The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits

Shuming Ma* **Hongyu Wang*** **Lingxiao Ma** **Lei Wang** **Wenhui Wang**
Shaohan Huang **Li Dong** **Ruiping Wang** **Jilong Xue** **Furu Wei◇**
https://aka.ms/GeneralAI

Abstract

Recent research, such as BitNet [WMD+23], is paving the way for a new era of 1-bit Large Language Models (LLMs). In this work, we introduce a 1-bit LLM variant, namely **BitNet b1.58**, in which every single parameter (or weight) of the LLM is ternary {-1, 0, 1}. It matches the full-precision (i.e., FP16 or BF16) Transformer LLM with the same model size and training tokens in terms of both perplexity and end-task performance, while being significantly more cost-effective in terms of latency, memory, throughput, and energy consumption. More profoundly, the 1.58-bit LLM defines a new scaling law and recipe for training new generations of LLMs that are both high-performance and cost-effective. Furthermore, it enables a new computation paradigm and opens the door for designing specific hardware optimized for 1-bit LLMs.

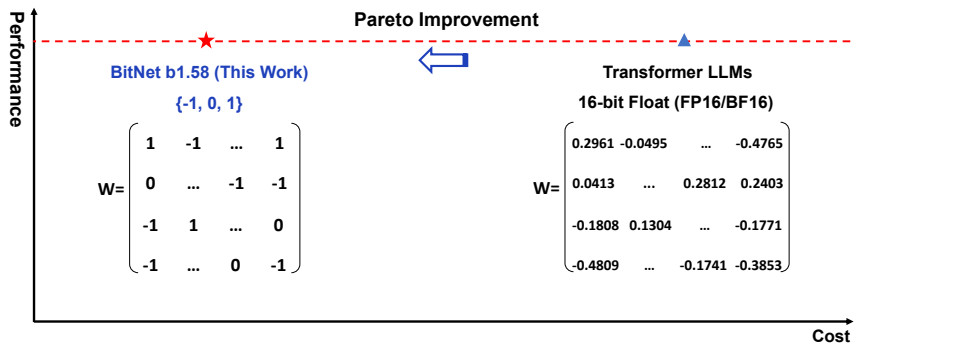

Figure 1: 1-bit LLMs (e.g., BitNet b1.58) provide a Pareto solution to reduce inference cost (latency, throughput, and energy) of LLMs while maintaining model performance. The new computation paradigm of BitNet b1.58 calls for actions to design new hardware optimized for 1-bit LLMs.

* Equal contribution. ◇ Corresponding author. S. Ma, L. Ma, L. Wang, W. Wang, S. Huang, L. Dong, J. Xue, F. Wei are with Microsoft Research. H. Wang and R. Wang are with University of Chinese Academy of Sciences.

1 The Era of 1-bit LLMs

In recent years, the field of AI has seen a rapid growth in the size and capabilities of Large Language Models (LLMs). These models have demonstrated remarkable performance in a wide range of natural language processing tasks, but their increasing size has posed challenges for deployment and raised concerns about their environmental and economic impact due to high energy consumption. One approach to address these challenges is to use post-training quantization to create low-bit models for inference [XLS+23, FAHA23, CCKS23, TCS+24]. This technique reduces the precision of weights and activations, significantly reducing the memory and computational requirements of LLMs. The trend has been to move from 16 bits to lower bits, such as 4-bit variants [FAHA23, LTT+23]. However, post-training quantization is sub-optimal, even though it is widely used in industry LLMs.

Recent work on 1-bit model architectures, such as BitNet [WMD+23], presents a promising direction for reducing the cost of LLMs while maintaining their performance. Vanilla LLMs are in 16-bit floating values (i.e., FP16 or BF16), and the bulk of any LLMs is matrix multiplication. Therefore, the major computation cost comes from the floating-point addition and multiplication operations. In contrast, the matrix multiplication of BitNet only involves integer addition, which saves orders of energy cost for LLMs. As the fundamental limit to compute performance in many chips is power, the energy savings can also be translated into faster computation.

In addition to computation, the process of transferring model parameters from DRAM to the memory of an on-chip accelerator (e.g., SRAM) can be expensive during inference. There have been attempts to enlarge SRAM to improve throughput, but this introduces significantly higher costs than DRAM. Compared to full-precision models, 1-bit LLMs have a much lower memory footprint from both a capacity and bandwidth standpoint. This can significantly reduce the cost and time of loading weights from DRAM, leading to faster and more efficient inference.

In this work, we introduce a significant 1-bit LLM variant called **BitNet b1.58**, where every parameter is ternary, taking on values of {-1, 0, 1}. We have added an additional value of 0 to the original 1-bit BitNet, resulting in 1.58 bits in the binary system. BitNet b1.58 retains all the benefits of the original 1-bit BitNet, including its new computation paradigm, which requires almost no multiplication operations for matrix multiplication and can be highly optimized. Additionally, it has the same energy consumption as the original 1-bit BitNet and is much more efficient in terms of memory consumption, throughput and latency compared to FP16 LLM baselines. Furthermore, BitNet b1.58 offers two additional advantages. Firstly, its modeling capability is stronger due to its explicit support for feature filtering, made possible by the inclusion of 0 in the model weights, which can significantly improve the performance of 1-bit LLMs. Secondly, our experiments show that BitNet b1.58 can match full precision (i.e., FP16) baselines in terms of both perplexity and end-task performance, starting from a 3B size, when using the same configuration (e.g., model size, training tokens, etc.).

2 BitNet b1.58

BitNet b1.58 is based on the BitNet architecture, which is a Transformer that replaces *nn.Linear* with *BitLinear*. It is trained from scratch, with 1.58-bit weights and 8-bit activations. Compared to the original BitNet, it introduces some modifications that we summarize below.

Quantization Function. To constrain the weights to -1, 0, or +1, we adopt an *absmean* quantization function. It first scales the weight matrix by its average absolute value, and then round each value to the nearest integer among {-1, 0, +1}:

$$\widetilde{W} = \text{RoundClip}(\frac{W}{\gamma + \epsilon}, -1, 1), \tag{1}$$

$$\text{RoundClip}(x, a, b) = \max(a, \min(b, \text{round}(x))), \tag{2}$$

$$\gamma = \frac{1}{nm} \sum_{ij} |W_{ij}|. \tag{3}$$

The quantization function for activations follows the same implementation in BitNet, except that we do not scale the activations before the non-linear functions to the range $[0, Q_b]$. Instead, the

Compare to absmax in BitNet paper for inputs X
For Weights it was:

$$\widetilde{W} = \text{Sign}(W - \alpha),$$

$$\text{Sign}(W_{ij}) = \begin{cases} +1, & \text{if } W_{ij} > 0, \\ -1, & \text{if } W_{ij} \leq 0, \end{cases}$$

$$\alpha = \frac{1}{nm} \sum_{ij} W_{ij}$$

Models	Size	Memory (GB)↓	Latency (ms)↓	PPL↓
LLaMA LLM	700M	2.08 (1.00x)	1.18 (1.00x)	12.33
BitNet b1.58	700M	0.80 (2.60x)	0.96 (1.23x)	12.87
LLaMA LLM	1.3B	3.34 (1.00x)	1.62 (1.00x)	11.25
BitNet b1.58	1.3B	1.14 (2.93x)	0.97 (1.67x)	11.29
LLaMA LLM	3B	7.89 (1.00x)	5.07 (1.00x)	10.04
BitNet b1.58	3B	**2.22 (3.55x)**	**1.87 (2.71x)**	**9.91**
BitNet b1.58	3.9B	**2.38 (3.32x)**	**2.11 (2.40x)**	**9.62**

Table 1: Perplexity as well as the cost of BitNet b1.58 and LLaMA LLM.

Models	Size	ARCe	ARCc	HS	BQ	OQ	PQ	WGe	Avg.
LLaMA LLM	700M	54.7	23.0	37.0	60.0	20.2	68.9	54.8	45.5
BitNet b1.58	700M	51.8	21.4	35.1	58.2	20.0	68.1	55.2	44.3
LLaMA LLM	1.3B	56.9	23.5	38.5	59.1	21.6	70.0	53.9	46.2
BitNet b1.58	1.3B	54.9	24.2	37.7	56.7	19.6	68.8	55.8	45.4
LLaMA LLM	3B	62.1	25.6	43.3	61.8	24.6	72.1	58.2	49.7
BitNet b1.58	3B	**61.4**	**28.3**	**42.9**	**61.5**	**26.6**	**71.5**	**59.3**	**50.2**
BitNet b1.58	3.9B	**64.2**	**28.7**	**44.2**	**63.5**	**24.2**	**73.2**	**60.5**	**51.2**

Table 2: Zero-shot accuracy of BitNet b1.58 and LLaMA LLM on the end tasks.

activations are all scaled to $[-Q_b, Q_b]$ per token to get rid of the zero-point quantization. This is more convenient and simple for both implementation and system-level optimization, while introduces negligible effects to the performance in our experiments.

LLaMA-alike Components. The architecture of LLaMA [TLI+23, TMS+23] has been the de-facto backbone for open-source LLMs. To embrace the open-source community, our design of BitNet b1.58 adopts the LLaMA-alike components. Specifically, it uses RMSNorm [ZS19], SwiGLU [Sha20], rotary embedding [SAL+24], and removes all biases. In this way, BitNet b1.58 can be integrated into the popular open-source software (e.g., Huggingface, vLLM [KLZ+23], and llama.cpp[2]) with minimal efforts.

3 Results

We compared BitNet b1.58 to our reproduced FP16 LLaMA LLM in various sizes. To ensure a fair comparison, we pre-trained the models on the RedPajama dataset [Com23] for 100 billion tokens. We evaluated the zero-shot performance on a range of language tasks, including ARC-Easy [YBS19], ARC-Challenge [YBS19], Hellaswag [ZHB+19], Winogrande [SBBC20], PIQA [BZB+19], OpenbookQA [MCKS18], and BoolQ [CLC+19]. We also reported the validation perplexity on the WikiText2 [MXBS16] and C4 [RSR+19] datasets.

We compared the runtime GPU memory and latency of both LLaMA LLM and BitNet b1.58. The results were measured using the FasterTransformer[3] codebase, which is well-optimized for LLM inference latency on GPU devices. The 2-bit kernel from Ladder [WMC+23] is also integrated for BitNet b1.58. We reported the time per output token, as it is the major cost for inference.

Table 1 summarizes the perplexity and the cost for BitNet b1.58 and LLaMA LLM. It shows that BitNet b1.58 starts to match full precision LLaMA LLM at 3B model size in terms of perplexity, while being 2.71 times faster and using 3.55 times less GPU memory. In particular, BitNet b1.58 with a 3.9B model size is 2.4 times faster, consumes 3.32 times less memory, but performs significantly better than LLaMA LLM 3B.

[2] https://github.com/ggerganov/llama.cpp
[3] https://github.com/NVIDIA/FasterTransformer

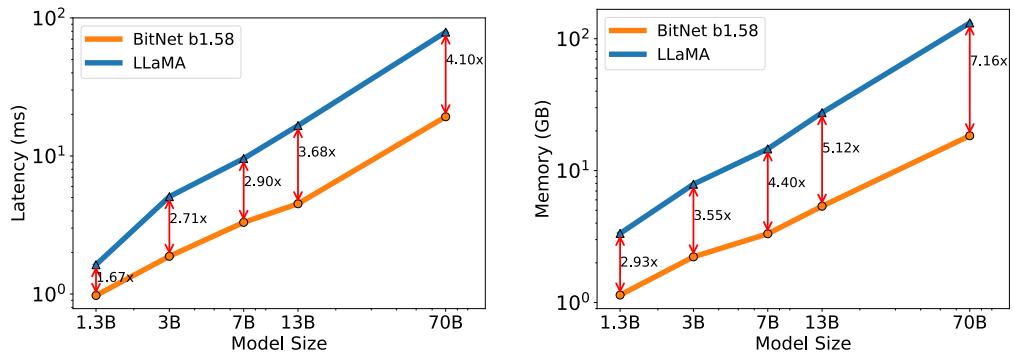

Figure 2: Decoding latency (Left) and memory consumption (Right) of BitNet b1.58 varying the model size.

Models	Size	Max Batch Size	Throughput (tokens/s)
LLaMA LLM	70B	16 (1.0x)	333 (1.0x)
BitNet b1.58	70B	**176 (11.0x)**	**2977 (8.9x)**

Table 3: Comparison of the throughput between BitNet b1.58 70B and LLaMA LLM 70B.

Why even train full precision anymore ???

Table 2 reports the detailed results of the zero-shot accuracy on the end tasks. We followed the pipeline from *lm-evaluation-harness*[4] to perform the evaluation. The results show that the performance gap between BitNet b1.58 and LLaMA LLM narrows as the model size increases. More importantly, BitNet b1.58 can match the performance of the full precision baseline starting from a 3B size. Similar to the observation of the perplexity, the end-task results reveal that BitNet b1.58 3.9B outperforms LLaMA LLM 3B with lower memory and latency cost. This demonstrates that BitNet b1.58 is a Pareto improvement over the state-of-the-art LLM models.

Memory and Latency We further scaled up the model size to 7B, 13B, and 70B and evaluated the cost. Figure 2 illustrates the trends of latency and memory, showing that the speed-up increases as the model size scales. In particular, BitNet b1.58 70B is 4.1 times faster than the LLaMA LLM baseline. This is because the time cost for *nn.Linear* grows with the model size. The memory consumption follows a similar trend, as the embedding remains full precision and its memory proportion is smaller for larger models. Both latency and memory were measured with a 2-bit kernel, so there is still room for optimization to further reduce the cost.

Energy We also estimate the arithmetic operations energy consumption of both BitNet b1.58 and LLaMA LLM. We focus mainly on the calculation for matrix multiplication, since it contributes the most to the cost of LLMs. Figure 3 illustrates the composition of the energy cost. The majority of BitNet b1.58 is INT8 addition calculation, while LLaMA LLM consists of both FP16 addition and FP16 multiplication. According to the energy model in [Hor14, ZZL22], BitNet b1.58 saves 71.4 times arithmetic operations energy consumption for matrix multiplication on 7nm chips. We further reported the end-to-end energy cost for models with 512 tokens. Our results show that as the model size scales, BitNet b1.58 becomes increasingly more efficient in terms of energy consumption compared to the FP16 LLaMA LLM baseline. This is due to the fact that the percentage of *nn.Linear* grows with the model size, while the cost from other components is smaller for larger models.

Throughput We compare the throughput of BitNet b1.58 and LLaMA LLM with 70B parameters on two 80GB A100 cards, using pipeline parallelism [HCB+19] so that LLaMA LLM 70B could be run on the devices. We increased the batch size until the GPU memory limit was reached, with a sequence length of 512. Table 3 shows that BitNet b1.58 70B can support up to 11 times the batch size of LLaMA LLM, resulting an 8.9 times higher throughput.

[4] https://github.com/EleutherAI/lm-evaluation-harness

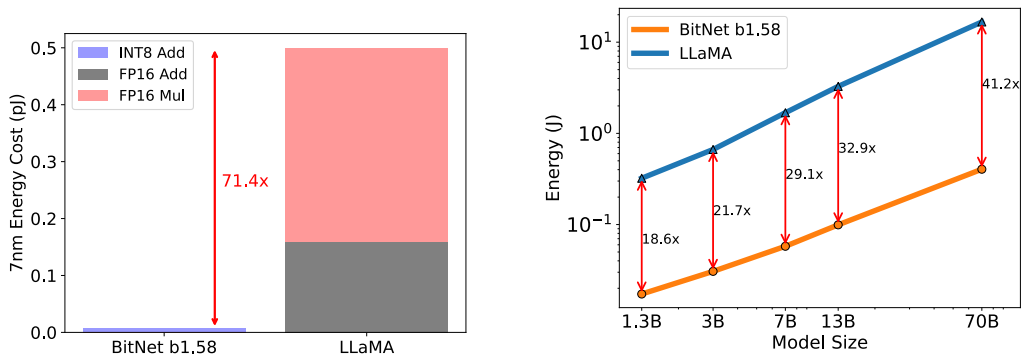

Figure 3: Energy consumption of BitNet b1.58 compared to LLaMA LLM at 7nm process nodes. On the left is the components of arithmetic operations energy. On the right is the end-to-end energy cost across different model sizes.

Models	Tokens	Winogrande	PIQA	SciQ	LAMBADA	ARC-easy	Avg.
StableLM-3B	2T	64.56	76.93	90.75	66.09	67.78	73.22
BitNet b1.58 3B	2T	**66.37**	**78.40**	**91.20**	**67.63**	**68.12**	**74.34**

Table 4: Comparison of BitNet b1.58 with StableLM-3B with 2T tokens.

BitNet b1.58 is enabling a new scaling law with respect to model performance and inference cost. As a reference, we can have the following equivalence between different model sizes in 1.58-bit and 16-bit based on the results in Figure 2 and 3.

- 13B BitNet b1.58 is more efficient, in terms of latency, memory usage and energy consumption, than 3B FP16 LLM.
- 30B BitNet b1.58 is more efficient, in terms of latency, memory usage and energy consumption, than 7B FP16 LLM.
- 70B BitNet b1.58 is more efficient, in terms of latency, memory usage and energy consumption, than 13B FP16 LLM.

Training with 2T Tokens The number of training tokens is a crucial factor for LLMs. To test the scalability of BitNet b1.58 in terms of tokens, we trained a BitNet b1.58 model with 2T tokens following the data recipe of StableLM-3B [TBMR], which is the state-of-the-art open-source 3B model. Both models were evaluated on a benchmark that consists of Winogrande [SBBC20], PIQA [BZB+19], SciQ [WLG17], LAMBADA [PKL+16], and ARC-easy [YBS19]. We reported the zero-shot accuracy in Table 4. For tasks measured with accuracy and normalized accuracy, we take the average of the two. The results of StableLM 3b at 2T tokens are taken directly from its technical report. Our findings shows that BitNet b1.58 achieves a superior performance on all end tasks, indicating that 1.58-bit LLMs also have strong generalization capabilities.

4 Discussion and Future Work

1-bit Mixture-of-Experts (MoE) LLMs

Mixture-of-Experts (MoE) have proven to be a cost-effective approach for LLMs. While it significantly reduces the computation FLOPs, the high memory consumption and inter-chip communication overhead limit its deployment and application. These challenges can be addressed by 1.58-bit LLMs. Firstly, the reduced memory footprint reduces the number of devices required to deploy MoE models. Moreover, it significantly reduces the overhead of transferring activations across networks. Ultimately, there would be no overhead if the entire models could be placed on a single chip.

1b- Mixtral
Soon ??

Native Support of Long Sequence in LLMs

In the era of LLMs, the ability to handle long sequence has become a critical demand. One major challenge for long sequence inference is the memory consumption introduced by the KV caches. BitNet b1.58 represents a significant step towards native support for long sequences, as it reduces the activations from 16 bits to 8 bits, allowing the context length to be doubled given the same resources. This can be further losslessly compressed to 4 bits or even lower for 1.58-bit LLMs, which we leave as future work.

LLMs on Edge and Mobile

The use of 1.58-bit LLMs has the potential to greatly improve the performance of language models on edge and mobile devices. These devices are often limited by their memory and computational power, which can restrict the performance and the scale of LLMs. However, the reduced memory and energy consumption of 1.58-bit LLMs allows them to be deployed on these devices, enabling a wide range of applications that were previously not possible. This can greatly enhance the capabilities of edge and mobile devices and enable new and exciting applications of LLMs. Moreover, 1.58-bit LLMs are more friendly to CPU devices, which are the main processors used in edge and mobile devices. This means that BitNet b1.58 can be efficiently executed on these devices, further improving their performance and capabilities.

New Hardware for 1-bit LLMs

Recent work like Groq[5] has demonstrated promising results and great potential for building specific hardware (e.g., LPUs) for LLMs. Going one step further, we envision and call for actions to design new hardware and system specifically optimized for 1-bit LLMs, given the new computation paradigm enabled in BitNet [WMD+23].

References

[BZB+19] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. *CoRR*, abs/1911.11641, 2019.

[CCKS23] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-bit quantization of large language models with guarantees. *CoRR*, abs/2307.13304, 2023.

[CLC+19] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *CoRR*, abs/1905.10044, 2019.

[Com23] Together Computer. Redpajama: an open dataset for training large language models, 2023.

[FAHA23] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[HCB+19] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Xu Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems*, pages 103–112, 2019.

[Hor14] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE International Conference on Solid-State Circuits Conference, ISSCC 2014, Digest of Technical Papers, San Francisco, CA, USA, February 9-13, 2014*, pages 10–14, 2014.

[KLZ+23] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[5] https://groq.com/

[LTT+23] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. AWQ: activation-aware weight quantization for LLM compression and acceleration. *CoRR*, abs/2306.00978, 2023.

[MCKS18] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. *CoRR*, abs/1809.02789, 2018.

[MXBS16] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[PKL+16] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.

[RSR+19] 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. *CoRR*, abs/1910.10683, 2019.

[SAL+24] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[SBBC20] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8732–8740, 2020.

[Sha20] Noam Shazeer. GLU variants improve transformer. *CoRR*, abs/2002.05202, 2020.

[TBMR] Jonathan Tow, Marco Bellagente, Dakota Mahan, and Carlos Riquelme. Stablelm 3b 4e1t.

[TCS+24] Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better LLM quantization with hadamard incoherence and lattice codebooks. *CoRR*, abs/2402.04396, 2024.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[TMS+23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, and et al. Llama 2: open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

[WLG17] Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In Leon Derczynski, Wei Xu, Alan Ritter, and Tim Baldwin, editors, *Proceedings of the 3rd Workshop on Noisy User-generated Text, NUT@EMNLP 2017, Copenhagen, Denmark, September 7, 2017*, pages 94–106. Association for Computational Linguistics, 2017.

[WMC+23] Lei Wang, Lingxiao Ma, Shijie Cao, Ningxin Zheng, Quanlu Zhang, Jilong Xue, Ziming Miao, Ting Cao, , and Yuqing Yang. Ladder: Efficient tensor compilation on customized data format. In *OSDI*, 2023.

[WMD+23] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. Bitnet: Scaling 1-bit transformers for large language models. *CoRR*, abs/2310.11453, 2023.

[XLS+23] Guangxuan Xiao, Ji Lin, Mickaël 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, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, 2023.

[YBS19] Vikas Yadav, Steven Bethard, and Mihai Surdeanu. Quick and (not so) dirty: Unsupervised selection of justification sentences for multi-hop question answering. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *EMNLP-IJCNLP*, 2019.

[ZHB+19] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4791–4800, 2019.

[ZS19] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems*, pages 12360–12371, 2019.

[ZZL22] Yichi Zhang, Zhiru Zhang, and Lukasz Lew. PokeBNN: A binary pursuit of lightweight accuracy. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12465–12475. IEEE, 2022.