2023 IEEE International Symposium on Workload Characterization (IISWC) | 979-8-3503-0317-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IISWC59245.2023.00033

CompressionGPT: Evaluating Fault Tolerance of a Compressed Large Language Model

Neil Kapur* America Rangel* Lillian Pentecost

Amherst College

Abstract

Deep neural networks (DNNs) currently require large amounts of memory to store weights. Consequently, inference is less efficient given that weights must be stored off-chip on DRAM, resulting in costly memory accesses. While compression techniques, including quantization and pruning, can significantly reduce model size, current memory technologies are unable to store compressed DNNs on-chip. Prior works have proposed multi-level cell emerging non-volatile memory technologies as a solution given their ability to store bits densely on-chip. While these memory technologies are fault prone, having higher bit error rates, it has been demonstrated that DNNs exhibit some fault tolerance. We build on previous work by examining the fault tolerance of a pruned and quantized large language model (LLM).

1. Introduction

With the emergence of ChatGPT and Bard, large language models have become mainstream and are gaining prevalence in various domains. The very attribute that drives their remarkable performance – their size – also presents a significant limitation: they must be stored and processed on dedicated servers rather than the system the end user utilizes. In addition, existing memory technologies are unable to store the large volume of weights on-chip, resulting in reads with greater latency and energy consumption [6].

Fortunately, prior work demonstrates that DNN weights are amenable to pruning (many can be set to zero) and reduced data precision at little to no loss of accuracy [6].

To address the inefficiency of off-chip weight storage, prior work has shown that it is possible to use multi-level cell emerging non-volatile memory (MLC eNVM) technologies, which are dense and have low read latencies, to store parameters on chip [6]. However, the main caveat is that MLC eNVMs are fault prone, with higher bit error rates during memory accesses [6]. Certain DNNs, including convolutional neural networks (CNNs) for image recognition, have shown to be fault tolerant and therefore, an ideal use case for MLC eNVMs [6, 8]. We build on this work by examining the fault tolerance of a compressed large language model.

2. Methodology

We implemented a methodology that built upon prior work on deep neural network compression and fault injection [6]. After evaluating baseline performance of our selected model, we applied different compression techniques to it, analyzing how performance differed from baseline benchmarks. We then used an existing fault injection framework to measure fault tolerance of the different compression techniques [8]. We incorporated an existing language model evaluation harness to measure performance, in particular model perplexity and accuracy [3].

2.1. nanoGPT

We tested compression techniques and fault injection on the "GPT2-large" [7] language model and used the nanoGPT framework [4] to access and modify weights.

2.2. Model Size Reduction Techniques

2.2.1. Quantization. Rather than using a traditional 32-bit float to represent each weight, we used a smaller, less-precise, fixed-point number. We fixed the number of integer bits at 3 and reduced fractional bits (starting at 29) until model performance degraded. We allowed minor degradation as a trade-off for reduced storage.

2.2.2. Pruning and Bit Mask Encoding. We applied magnitude-based weight pruning to set weights below a certain threshold to zero. We increased the threshold, and therefore sparsity, until performance degraded. After weights were pruned, we used bit mask sparse encoding to store the sparse weights more efficiently. For our achieved sparsity, we did not consider compressed space row (CSR) format as a bit mask could store data more efficiently.

2.3. Ares Fault Injection

To simulate bit flip errors, we used the Ares Fault Injection framework [8]. We injected faults at different bit error rates into pruned and quantized weights to determine which combination of compression techniques were fault tolerant. For bit mask sparse encoded weights, we did not inject faults into the bit mask (we assumed the bit mask would be stored on a more reliable memory system). For our selected compressed model, we completed 100 trials of fault injection for each bit error rate we tested.

2.4. LAMBADA Evaluation

To measure model performance after pruning, quantization, sparse encoding, and fault injection, we used the EleutherAI Language Model evaluation harness [3],

2835-2238/23/$31.00 ©2023 IEEE
DOI 10.1109/IISWC59245.2023.00033

and in particular the LAMBADA dataset (with OpenAI modifications) [5]. We examined accuracy (how well it could predict the next word when given a set of text) and perplexity (a measure of how certain the model is about predicting the next word).

3. Results

We reduced model size without a significant reduction in performance, and noted some fault tolerance in the model. Our compressed model used 3 integer bits and 8 fractional bits for weights and 35% of weights were pruned.

3.1. Model Size

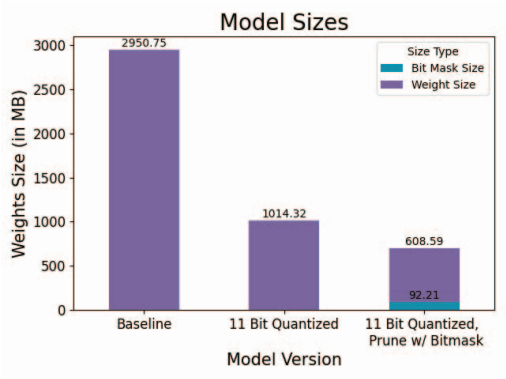

Figure 1: Size of Model in MB Before and After Compression

We were able to reduce model size from 2950 MB to 700.8 MB through quantization, pruning and sparse encoding.

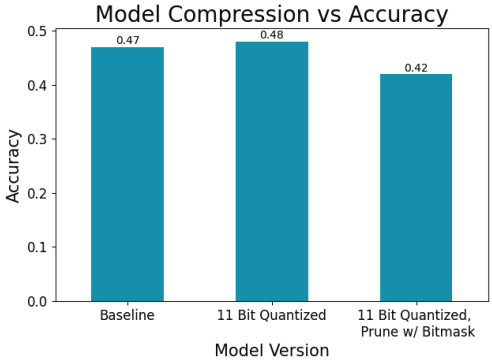

Figure 2: Accuracy of Model Before and After Compression

3.2. Compressed Model Performance

We allowed for a small reduction in accuracy (5%) for a more compressed model. We chose this quantization (11 bits) and sparsity level (35%) as it maximized sparsity and fault tolerance. The baseline perplexity was 12.98 while the compressed perplexity was 24.07.

3.3. Fault Tolerance

The final compressed model did demonstrate fault tolerance up to 10^{-7} bit error rate. After that error rate, performance degraded rapidly. The model was less fault tolerant than models prior works have explored, with some models being able to tolerate bit error rates between 10^{-5} and 10^{-3} [8].

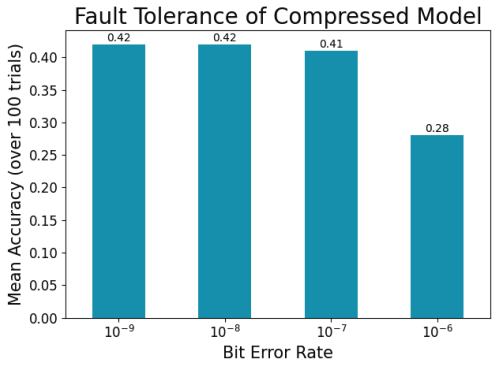

Figure 3: Fault Tolerance of Compressed Model

4. Conclusion

We were able to reduce model size by over 75%, with minimal reduction in performance. We also noted fault tolerance up to a bit error rate of 10^{-7}. Future research could include considering alternative pruning and quantization methods, measuring fault tolerance of recent one-shot pruning [1] and quantization [2] techniques, and finding more fault tolerant data types to store weights.

References

[1] E. Frantar and D. Alistarh, "SparseGPT: Massive language models can be accurately pruned in one-shot," *arXiv preprint arXiv:2301.00774*, 2023.

[2] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training compression for generative pretrained transformers," *arXiv preprint arXiv:2210.17323*, 2022.

[3] L. Gao, J. Tow, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, K. McDonell, N. Muennighoff, J. Phang, L. Reynolds, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou, "A framework for few-shot language model evaluation," Sep. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5371628

[4] A. Karpathy, "nanogpt," 2022. [Online]. Available: https://github.com/karpathy/nanoGPT/

[5] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández, "The lambada dataset: Word prediction requiring a broad discourse context," 2016.

[6] L. Pentecost, M. Donato, B. Reagen, U. Gupta, S. Ma, G.-Y. Wei, and D. Brooks, "Maxnvm: Maximizing dnn storage density and inference efficiency with sparse encoding and error mitigation," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '52. New York, NY, USA: Association for Computing Machinery, 2019, p. 769–781. [Online]. Available: https://doi.org/10.1145/3352460.3358258

[7] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[8] B. Reagen, U. Gupta, L. Pentecost, P. Whatmough, S. K. Lee, N. Mulholland, D. Brooks, and G.-Y. Wei, "Ares: A framework for quantifying the resilience of deep neural networks," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18. New York, NY, USA: ACM, 2018, pp. 17:1–17:6. [Online]. Available: http://doi.acm.org/10.1145/3195970.3195997