens in the respective training datasets. Chunk length specifies the number of tokens per sample within a mini-batch. It is worth noting that the actual number of tokens used slightly differs from previous counts in the dataset section. This discrepancy arises because a subset was sampled with a limited number of training steps.

this stage. Stage 2 further includes $\sim 59\%$ samples with 1.58T tokens and increases the chunk length from 2K to 4K and 8K. The actual chunking lengths vary across different data sources, which are decided based on the natural average lengths in each data source. The batch size also grows to 4M for improving the training efficiency, given stage 2 being the most time-consuming stage with diverse sources of different datasets. Finally, in stage 3, Fox is trained with 62B tokens ($\sim 0.02\%$) of high quality data to lay the groundwork for different downstream task abilities, such as instruction-following, chitchat, domain-specific question-answer, etc. Across all stages, we utilize AdamW and WSD schedule [Hu et al., 2024] to train Fox, along with learning rate 4×10^{-4}, weight decay 0.1, AdamW $\beta_1 = 0.9, \beta_2 = 0.95$, WSD temperature $16,000$.

3 Experimental Results

We evaluated Fox-1 and other SLMs on ARC Challenge (25-shot) [Clark et al., 2018], HellaSwag (10-shot) [Zellers et al., 2019], TruthfulQA (0-shot) [Lin et al., 2021], MMLU (5-shot) [Hendrycks et al., 2020], Winogrande (5-shot) [Sakaguchi et al., 2021], and GSM8k (5-shot) [Cobbe et al., 2021]. We follow the Open LLM Leaderboard's evaluation setup and report the average score of the 6 benchmarks. All models are evaluated on the same machine with 8$\times$H100 GPUs and the same software environment.

Model	Parameters	Average score*	Math (GSM8k)	World Knowledge (MMLU)
TensorOpera/Fox-1-1.6B	1.67B	**47.13%**	**36.39%**	43.05%
Alibaba/Qwen-1.5-1.8B	1.84B	46.81%	34.04%	**47.15%**
Google/Gemma-2B	2.51B	46.36%	17.06%	41.15%
StabilityAI/StableLM-2-1.6B	1.64B	45.92%	17.74%	39.16%
Apple/OpenELM-1.1B	1.05B	38.28%	2.27%	27.28%

Table 6: Fox-1 performance compared to other SLMs. *Average score: the average of ARC, HellaSwag, MMLU, GSM8k, TruthfulQA, and Winograde.

As shown in figure 2 and table 6, Fox-1 is better than or on par with Gemma-2B [Team et al., 2024], Qwen1.5-1.8B [Bai et al., 2023], StableLM-2-1.6B [Bellagente et al., 2024], and OpenELM1.1B [Mehta et al., 2024] on standard LLM benchmarks. For GSM8k, Fox-1 achieves 36.39%, which is higher than all of our baselines. Fox-1 also outperforms Gemma-2B, StableLM-2-1.6B, and OpenELM 1.1B on MMLU despite having 50% fewer parameters than Gemma-2B.

Inference Efficiency of Fox-1 We evaluated the end-to-end inference efficiency of Fox-1, Qwen1.5-1.8B, and Gemma-2B using vLLM with the TensorOpera serving platform in BF16 precision on one NVIDIA H100. This benchmark simulates real-world usage as closely as possible by sending multiple concurrent requests to the same inference server to mimic multi-user scenarios. We use the OpenOrca dataset and evaluate output tokens per user per second. Each user request contains a prompt with 234 tokens on average, and each response comprises 512 tokens. As shown in figure 3, in terms of operational efficiency, Fox-1 achieves an impressive throughput of over 200 tokens per second on the TensorOpera model serving platform, surpassing the performance of Gemma-2B and equaling that of Qwen1.5-1.8B in identical deployment environments. The high throughput of Fox-1 can largely be attributed to its architectural design, which incorporates Grouped Query Attention (GQA) for more efficient query processing. More specifically, by divining the query heads into groups where each group shares the same key-value head, Fox-1 significantly improves inference latency and enhances response times.

We omitted OpenELM since vLLM does not support it. With BF16, Fox-1 only needs 3703MB of GPU Memory, while Qwen1.5-1.8B, StableLM-2-1.6B, and Gemma-2B, respectively, need 4739MB, 3852MB, and 5379MB.

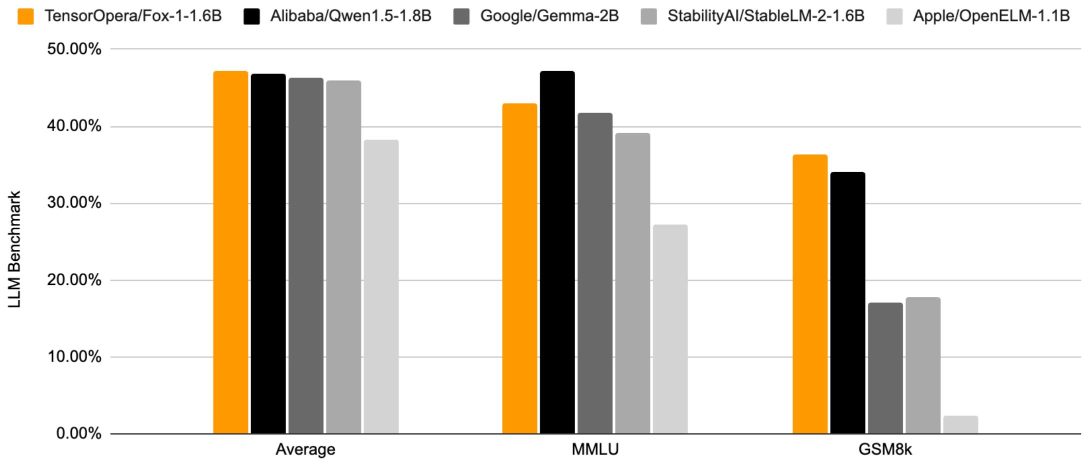

Figure 2: Performance compared to other SLMs.

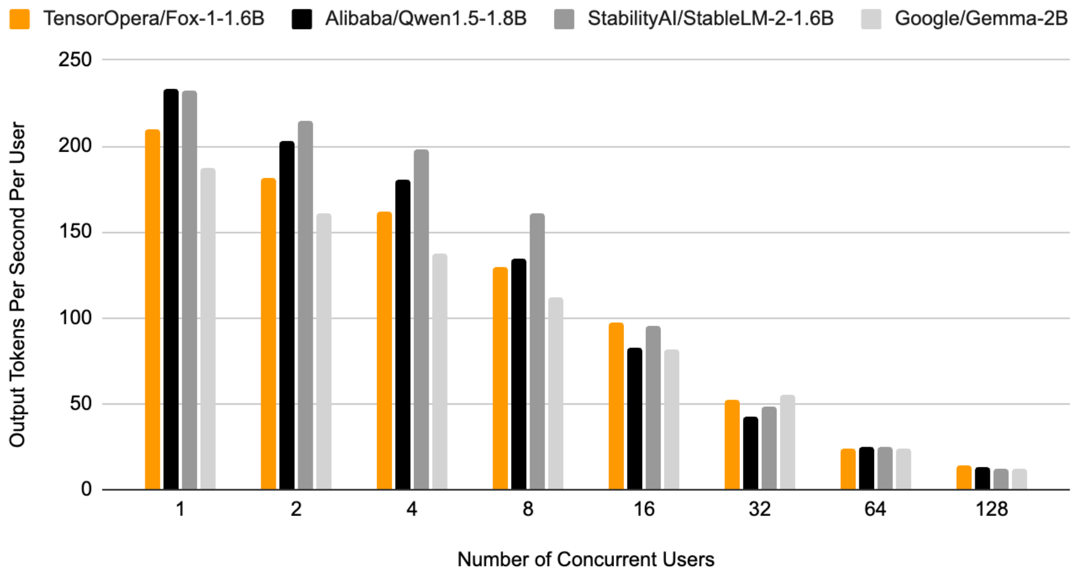

Figure 3: Inference speed compared to other SLMs.

We evaluated the end-to-end inference efficiency of Fox-1, Qwen1.5-1.8B, and Gemma-2B using vLLM with the TensorOpera serving platform in BF16 precision on one NVIDIA H100. This benchmark simulates real-world usage as closely as possible by sending multiple concurrent requests to the same inference server to mimic multi-user scenarios. We use the OpenOrca dataset and evaluate output tokens per user per second. Each user request contains a prompt with 234 tokens on average, and each response comprises 512 tokens. As shown in figure 3, in terms of operational efficiency, Fox-1 achieves an impressive throughput of over 200 tokens per second on the TensorOpera model serving platform, surpassing the performance of Gemma-2B and equaling that of Qwen1.5-1.8B in identical deployment environments. The high throughput of Fox-1 can largely be attributed to its architectural design, which incorporates Grouped Query Attention (GQA) for more efficient query processing. More specifically, by divining the query heads into groups where each group shares the same key-value head, Fox-1 significantly improves inference latency and enhances response times.

4 Conclusion

The Fox-1 series of small language models, including Fox-1-1.6B and Fox-1-1.6B-Instruct-v0.1, show advancement in efficient pre-training and instruction-following capabilities. Through the 3-stage data curriculum and a deeper architecture with a large vocabulary size, Fox-1 excels across various benchmarks, outperforming or matching other small language models like StableLM-2-1.6B and Gemma-2B. The success of Fox-1 demonstrates the possibility of pre-training language models with competitive performance even with limited data resources.

References

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022a.

Shizhe Diao, Rui Pan, Hanze Dong, Ka Shun Shum, Jipeng Zhang, Wei Xiong, and Tong Zhang. Lmflow: An extensible toolkit for finetuning and inference of large foundation models. *arXiv preprint arXiv:2306.12420*, 2023.

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

OpenAI. Gpt-4 technical report, 2023.

Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. Rephrase and respond: Let large language models ask better questions for themselves. *arXiv preprint arXiv:2311.04205*, 2023.

Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. Branch-solve-merge improves large language model evaluation and generation. *arXiv preprint arXiv:2310.15123*, 2023.

Jason Weston and Sainbayar Sukhbaatar. System 2 attention (is something you might need too). *arXiv preprint arXiv:2311.11829*, 2023.

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.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need. *CoRR*, abs/2306.11644, 2023a. doi:10.48550/ARXIV.2306.11644. URL `https://doi.org/10.48550/arXiv.2306.11644`.

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*, 2023.

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, et al. Openelm: An efficient language model family with open-source training and inference framework. *arXiv preprint arXiv:2404.14619*, 2024.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, April 2023. URL `https://github.com/togethercomputer/RedPajama-Data`.

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, June 2023. URL `https://huggingface.co/datasets/cerebras/SlimPajama-627B`.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, et al. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint arXiv:2402.00159*, 2024.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

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. *arXiv preprint arXiv:2306.01116*, 2023.

Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7): 422–426, 1970.

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. *Journal of machine learning research*, 21(140):1–67, 2020.

Shawn Presser. Books3. `https://twitter.com/theshawwn/status/1320282149329784833`, 2020.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint*, 2019. URL `https://arxiv.org/abs/1911.05507`.

Luca Soldaini and Kyle Lo. peS2o (Pretraining Efficiently on S2ORC) Dataset. Technical report, Allen Institute for AI, 2023. ODC-By, `https://github.com/allenai/pes2o`.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, et al. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. *arXiv preprint arXiv:2310.06786*, 2023.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022b.

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*, 2023b.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. 2023.

Chaofan Tao, Qian Liu, Longxu Dou, Niklas Muennighoff, Zhongwei Wan, Ping Luo, Min Lin, and Ngai Wong. Scaling laws with vocabulary: Larger models deserve larger vocabularies. *arXiv preprint arXiv:2407.13623*, 2024.

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

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *arXiv preprint arXiv:2402.14905*, 2024.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10, 2016.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *arXiv preprint arXiv:1910.07467*, 2019. URL `https://arxiv.org/abs/1910.07467`.

Jimmy Lei Ba. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

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

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

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

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 preprint arXiv:1803.05457*, 2018.

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

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, et al. Stable lm 2 1.6 b technical report. *arXiv preprint arXiv:2402.17834*, 2024.