MOBILLAMA: TOWARDS ACCURATE & LIGHTWEIGHT FULLY TRANSPARENT GPT

Omkar Thawakar[1*] **Ashmal Vayani**[2*] **Salman Khan**[1,3] **Hisham Cholakkal**[1]
Rao Muhammad Anwer[1,4] **Michael Felsberg**[6] **Timothy Baldwin**[1,5] **Eric P. Xing**[1]
Fahad Shahbaz Khan[1,6]
[1]Mohamed bin Zayed University of AI, [2]University of Central Florida,
[3]Australian National University, [4]Aalto University, [5]The University of Melbourne
[6]Linköping University
{omkar.thawakar@mbzuai.ac.ae}

ABSTRACT

'*Bigger the better*' has been the predominant trend in recent Large Language Models (LLMs) development. However, LLMs do not suit well for scenarios that require on-device processing, energy efficiency, low memory footprint, and response efficiency. These requisites are crucial for privacy, security, and sustainable deployment. This paper explores the '*less is more*' paradigm by addressing the challenge of designing accurate yet efficient Small Language Models (SLMs) for resource constrained devices. Our primary contribution is the introduction of an accurate and fully transparent open-source 0.5 billion (0.5B) parameter SLM, named *MobiLlama*, catering to the specific needs of resource-constrained computing with an emphasis on enhanced performance with reduced resource demands. *MobiLlama* is a SLM design that initiates from a larger model and applies a careful parameter sharing scheme to reduce both the pre-training and the deployment cost. Our work strives to not only bridge the gap in open-source SLMs but also ensures full transparency, where complete training data pipeline, training code, model weights, and over 300 checkpoints along with evaluation codes is available at : https://github.com/mbzuai-oryx/MobiLlama.

1 INTRODUCTION

Recent years have witnessed a tremendous surge in the development of Large Language Models (LLMs) with the emergence of prominent closed-source commercial models such as ChatGPT, Bard, and Claude. These LLMs exhibit surprising capabilities, typically called emergent abilities, towards solving complex tasks. Most existing popular LLMs follow a similar trend that bigger is always better, where scaling model size or data size typically provides improved model capacity and performance on downstream tasks. For instance, the recent Llama-2 70 billion (70B) model Touvron et al. (2023) is considered more favorable in different chat applications due to its effectiveness towards handling dialogues, logical reasoning, coding, compared to its 7B counterpart which is typically better suited for basic tasks such as categorization or summaries. While these LLMs demonstrate impressive performance in handling complex language tasks, a key limitation is their size and computational requirements.

Recently, Small Language Models (SLMs) have shown potential in terms of providing decent performance with emergent abilities achieved at a significantly smaller scale compared to their large-scale LLM counterparts. Modern SLMs like Microsoft's Phi-2 2.7 billion Li et al. (2023b) highlight the growing focus in the community on achieving more with less. SLMs offer advantages in terms of efficiency, cost, flexibility, and customizability. With fewer parameters, SLMs offer significant computational efficiency in terms of fast pre-training and inference with reduced memory and storage requirements. This is critical in real-world applications where efficient resource utilization is highly desired. It particularly opens up possibilities in resource-constrained computing, where the models are required to be memory efficient to operate on low-powered devices (e.g., edge). SLMs support

[*]Equal Contribution

Model	#Params	Training Time	GPU Hours	GPU memory	No. of layers	Hidden dim size
baseline1	0.54B	7.5 days	28.8K	3.2 GB	22	1024
baseline2	0.52B	7 days	26.9K	3 GB	8	2048
large-base	1.2B	12 days	46.1K	6 GB	22	2048
MobiLlama	0.52B	7 days	26.6K	3 GB	22	2048

Table 1: Our *MobiLlama* demonstrates significant efficiency and scalability improvements compared to *large-base*, *baseline1*, and *baseline2*. Evaluated on A100 GPUs with 80 GB memory, *MobiLlama* reduces GPU training hours by 42% and lowers GPU memory usage under the same design configuration. Additionally, it achieves increased model capacity (more layers and larger hidden dimensions) while maintaining comparable training costs and parameter counts to the baselines.

on-device processing that enhances privacy, security, response time, and personalization. Such an integration can lead to advanced personal assistants, cloud-independent applications, and improved energy efficiency with a reduced carbon footprint.

The limited landscape of open-source SLMs restricts transparency and accessibility, hindering the exploration of compact, efficient, and high-performing models. Addressing this gap is crucial to democratize access and foster innovation in the community by enabling a deeper understanding of SLM capabilities and limitations. To this end, we focus on designing accurate yet efficient SLMs from scratch, offering full transparency with access to training pipelines, model weights, over 300 checkpoints, and evaluation codes. While scaling down larger LLMs by reducing hidden dimensions or layers often results in inferior performance, we propose an alternative approach to develop SLMs that ensure accuracy, efficiency in on-device memory, and complete transparency.

Contributions: We introduce a SLM framework, named *MobiLlama*, with an aim to develop accurate SLMs by alleviating the redundancy in the transformer blocks. Different to the conventional SLM design where dedicated feed forward layers (FFN) are typically allocated to each transformer block, we propose to employ a shared FFN design for all the transformer blocks within SLM. Our *MobiLlama* models outperform existing SLMs under 1B parameters, with the 0.5B model achieving a 2.4% average performance gain across nine benchmarks and the 0.8B model achieving top performance using an enhanced shared-FFN transformer design. Our *MobiLlama* leveraging a shared FFN-based SLM design is accurate and maintains efficiency, while offering full transparency in terms of data pipeline, training code, model weights and extensive intermediate checkpoints along with evaluation codes.

2 METHOD

Baseline SLM Design: We first describe our baseline 0.5B SLM architecture that is adapted from recent TinyLlama Zhang et al. (2024a) and Llama-2 Touvron et al. (2023). The baseline architecture comprises N layers, where each layer consists of hidden dimensions of M and intermediate size (MLPs) of 5632. The vocabulary size is $32K$ and max. context length is C. We consider two different design choices when constructing a 0.5B model from scratch. In first design choice, named $baseline1$, the number of layer is set to $N = 22$ and hidden size of each layer is set to $M = 1024$. In second design choice, named $baseline2$, we set the number of layer to $N = 8$ and hidden size of each layer is set to $M = 2048$.

We note that both the aforementioned baseline designs struggle to strike an optimal balance between accuracy and efficiency. While a reduced size of hidden dimensions (1024) in case of $baseline1$ aids in computational efficiency, it can likely hamper the model's capacity to capture complex patterns within the data. Such a reduction in dimension can potentially lead to a bottleneck effect, where the model's ability to represent intricate relationships and nuances in the data is constrained, thereby affecting the overall accuracy. On the other hand, reducing the number of hidden layers (22 to 8), as in the $baseline2$, affects the model's depth that in turn hampers its ability to learn hierarchical representations of the language. Achieving superior performance on tasks requiring deeper linguistic comprehension and contextual analysis likely requires combining the advantages of the two aforementioned baselines. However, increasing the model capacity of $baseline1$ and $baseline2$ into a single model (22 layers and hidden dimension size of 2048) results in a significantly larger parameterized model of 1.2B with increased training cost (see Tab. 1). We name this larger model as *large-base*. Next, we present our proposed *MobiLlama* 0.5B model design that does not

Model Name	#Params	HellaSwag	Truthfulqa	MMLU	Arc_C	CrowsPairs	piqa	race	siqa	winogrande	Average
gpt-neo-125m	0.15B	30.26	45.58	25.97	22.95	61.55	62.46	27.56	40.33	51.78	40.93
tiny-starcoder	0.17B	28.17	47.68	26.79	20.99	49.68	52.55	25.45	38.28	51.22	37.86
cerebras-gpt-256m	0.26B	28.99	45.98	26.83	22.01	60.52	61.42	27.46	40.53	52.49	40.69
opt-350m	0.35b	36.73	40.83	26.02	23.55	64.12	64.74	29.85	41.55	52.64	42.22
megatron-gpt2-345m	0.38B	39.18	41.51	24.32	24.23	64.82	66.87	31.19	40.28	52.96	42.81
LiteLlama	0.46B	38.47	41.59	26.17	24.91	62.90	67.73	28.42	40.27	49.88	42.26
gpt-sw3-356m	0.47B	37.05	42.55	25.93	23.63	61.59	64.85	32.15	41.56	53.04	42.48
pythia-410m	0.51B	40.85	41.22	27.25	26.19	64.20	67.19	30.71	41.40	53.12	43.57
xglm-564m	0.56B	34.64	40.43	25.18	24.57	62.25	64.85	29.28	42.68	53.03	41.87
Lamini-GPT-LM	0.59B	31.55	40.72	25.53	24.23	63.09	63.87	29.95	40.78	47.75	40.83
MobiLlama (Ours)	0.5B	52.52	38.05	26.45	29.52	64.03	72.03	33.68	40.22	57.53	**46.00**
Lamini-GPT-LM	0.77B	43.83	40.25	26.24	27.55	66.12	69.31	37.12	42.47	56.59	45.49
MobiLlama (Ours)	0.8B	54.09	38.48	26.92	30.20	64.82	73.17	33.37	41.60	57.45	**46.67**

Table 2: State-of-the-art comparisons with existing ¡ *1B params models* on *nine* benchmarks. In case of around 0.5B model series, our *MobiLlama* achieves a substantial gain of 2.4% in terms of average performance on nine benchmarks. Further, our *MobiLlama* 0.8B model achieves an average of 46.67.

Model	HellaSwag	Truthfulqa	MMLU	Arc_C	Average
baseline1	42.44	38.16	25.12	26.18	32.97
baseline2	43.66	38.54	25.76	26.32	33.57
MobiLlama	48.42	39.36	26.56	27.88	**35.55**

Table 3: Baseline comparison on four benchmarks. Here, both the baselines and our *MobiLlama* comprise the same parameters (0.5B) and are pre-trained on 120B tokens from Amber.

Model	Load (ms)	Init (ms)	Forward-Pass (ms)
large-base	52	1896	15.7
MobiLlama-0.5B	27	642	9.3

Table 4: Latency analysis of our MobiLlama-0.5B vs. large-base using a profiler at inference time on RTX2080Ti.

reduce hidden dimension size in each layer ($baseline1$) or the total number of layers ($baseline2$), while maintaining a comparable training efficiency (see Tab. 1).

Proposed SLM Design: MobiLlama: The proposed approach, *MobiLlama*, constructs a SLM of desired sizes (e.g., 0.5B model) by first initiating from a larger model size design, *large-base*. Then, we employ a careful parameter sharing scheme to reduce the model size to a pre-defined model configuration, thereby significantly reducing the training cost. Generally, both SLMs and LLMs typically utilize a dedicated multilayer perceptron (MLP) block comprising multiple feed forward network (FFN) layers within each transformer block. In such a configuration (e.g., *large-base*), the FFN layers account for a substantial 65% of the total trainable parameters, with attention mechanisms and heads contributing 30% and 5%, respectively. As a consequence, a significant number of parameters are concentrated within the FFN layers, thereby posing challenges during pre-training with respect to computational cost and the model's ability to achieve faster convergence. To address these issues, we propose to use a sharing scheme where the FFN parameters are shared across all transformer layers within the SLM. This enables us to significantly reduce the overall trainable parameters by 60% in our *MobiLlama*, compared to the *large-base*. Such a significant parameter reduction also enables us to increase the model capacity in terms of number of layers and hidden dimension size without any substantial increase in the training cost (see Tab. 1). For additional comparative details on the design choices, please refer to Fig. 1 in the Appendix.

Evaluation Benchmarks and Metrics: For a comprehensive evaluation, we use nine benchmarks from the Open LLM Leaderboard[1]. These include: HellaSwag Zellers et al. (2019) for scenario completion and common sense reasoning, TruthfulQA Lin et al. (2021a) for factual accuracy, and MMLU Hendrycks et al. (2020) for multidisciplinary knowledge. ARC_Challenge Clark et al. (2018) tests complex reasoning, while CrowsPairs Nangia et al. (2020) evaluates bias and fairness. PIQA Bisk et al. (2020) assesses physical commonsense, Race Lai et al. (2017) measures reading comprehension, SIQA Sap et al. (2019) focuses on social reasoning, and Winogrande Sakaguchi et al. (2021) tests text disambiguation and commonsense reasoning.

[1] https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

Platform	Model	#Params (↓)	Precision	Avg Tokens/Sec (↑)	Avg Memory Consumption (↓)	Avg Battery Consumption /1k Tokens (↓)	CPU Utilization (↓)
RTX2080Ti	Llama2	7B	bf16	14.85	27793 MB	135.51 mAH	31.62%
	Phi2	2.7B	bf16	32.19	12071 MB	59.13 mAH	24.73%
	large-base	1.2B	bf16	50.61	6254 MB	18.91 mAH	18.25%
	MobiLlama	0.5B	bf16	**63.38**	**3046** MB	**8.19** mAH	**14.79**%
CPU-i7	Llama2	7B	4bit	5.96	4188 MB	73.5 mAH	49.16%
	Phi2	2.7B	4bit	22.14	1972 MB	27.36 mAH	34.92%
	large-base	1.2B	4bit	29.23	1163 MB	10.81 mAH	30.84%
	MobiLlama	0.5B	4bit	**36.32**	**799** MB	**4.86** mAH	**24.64**%
Snapdragon-685	Llama2	7B	4bit	1.193	4287 MB	10.07 mAH	77.41%
	Phi2	2.7B	4bit	2.882	1893 MB	14.61 mAH	56.82%
	large-base	1.2B	4bit	6.687	780 MB	6.00 mAH	17.15%
	MobiLlama	0.5B	4bit	**7.021**	**770** MB	**5.32** mAH	**13.02**%

Table 5: Comparison in terms of efficiency and resource consumption on different low-end hardware devices. We show the comparison on: a PC with RTX-2080Ti GPU, a laptop with i7 CPU and a smartphone with Snapdragon-685 processor. In addition to our *large-base* model, we also present the comparison with Llama2 7B and Phi2 2.7B. The different metrics measure the model's operational efficiency, model's footprint in the device's RAM and the energy efficiency of processing 1,000 tokens. Our *MobiLlama* performs favorably in terms of efficiency on these low-end hardware devices.

3 RESULTS

Baseline Comparison: We first present a comparison with the two baselines in Tab. 3) for 0.5B model series. For the baseline evaluation, we pre-train all the models on the same 120B tokens from the Amber dataset and report the results on four benchmarks: HellaSwag, TruthfulQA, MMLU, and Arc_C. Our *MobiLlama* achieves favourable performance compared to the two baselines by achieving an average score of 34.4 over the four benchmarks. We note that this performance improvement is achieved without any significant increase in the training cost (see Tab. 1), highlighting the merits of the proposed SLM design. Additional results and analysis are present in the Appendix.

State-of-the-art Comparison: We compare our *MobiLlama* 0.5B and 0.8B with existing SLMs having comparable (less than 1B) parameters: gpt-neo Black et al. (2021), tiny-starcoder Li et al. (2023a), cerebras-gpt Dey et al. (2023), opt Zhang et al. (2022), megatron-gpt-2 Shoeybi et al. (2019), LiteLlama, gpt-sw3, pythia Biderman et al. (2023), xglm Lin et al. (2021b), Lamini-LM Wu et al. (2023). Among existing methods falling around 0.5B model series category, pythia-410m achieves an average score of 43.57. Our *MobiLlama* 0.5B model achieves superior performance with an average score of 46.0, outperforming pythia-410m by 2.4% in terms of average performance on nine benchmarks. Notably, *MobiLlama* achieves superior performance on the HellaSwag benchmark which is designed to evaluate the model's capabilities in the NLP text completion task. Additionally, *MobiLlama* also performs favorably on commonsense reasoning tasks with superior results on piqa and winogrande benchmarks. Further, our *MobiLlama* 0.8B model achieves an average of 46.67.

Efficiency and Speed Gains with Shared-FFN Design: Our *MobiLlama-0.5B* leverages a shared-FFN design, reducing unique trainable parameters and enhancing efficiency compared to the large-base 1.2B model. Profiling analysis (Tab. 4) demonstrates superior inference performance, with reduced latency in loading, initialization, and forward passes. Unlike the large-base model, which requires frequent parameter loading and switching during layer transitions, the shared-FFN design enables faster processing, achieving higher token throughput and lower energy consumption, making *MobiLlama-0.5B* a more efficient and practical solution for real-world deployment.

Efficiency Comparison: We present the comparison of our model in terms of efficiency and resource consumption on various low-end hardware platforms: a PC with RTX-2080Ti GPU, a laptop with i7 CPU, and a smartphone with Snapdragon-685 processor. Tab. 5 shows the comparison of our *MobiLlama* 0.5B with *large-base* 1.2B, Llama2-7B Touvron et al. (2023) and Phi2-2.7B Li et al. (2023b) model, in terms of the average processing speed in tokens per second (Average Tokens/Sec), average memory consumption (Avg Memory Consumption) in megabytes (MB), and the average battery consumption (Average Battery Consumption/1000 Tokens) in milliampere-hours (mAH). Our *MobiLlama* performs favorably in terms of efficiency across different hardware platforms.

4 CONCLUSION

We present a fully transparent SLM, *MobiLlama*, that alleviates redundancy in the transformer block. Within *MobiLlama*, we propose to utilize a shared FFN design for all the blocks within the SLM. Our *MobiLlama* is accurate yet efficient in terms of training cost, on-device memory and storage efficiency. We evaluate *MobiLlama* on nine benchmarks, achieving favourable results compared to existing methods falling under less than 1B category. We also build a multimodal model on top of *MobiLlama* SLM to demonstrate visual reasoning capabilities. We hope that our *MobiLlama* will help accelerate research efforts towards building fully-transparent, accurate yet efficient SLMs that bridge the gap with their resource hungry LLM counterparts.

5 ACKNOWLEDGEMENT

The computations were enabled by resources provided by NAISS at Alvis partially funded by Swedish Research Council through grant agreement no. 2022-06725, LUMI hosted by CSC (Finland) and LUMI consortium, and by Berzelius resource provided by the Knut and Alice Wallenberg Foundation at the NSC. This work was partially supported by the Swedish Research Council (2022-04266), starting grant (2016-05543), and from KAW (DarkTree project; 2024.0076).

REFERENCES

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*, 2024.

Srinadh Bhojanapalli, Ayan Chakrabarti, Andreas Veit, Michal Lukasik, Himanshu Jain, Frederick Liu, Yin-Wen Chang, and Sanjiv Kumar. Leveraging redundancy in attention with reuse transformers. *arXiv preprint arXiv:2110.06821*, 2021.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL `https://doi.org/10.5281/zenodo.5297715`. If you use this software, please cite it using these metadata.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*, 2022.

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.

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

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

Nolan Dey, Gurpreet Gosal, Hemant Khachane, William Marshall, Ribhu Pathria, Marvin Tom, Joel Hestness, et al. Cerebras-gpt: Open compute-optimal language models trained on the cerebras wafer-scale cluster. *arXiv preprint arXiv:2304.03208*, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, pp. 291–326. Chapman and Hall/CRC, 2022.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, A. Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Daniel Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hanna Hajishirzi. Olmo: Accelerating the science of language models. *arXiv preprint*, 2024. URL https://api.semanticscholar.org/CorpusID:267365485.

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.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *The Journal of Machine Learning Research*, 22(1):10882–11005, 2021.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6700–6709, 2019.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

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. Starcoder: may the source be with you! 2023a.

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

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

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona T. Diab, Veselin Stoyanov, and Xian Li. Few-shot learning with multilingual language models. *CoRR*, abs/2112.10668, 2021b. URL https://arxiv.org/abs/2112.10668.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023a.

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

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

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R Bowman. Crows-pairs: A challenge dataset for measuring social biases in masked language models. *arXiv preprint arXiv:2010.00133*, 2020.

Bowen Pan, Rameswar Panda, Rogerio Schmidt Feris, and Aude Jeanne Oliva. Interpretability-aware redundancy reduction for vision transformers, June 22 2023. US Patent App. 17/559,053.

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. URL https://arxiv.org/abs/2306.01116.

Telmo Pessoa Pires, António V Lopes, Yannick Assogba, and Hendra Setiawan. One wide feedforward is all you need. *arXiv preprint arXiv:2309.01826*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

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.

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

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

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8317–8326, 2019.

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

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*, 2023.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *CoRR*, abs/2304.14402, 2023. URL https://arxiv.org/abs/2304.14402.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

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.

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

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

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

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*, 2023.

A APPENDIX

In this appendix, we provide additional details and insights to complement the main content of the paper. First, we present an overview of related work, offering context to situate our contributions within the broader landscape of efficient SLM development. Next, we delve deeper into the architectural design of *MobiLlama*, highlighting the key innovations and design principles that contribute to its efficiency and performance. Finally, we provide empirical evidence demonstrating how *MobiLlama* outperforms conventional SLMs across multiple benchmarks, showcasing its superiority in terms of accuracy, efficiency, and scalability. These additional details further emphasize the robustness and impact of *MobiLlama* as a state-of-the-art solution in efficient language model design.

B RELATED WORK

While LLMs have gained tremendous popularity Zhao et al. (2023), one of their key limitations is the size and computational requirements both during pre-training and deployment. Another issue is limited availability of fully transparent opens-source LLMs that provide complete access to data pipeline, training code along with checkpoints and evaluation protocols. Prior works explore making several components of LLM framework efficient such as, attention mechanism Dao (2023) and optimization strategies Loshchilov & Hutter (2017). Further, existing efforts also include exploring post-training sparsification schemes Ashkboos et al. (2024) or quantization Hoefler et al. (2021); Zhu et al. (2023); Xiao et al. (2023) of computationally expensive LLM. In several cases, such a post-hoc sparsification can reduce the performance of LLMs with more on-device memory consumption, compared to a SLM trained from scratch. Further, these techniques typically employ LLMs with limited transparency and accessibility.

Recently, designing SLMs from scratch have gained attention Biderman et al. (2023); Wu et al. (2023); Zhang et al. (2024a); Li et al. (2023a); Lin et al. (2021b); Shoeybi et al. (2019); Zhang et al. (2022). SLMs have shown potential as an alternative especially in case of limited pre-training compute as well as deployment in resource-constrained environments (e.g., edge devices). Further, SLMs can support on-device processing which in turn can enhance security, privacy, response efficiency, and personalization. Here, we strive to construct fully transparent accurate yet computationally efficient SLMs by maintaining the model's capacity to capture complex patterns and relationships in data while reducing the redundancy often present in the parameters of SLMs. Prior works Frantar et al. (2022); Gholami et al. (2022); Pires et al. (2023); Pan et al. (2023); Bhojanapalli et al. (2021) exploring alleviating redundancy in transformer design either focusing on the attention mechanism or on the single feed-forward layer in BERT style architectures. Different from these approaches, we explore alleviating the redundancy in the SLM architectures with an LLM objective function by focusing on the sharing mechanism of MLP blocks having multiple feed-forward network (FFN) layers.

Our design thoughtfully maintains the model's capacity to capture complex patterns and relationships in data, ensuring that the reduction in size does not detrimentally affect the model's performance. Instead, we leverage the redundancy often present in the parameters of large models, identifying and consolidating these overlaps through shared FFN layers.

Several existing works explore post-training sparsification schemes Ashkboos et al. (2024) or quantization (cite) of computationally expensive LLM. In several cases, such a post-hoc sparisification can dramatically reduce the performance of LLMs with more on-device memory consumption, compared to a SLM trained from scratch. Further, these techniques typically employ LLMs with limited transparency and accessibility.

The quest for efficiency in Large Language Models (LLMs) has garnered significant attention in recent years, leading to a plethora of innovative designs aimed at reducing computational demands while preserving, or even enhancing, model performance. A notable direction in this field is the development of models that leverage parameter-sharing mechanisms and sparsity to achieve lower memory footprints and faster inference times. For instance, the introduction of models like GPT-NeoX-20B Black et al. (2022) and the subsequent adaptations for mobile deployment, exemplify the industry's shift towards creating LLMs that are not only powerful but also accessible for a wide range of applications.

Figure 1: Illustrative comparison of our *MobiLlama* with the two baselines. For each case, we show two transformer blocks denoted by different self-attention layers. In the case of both *baseline1* and *baseline2*, a dedicated MLP block comprising three FFN layers is utilized for each transformer layer. In contrast, our *MobiLlama* utilizes a single MLP block (highlighted by the same color) that is shared across different transformer layers. This enables to increase the capacity of the network in terms of layers and hidden dimension size without any significant increase in the total number of trainable parameters.

Another pivotal area of research focuses on optimizing attention mechanisms, a core component of transformer-based LLMs. Techniques such as the flash-attention algorithm Dao (2023) have been instrumental in speeding up the training and inference phases of LLMs by optimizing how models attend to different parts of the input data. Similarly, the exploration of alternative optimization strategies, such as those presented in the work on AdamW Loshchilov & Hutter (2017), showcases the ongoing efforts to refine the training processes of LLMs to achieve better efficiency without sacrificing model quality. These efforts are complemented by the use of advanced hardware and distributed training frameworks, allowing researchers to push the boundaries of what's possible with LLMs in terms of scale and efficiency. Collectively, these advancements highlight a vibrant and rapidly evolving landscape in the development of efficient designs for LLMs, underscoring the research community's commitment to making these powerful models more sustainable and broadly accessible.

C TOWARDS FULLY TRANSPARENT MOBILLAMA

As discussed earlier, fully transparent open-source SLM development is desired to foster a more inclusive, data/model provenance, and reproducible collaborative SLM research development environment. To this end, we present here pre-training dataset and processing details, architecture design configuration with training details, evaluation benchmarks and metrics. In addition, we will publicly release complete training and evaluation codes along with intermediate model checkpoints.

Architecture Design: Our *MobiLlama* 0.5B comprises a hidden size of 2048, an intermediate size of 5632 in its MLPs, and operates with 32 attention heads across 22 hidden layers. It is designed to handle sequences up to 2048 tokens long, supported by a vocabulary size of 32,000. The precision in normalization is ensured by an RMSNorm epsilon of $1e^{-6}$ to obtain a more stable training. We utilize RoPE (Rotary Positional Embedding) Su et al. (2024) to encode positional information in our *MobiLlama*. Similar to Zhang et al. (2024a), we employ a combination of Swish and Gated Linear Units together as activation functions. We also derive a 0.8B version from our *MobiLlama* by widening the shared FFN design. Compared to the 0.5B model, our 0.8B design increases the hidden dimension size to 2532 and the intermediate size to 11,080 while the rest of the configuration is same.

Pre-training Dataset and Processing: For pre-training, we use 1.2T tokens from LLM360 Amber dataset Liu et al. (2023b). The Amber dataset provides a rich and varied linguistic landscape having different text types, topics, and styles.

Tab. 6 shows the data mix from Amber dataset gathered from various sources. The dataset's comprehensive nature supports the model's ability to grasp subtle distinction of language, enhancing

Subset	Tokens (Billion)
Arxiv	30.00
Book	28.86
C4	197.67
Refined-Web	665.01
StarCoder	291.92
StackExchange	21.75
Wikipedia	23.90
Total	1259.13

Hyperparameter	Value
Number Parameters	$0.5B$
Hidden Size	2048
Intermediate Size (in MLPs)	5632
Number of Attention Heads	32
Number of Hidden Layers	22
RMSNorm ϵ	$1e^{-6}$
Max Seq Length	2048
Vocab Size	32000

Table 6: Datasets and architecture details. (Left) Data mix in Amber-Dataset. (Right) *MobiLlama* architecture and hyperparameters.

its performance on a variety of tasks, from language understanding to content generation. Each subset within this data compilation plays a pivotal role in enhancing the language learning capabilities of Large Language Models (LLMs):

- Arxiv (30 Billion Tokens): This subset, drawn from the repository of scientific papers, provides complex, domain-specific language and technical terminology, enriching the understanding of academic prose.
- Book (28.9 Billion Tokens): This subset comprises tokens from a broad range of literature with diverse narrative styles, cultural contexts, and rich vocabulary, deepening the grasp of storytelling and language nuances.
- C4 (197.7 Billion Tokens): The Colossal Clean Crawled Corpus (C4) offers a vast and cleaned selection of web text, providing a broad linguistic foundation that includes various registers, styles, and topics.
- Refined-Web (665 Billion Tokens): This subset, likely a curated web crawl, offers the model exposure to contemporary, informal, and varied internet language, enhancing the relevance and applicability to modern communication.
- StarCoder (291.9 Billion Tokens): The StarCoder is a vast collection used for code understanding featuring 783GB of code across 86 programming languages. It includes GitHub issues, Jupyter notebooks, and commits, totaling approximately 250 billion tokens. These are meticulously cleaned and de-duplicated for training efficiency.
- StackExchange (21.8 Billion Tokens): From the network of Q&A websites, this subset aids the model in learning question-answering formats and technical discussions across diverse topics.
- Wikipedia (23.9 Billion Tokens): As an encyclopedia collection, it offers well-structured and factual content that helps the model to learn encyclopedic knowledge and formal writing styles.

From the above-mentioned subsets, Arxiv, Book, C4, StackExchange and Wikipedia are sourced from RedPajama-v1 Computer (2023). The Amber dataset uses RefinedWeb Penedo et al. (2023) data to replace common_crawl subset of RedPajama-v1. These subsets amount to 1259.13 billion tokens. offering LLMs a rich and diverse linguistic diet that is essential for developing a broad, deep understanding of human language and its many applications.

Initially, raw data sourced from the above sources is tokenized using Huggingface LLaMA tokenizer Touvron et al. (2023). Subsequently, these tokens are organized into sequences with each containing 2048 tokens. To manage data, these sequences are merged to the token sequences and divided the amalgamated dataset into 360 distinct segments. Each data segment, structured as a jsonl file, carries an array of token IDs along with a source identifier that denotes the originating dataset. Each data sample is designed to have 2049 tokens.

Pretraining Details: For pre-training of our *MobiLlama*, we use a public cluster having 20 GPU nodes each equipped with 8 NVIDIA A100 GPUs with 80 GB memory each and 800 Gbps interconnect for model training. Each GPU is interconnected through 8 NVLink links, complemented by a cross-node connection configuration of 2 port 200 Gb/sec (4$\times$ HDR) InfiniBand, optimizing the model's training process. To further enhance the training efficiency, we employ flash-attention mechanism and follow

the pre-training hyper-parameters established by the LLaMA Touvron et al. (2023) model. Our *MobiLlama* model's training is performed using the AdamW optimizer, leveraging hyperparameters $\beta_1 = 0.9, \beta_2 = 0.95$, with an initial learning rate of $\eta = 3e^{-4}$. This rate follows a cosine learning rate schedule, tapering to a final rate of $\eta = 3e^{-5}$. We further incorporate a weight decay of 0.1 and apply gradient clipping at 1.0 with a warm-up period over $2,000$ steps. Adapting to our hardware configuration of 20 GPU nodes, we optimize the pre-training batch size to 800 (160×5), achieving a throughput of approximately $14k$-$15k$ tokens per second on a single GPU. During our model pre-training, we save intermediate checkpoints after every 3.3B tokens which will be publicly released.

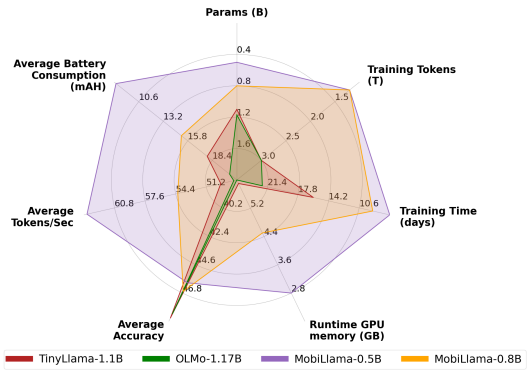

Figure 2: Comparison of our *MobiLlama* 0.5B and 0.8B models with recent OLMo-1.17B Groeneveld et al. (2024) and TinyLlama-1.1B Zhang et al. (2024a) in terms of pre-training tokens, pre-training time and memory, model parameters, overall accuracy across nine benchmarks, and on-device efficiency (average battery consumption and average tokens/second on a PC with RTX2080Ti). Our *MobiLlama* achieves comparable accuracy while requiring significantly fewer pre-training data (1.2T tokens vs. 3T tokens), less pre-training time and GPU memory, along with being efficient for deployment on resource-constrained devices.

D Additional Results

We present additional empirical evidence demonstrating that our *MobiLlama* outperforms conventional SLM design schemes in pre-training from scratch. The *MobiLlama* 0.5B model achieves a 2.4% improvement in average performance across nine benchmarks compared to the best existing 0.5B SLMs in the literature. Additionally, we extend this design to develop a 0.8B SLM by adopting a wider shared-FFN scheme within the transformer blocks, achieving state-of-the-art performance among SLMs with fewer than 1B parameters. Furthermore, we enhance our SLM by building multimodal models to demonstrate advanced visual perception and reasoning capabilities. Fig. 2 highlights the competitive advantages of our *MobiLlama* compared to larger, fully transparent SLMs in terms of accuracy, pre-training complexity, and onboard deployment cost.

Evaluating Large-base Model: As discussed earlier, we strive to develop fully transparent models for democratization of SLMs and fostering future research. To this end, we compare our *large-base* 1.2B with existing fully transparent SLMs falling within the less than 2B category. These existing SLMs are: Boomer, pythia Biderman et al. (2023), Falcon-RW Penedo et al. (2023), TinyLlama Zhang et al. (2024b), OLMo Groeneveld et al. (2024), cerebras-gpt Dey et al. (2023), Lamini-LM Wu et al. (2023), opt Zhang et al. (2022) and gpt-neo Black et al. (2021). Tab. 7 shows that compared to recent OLMo and TinyLlama that are pre-trained on a larger dataset of 3T tokens, our *large-base* 1.2B model pre-trained on 1.2T tokens achieves favourable results with an average score of 49.06 over nine benchmarks. We hope that our *large-base* model will serve as a solid baseline and help ease future research in SLM development.

Post-Training Sparsification: We further perform an efficiency comparison to a recent post-training sparsification scheme Ashkboos et al. (2024), where each weight matrix is substituted with a smaller (dense) matrix, thereby reducing dimensions of the embeddings in the model. In such a scheme, the parameters of the original LLM are reduced significantly up to 70% followed by post-slicing fine-tuning using a dataset such as WikiText-2 Merity et al. (2016). Tab. 8 shows the comparison of our *MobiLlama* with existing LLMs (e.g., Llama-2-7B, OPT-6.7B) on four benchmarks following Ashkboos et al. (2024). Our *MobiLlama* 0.5B and 0.8B models perform favorably against representative LLMs, with an average score of 53.72 computed over four benchmarks. These results highlight the potential of designing new fully transparent SLMs that can achieve comparable capabilities of their larger sliced model counterparts.

Model	#Params	HellaSwag	Truthfulqa	MMLU	Arc_C	CrowsPairs	piqa	race	siqa	winogrande	Average
Boomer	1B	31.62	39.42	25.42	22.26	61.26	57.99	28.99	40.32	50.98	39.80
Pythia-Dedup	1B	49.63	38.92	24.29	29.09	67.11	70.23	32.44	42.63	53.98	45.36
Falcon-RW	1B	63.12	35.96	25.36	35.06	69.04	74.10	36.07	40.23	61.88	48.98
TinyLlama	1.1B	60.22	37.59	26.11	33.61	70.60	73.28	36.45	41.65	59.18	48.74
OLMo	1.2B	62.50	32.94	25.86	34.45	69.59	73.70	36.74	41.14	58.90	48.42
Cerebras-GPT	1.3B	38.51	42.70	26.66	26.10	63.67	66.75	30.33	42.42	53.59	43.41
Lamini	1.3B	38.05	36.43	28.47	26.62	64.62	67.89	33.39	43.19	50.59	43.25
OPT	1.3B	54.50	38.67	24.63	29.6	70.70	72.47	34.16	42.47	59.74	47.43
GPT-NEO	1.3B	48.49	39.61	24.82	31.31	65.67	71.05	34.06	41.81	57.06	45.98
Pythia-Deduped	1.4B	55.00	38.63	25.45	32.59	67.33	72.68	34.64	42.68	56.90	47.32
Qwen-2	1.8B	26.99	47.30	25.83	24.57	50.64	51.19	24.59	36.28	49.72	37.45
large-base	1.2B	62.99	35.90	24.79	34.55	68.49	75.57	35.31	41.96	62.03	**49.06**

Table 7: Comprehensive comparisons with existing ¡ *2B params fully open-source LLM models* on *9* benchmarks. Our 1.2B *large-base* model pre-trained on 1.2T tokens achieves superior performance compared to both the recent OLMo 1.17B model Groeneveld et al. (2024) and TinyLlama 1.1B model Zhang et al. (2024a), which are pre-trained on a substantially larger data of 3T tokens.

Model	#Slice	#Params	HellaS	Arc_C	piqa	wino	Average
OPT-1.3B	30%	0.91B	39.81	25.77	60.77	54.7	45.26
OPT-6.7B	30%	4.69B	54.56	29.01	68.61	60.69	53.21
Llama-2-7B	30%	4.9B	49.62	31.23	63.55	61.33	51.43
Phi2-2.7B	30%	1.89B	47.56	30.29	65.94	63.14	51.73
MobiLlama	Dense	0.5B	52.52	29.52	72.03	57.53	52.90
	Dense	0.8B	54.09	30.20	73.17	57.45	53.72

Table 8: Comparison on *4 open LLM benchmarks* when parameters are sliced down to 30% using Wiki2Text dataset, following Ashkboos et al. (2024).

Model	GQA	SQA	TextQA	MME
MobileLLM-1.7B	56.1	54.7	41.5	1196.2
MobileLLM-3B	59.0	61.0	47.5	1288.9
MobiLlama-V-0.5B	51.0	52.2	32.4	1032.1
MobiLlama-V-0.8B	58.5	53.1	41.4	1191.9

Table 9: Quantitative performance of our multimodal design, *MobiLlama-V* 0.8B, on different benchmarks. Our model achieves favorable performance across different benchmarks. Comparative analysis of our models against the MobileVLM, across a range of benchmarks designed to evaluate various aspects of language model proficiency. For the LLM size category, our models demonstrate impressive efficiency, with the 0.5B and 0.8B variants delivering competitive performance despite their smaller sizes. The benchmarks include General Question Answering (GQA), Specific Question Answering (SQA), Text-based Question Answering (TextQA), POPE, Multi-Modal Evaluation (MME), and Multi-Modal Benchmark (MMB), with our 0.8B model particularly excelling in GQA and POPE. These results showcase the capability of our models to achieve high performance in complex tasks, highlighting the effectiveness of our architectural optimizations even with a reduced parameter count.

Multimodal MobiLlama: We further build a multimodal model on top of our *MobiLlama* by combining it with a vision encoder to develop a general-purpose visual assistant having visual reasoning capabilities. Our multimodal model, *MobiLlama-V*, is trained by bridging the visual encoder of CLIP Radford et al. (2021) with the language decoder of our *MobiLlama*, and fine-tuning it in an end-to-end fashion on a 665k vision-language instruction set Liu et al. (2023a). We conduct evaluation on GQA Hudson & Manning (2019), SQA Lu et al. (2022), TextQA Singh et al. (2019), and MME Fu et al. (2023). Tab. 9 shows the performance of *MobiLlama-V* 0.8B model.

Limitation and Future Direction: A potential direction is to further improve *MobiLlama* for enhanced context comprehension and understanding subtlety of linguistic nuances. Domain-specific expertise of the model can also be explored (e.g., healthcare). While *MobiLlama* offers a fully transparent SLM framework, a follow-up study to understand any misrepresentations and biases is desired to improve model's robustness. While MobiLlama marks a significant stride in the development of lightweight, efficient language models, it is not without limitations.