# Optimizing Communication in Parallel Deep Learning via Parameter Pruning

Siddharth Singh, Abhinav Bhatele

Department of Computer Science, University of Maryland, College Park, MD 20742 USA
E-mail: ssingh37@umd.edu, bhatele@cs.umd.edu

***Abstract*—Parallel training of neural networks at scale is challenging due to significant overheads arising from communication. Recently, deep learning researchers have developed a variety of pruning algorithms that are capable of pruning (i.e. setting to zero) 80-90% of the parameters in a neural network to yield sparse subnetworks that equal the accuracy of the unpruned parent network. In this work, we propose a novel approach that exploits these sparse subnetworks to reduce communication overheads. We integrate our approach into AxoNN, a highly scalable framework for parallel deep learning that relies on data and inter-layer parallelism, and empirically demonstrate the reduction in communication times. Our approach yields a speedup of 17% when training a 2.7 billion parameter transformer model on 384 GPUs.**

***Index Terms*—parameter pruning, distributed deep learning, GPT style transformers**

## I. Introduction

Parallel training of neural networks at scale is challenging due to significant overheads arising from communication. Recently, deep learning researchers have developed a variety of pruning algorithms that are capable of pruning (i.e. setting to zero) 80-90% of the parameters in a neural network to yield sparse subnetworks that equal the accuracy of the unpruned parent network [1], [2]. However, at the level of sparsities of the parameter tensors enforced by these pruning algorithms (0.8-0.9), sparse matrix computations on modern GPUs are prohibitively expensive even with state-of-the-art sparse matrix libraries (see Figure 1). Thus, these pruning algorithms haven't been utilized from a systems perspective to improve the performance of distributed neural network training.

Instead of trying to improve computation times, we introduce a novel method which exploits these accuracy-preserving sparse subnetworks to reduce communication. To circumvent the overhead of sparse tensor computation, we dynamically rematerialize sparse matrices as ephemeral dense matrices whenever it is required by the training procedure and delete it after the computation is finished. In a number of distributed deep learning frameworks, the overhead of collective communication is directly proportional to the number of parameters in the network [4]–[6]. Since pruning reduces the number of parameters, we observe a significant reduction in these overheads and a corresponding improvement in hardware utilization. For frameworks that rely on point-to-point communication like AxoNN [7], our method allows the packaging of more parameters per GPU post-pruning, thus increasing the ratio of computation to point-to-point communication.

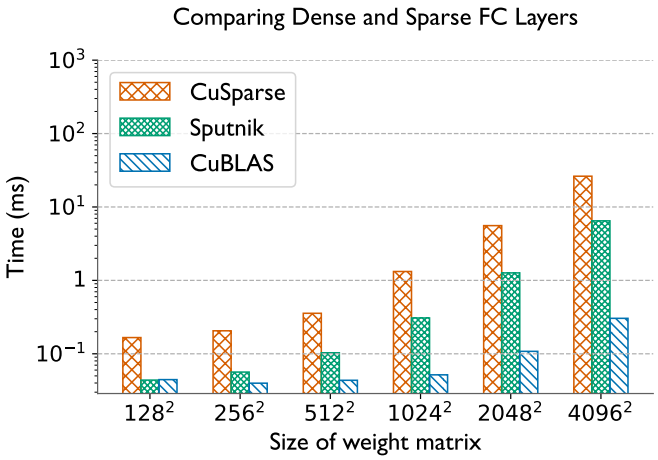


Fig. 1. Comparison of the execution times of a fully-connected (FC) layer with a randomly generated 90% sparse square weight matrix in mixed precision. For dense GPU kernels we use NVIDIA's CuBLAS, whereas for sparse GPU kernels we use NVIDIA's CuSparse and Sputnik [3], the state-of-the-art for accelerated sparse GPU computation for DL workloads. We fix the input batch size to 576 and vary the size of the weight matrix from $128^2$ to $4096^2$.

## II. Method

In this section, we describe our method that utilizes the existence of sparse subnetworks to improve the efficiency of distributed neural network training.

- While our approach is independent of the choice of pruning algorithm, for this work we use You et al.'s Early Bird Ticket [2] approach to prune our models.
- Post pruning, we store the sparse parameter matrices in a one-dimensional Sparse COOrdinate (COO) format. This is memory efficient because we only need to store one non-zero index per unpruned parameter.
- Since sparse matrix computation libraries are slow, we do the computation in dense using CuBLAS. This is done by dynamically converting a sparse parameter matrix into dense when needed for compute. After computation is completed, we immediately discard the dense matrix. We are thus able to save memory while matching the performance of the unpruned network.

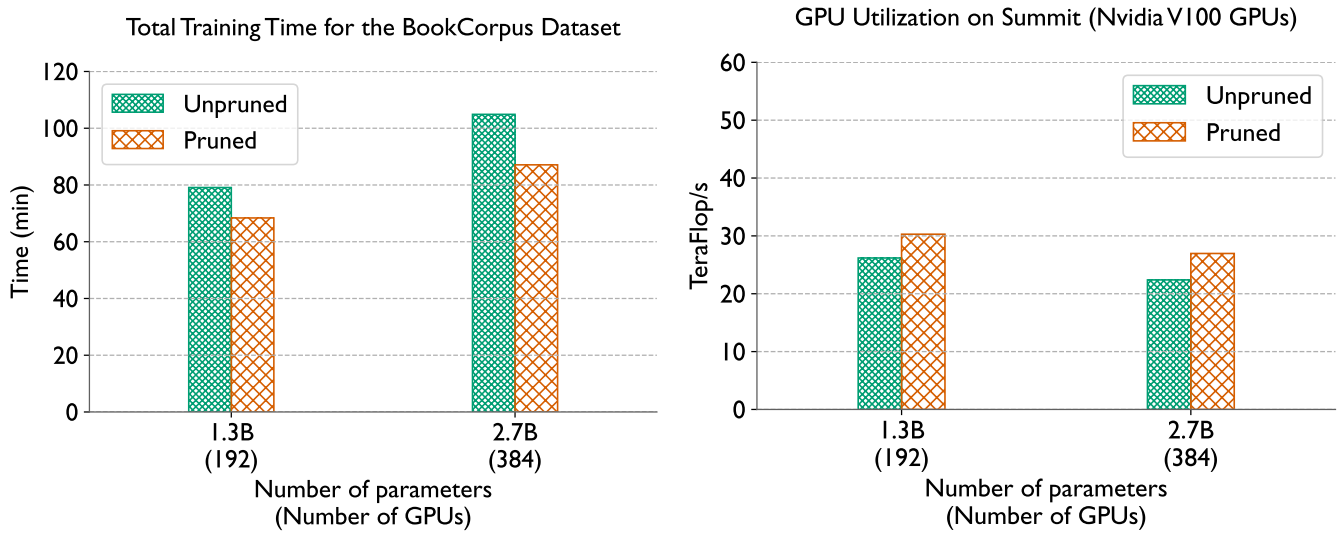


Fig. 2. Total Training Time (left) and GPU FLOP/s (right) for GPT-1.3B and GPT-2.7B on 192 and 384 NVIDIA V100 GPUs on Summit respectively. Models are trained using AxoNN [7].

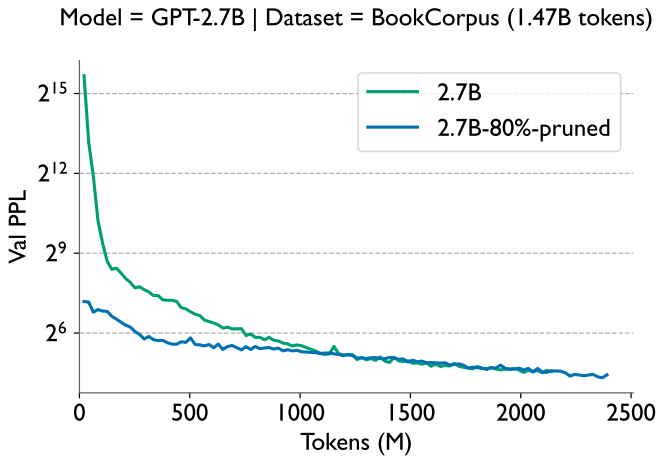


Fig. 3. Validation Perplexity of GPT-2.7B when pruned with the Early Bird Ticket approach [2] matches the unpruned network.

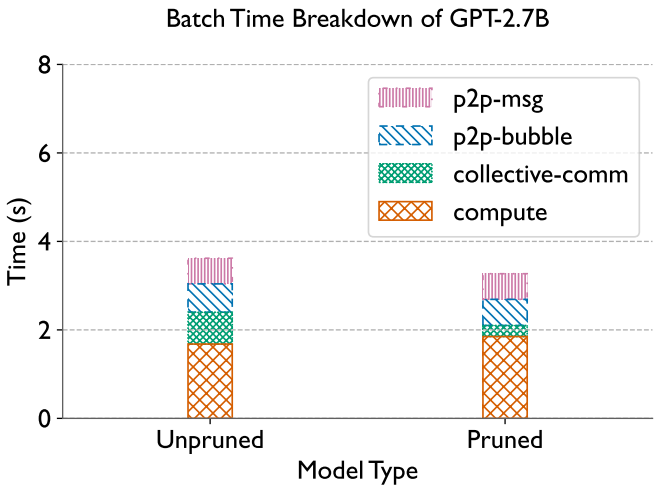


Fig. 4. Batch time breakdown of GPT-2.7B trained on 384 NVIDIA V100 GPUs on Summit.

## III. Results and Analysis

We integrate our method in AxoNN [7] and demonstrate the results in Figure 2. We profile the 1.3 billion parameter model GPT-1.3B [8] and 2.7 billion parameter model GPT-2.7B [8] on 192 and 384 GPUs on Summit respectively. For both the models, we prune 80 percent of the parameters using our method. We observe a significant speedup of 14% and 17% for GPT-1.3B and GPT-2.7B respectively. For GPT-1.3B our algorithm improves AxoNN's GPU utilization from 26.18 Tflop/s to 30.29 TFLOP/s, whereas for GPT-2.7B we observe an improvement from 22.39 to 27 TFLOP/s.

Figure 4 provides a breakdown of the batch time. We notice that most of the improvement in performance comes from a 66% reduction in the collective communication time. We also observe a minor improvement of 6% in the pipeline bubble time, which is the idle time AxoNN spends in pipeline parallelism. We also note that the rematerialization of dense tensors from sparse tensors adds an overhead of 10% in the compute phase.

## Acknowledgment

This work was supported by funding provided by the University of Maryland College Park Foundation. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

## References

[1] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *International*

*Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=rJl-b3RcF7

[2] H. You, C. Li, P. Xu, Y. Fu, Y. Wang, X. Chen, R. G. Baraniuk, Z. Wang, and Y. Lin, "Drawing early-bird tickets: Toward more efficient training of deep networks," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=BJxsrgStvr

[3] T. Gale, M. Zaharia, C. Young, and E. Elsen, "Sparse GPU kernels for deep learning," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2020*, 2020.

[4] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, "Zero: Memory optimizations toward training trillion parameter models," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '20. IEEE Press, 2020.

[5] J. Ren, S. Rajbhandari, R. Y. Aminabadi, O. Ruwase, S. Yang, M. Zhang, D. Li, and Y. He, "Zero-offload: Democratizing billion-scale model training," *CoRR*, vol. abs/2101.06840, 2021. [Online]. Available: https://arxiv.org/abs/2101.06840

[6] S. Rajbhandari, O. Ruwase, J. Rasley, S. Smith, and Y. He, "Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning," ser. SC '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3458817.3476205

[7] S. Singh and A. Bhatele, "AxoNN: An asynchronous, message-driven parallel framework for extreme-scale deep learning," in *Proceedings of the IEEE International Parallel & Distributed Processing Symposium*, ser. IPDPS '22. IEEE Computer Society, May 2022 (to appear).

[8] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, vol. abs/2005.14165, 2020. [Online]. Available: https://arxiv.org/abs/2005.14165