How to Exploit Flash Memory for DNN Training

Yejin Han
Dankook University

Bryan S. Kim
Syracuse University

Seehwan Yoo
Dankook University

Jongmoo Choi
Dankook University

Abstract

Deep Neural Networks (DNNs) have been actively adopted in multiple AI applications. Many recent DNNs require millions of parameters and operations for increased accuracy [1, 3, 8, 19], making them highly compute-heavy and memory-intensive.

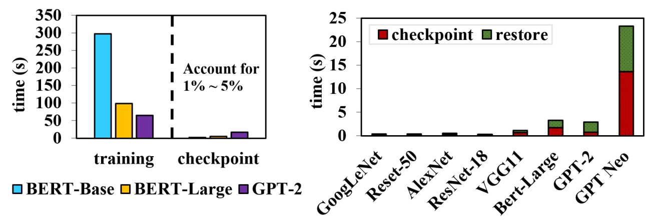
For deeper and wider DNNs, they are trained with massive-scale datasets [10, 14]. Deep Learning (DL) training jobs use entire resources in a server, including storage for storing datasets and model states checkpointed [5, 11, 13], CPU for pre-processing [7], and GPU for computation [12].

Table 1: Experimental environment.

CPU	Intel(R) Xeon(R) Silver 4310 CPU @ 2.10GHz 12-core X 2
GPU	NVIDIA GeForce RTX 3090 24GB X 2
Memory	256GB (32GB X 8 DDR4 3200Mbps)
Storage	480GB Enterprise SAMSUNG SSD X 2
OS	Ubuntu 20.04, Linux kernel 5.15.0
Python	Version 3.8.13
PyTorch	Version 1.12.1

To deeply understand how flash memory is involved in DNN training among those resources, we first conduct an experiment that evaluates the performance of DNN training and model checkpoint/restore tasks. Table 1 summarizes the hardware and software configuration details. We use widely-adopted language models with BERT-Base, BERT-Large [4], GPT-2 [3], GPT-3 (GPT-Neo) [2], and classification models with ResNet-18, ResNet-50 [6], GoogLeNet [17], AlexNet [9], VGG11 [16].

Figure 1 (a) shows elapsed time for three epochs on training language models, which typically have a larger memory footprint than the classification model. For GPT-2, we pre-trained it and then finetuned the resulting model. Traditionally, storage I/O is considered high overhead from the perspective of a memory hierarchy [15, 18]. So we assume that checkpoint overhead can be significant. Still, following our observation, checkpoint stall is relatively small compared to overall training time, less than 1% on BERT and 5% on GPT-2.



(a) checkpoint stall in training

(b) checkpoint/restore time

Figure 1: Evaluation on training 9 DNN models. Checkpoint stall is insignificant compared to the overall training time, accounting for 1% - 5%.

We further analyze the impact of storage I/O under checkpoint/restore situations. Figure 1 (b) presents various model checkpoint/restore latency on flash memory. Although some differences depending on the model, it shows low overhead compared to overall training time. If the model size becomes much larger, there can be increasing I/O latency, but training time does simultaneously. So the tendency of little checkpoint overhead in overall training remains the same. As a result, our counter-result shows that checkpointing itself is not a bottleneck in DNN training.

Using this insight, we are currently working and implementing the policy on saving DRAM resources to aggressively use checkpointing, which has little effect on total training time. According to another observation in our experiment, when multiple models are co-located, CPU and GPU become bottlenecks increasing training time significantly, while checkpoint I/O interference is not noticeable. The checkpoint can be used not only for fast recovery from failures [11] and load balancing across different nodes [5] but also for expensive DRAM saving and energy efficiency.

The following directions will be investigated for future work: (1) deeper analysis of on-demand dataset loadind for reducing DRAM (2) validating our proposed scheme under different DNN models and co-locating environment (3) investigating the impact of model/data parallelism on our policy

References

[1] ATHLUR, S., SARAN, N., SIVATHANU, M., RAMJEE, R., AND KWATRA, N. Varuna: scalable, low-cost training of massive deep learning models. In *Proceedings of the Seventeenth European Conference on Computer Systems* (2022), pp. 472–487.

[2] BLACK, S., GAO, L., WANG, P., LEAHY, C., AND BIDERMAN, S. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, Mar. 2021. If you use this software, please cite it using these metadata.

[3] BROWN, T., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J. D., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., ET AL. Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901.

[4] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[5] EISENMAN, A., MATAM, K. K., INGRAM, S., MUDIGERE, D., KRISHNAMOORTHI, R., NAIR, K., SMELYANSKIY, M., AND ANNAVARAM, M. {Check-N-Run}: a checkpointing system for training deep learning recommendation models. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)* (2022), pp. 929–943.

[6] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[7] JIANG, Y., ZHU, Y., LAN, C., YI, B., CUI, Y., AND GUO, C. A unified architecture for accelerating distributed {DNN} training in heterogeneous {GPU/CPU} clusters. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)* (2020), pp. 463–479.

[8] KIM, S., JIN, Y., SOHN, G., BAE, J., HAM, T. J., AND LEE, J. W. Behemoth: A flash-centric training accelerator for extreme-scale dnns. In *USENIX Conference on File and Storage Technologies* (2021).

[9] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM 60*, 6 (may 2017), 84–90.

[10] LIN, T.-Y., MAIRE, M., BELONGIE, S., HAYS, J., PERONA, P., RAMANAN, D., DOLLÁR, P., AND ZITNICK, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision* (2014), Springer, pp. 740–755.

[11] MOHAN, J., PHANISHAYEE, A., AND CHIDAMBARAM, V. Checkfreq: Frequent, fine-grained dnn checkpointing. *Proceedings of the USENIX Conference on File and Storage Technologies (FAST 2021)*.

[12] MOHAN, J., PHANISHAYEE, A., RANIWALA, A., AND CHIDAMBARAM, V. Analyzing and mitigating data stalls in dnn training. *Proc. VLDB Endow. 14*, 5 (mar 2021), 771–784.

[13] ROJAS, E., KAHIRA, A. N., MENESES, E., GOMEZ, L. B., AND BADIA, R. M. A study of checkpointing in large scale training of deep neural networks. *arXiv preprint arXiv:2012.00825* (2020).

[14] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M., ET AL. Imagenet large scale visual recognition challenge. *International journal of computer vision 115*, 3 (2015), 211–252.

[15] SHIN, W., CHEN, Q., OH, M., EOM, H., AND YEOM, H. Y. Os i/o path optimizations for flash solid-state drives. USENIX ATC'14, USENIX Association, p. 483–488.

[16] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[17] SZEGEDY, C., LIU, W., JIA, Y., SERMANET, P., REED, S., ANGUELOV, D., ERHAN, D., VANHOUCKE, V., AND RABINOVICH, A. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 1–9.

[18] WU, K., GUO, Z., HU, G., TU, K., ALAGAPPAN, R., SEN, R., PARK, K., C. ARPACI-DUSSEAU, A., AND H. ARPACI-DUSSEAU, R. The storage hierarchy is not a hierarchy: Optimizing caching on modern storage devices with orthus. In *19th USENIX Conference on File and Storage Technologies* (February 2021), pp. 307–323.

[19] XU, X., DING, Y., HU, S. X., NIEMIER, M., CONG, J., HU, Y., AND SHI, Y. Scaling for edge inference of deep neural networks. *Nature Electronics 1*, 4 (2018), 216–222.