AxoNN: Hybrid Asynchronous Algorithms for Parallel Deep Learning

Siddharth Singh, Abhinav Bhatele
Department of Computer Science, University of Maryland

Abstract

1. Due to high communication overheads, training multi-billion parameter neural networks at scale is a challenging problem.

2. We present AxoNN, an asynchronous hybrid parallel framework for training such models on networked GPU clusters.

3. AxoNN features two highly scalable implementations of inter-layer and tensor parallelism for efficiently training models that do not fit on a single GPU.

4. On 256 A100 GPUs, AxoNN trains a 28B parameter CNN 2.5x faster than the state-of-the-art.

Designing a Hybrid Parallel Framework

1. AxoNN's parallelism is a hybrid of data and model parallelism.

2. Organize GPUs in a virtual two-dimensional topology, with dimensions $G_{data} \times G_{model}$.

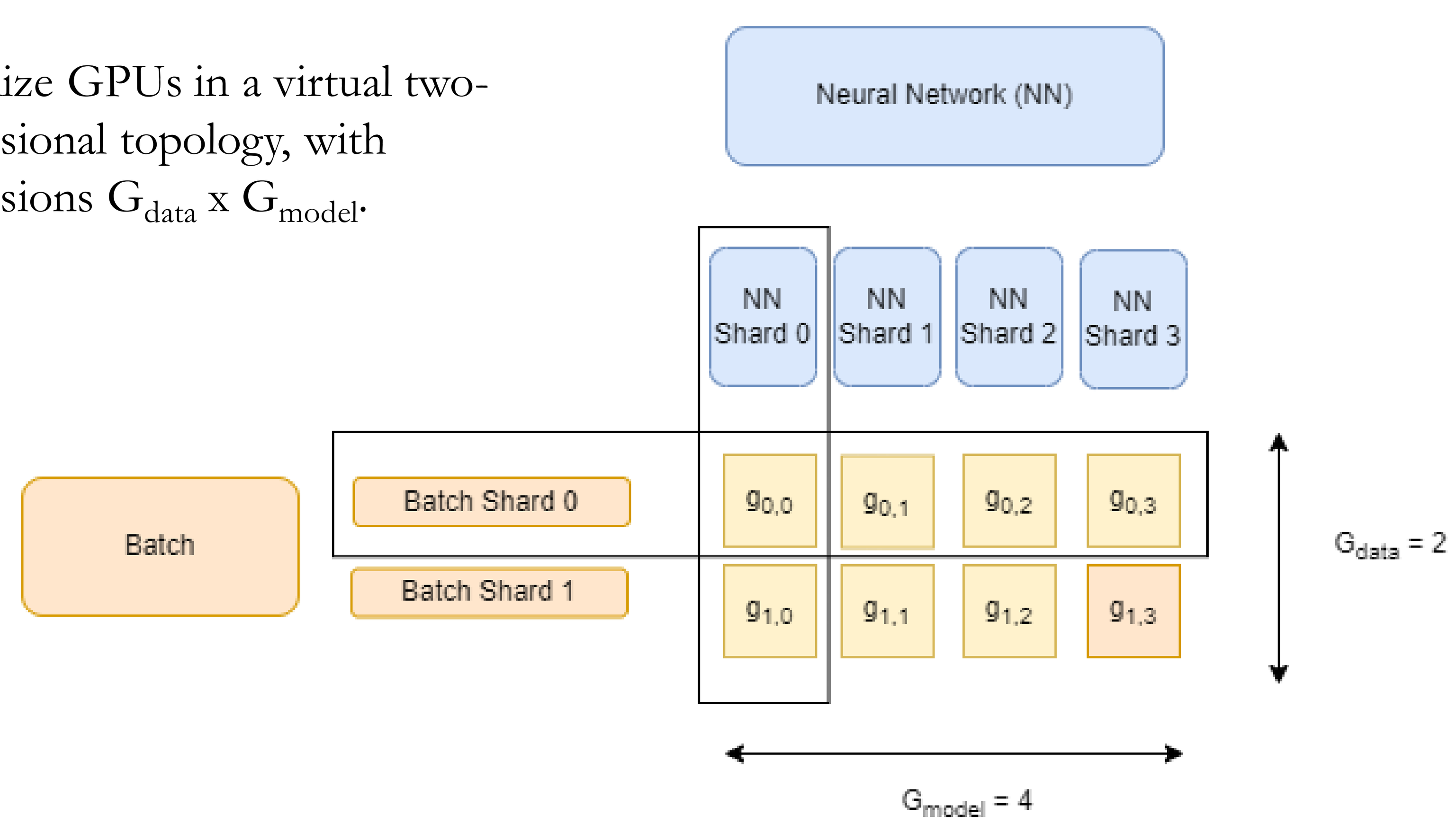

Fig 1: Schematic diagram for AxoNN's hybrid parallelism on 8 GPUs with $G_{data} = 2$ and $G_{model}=4$. GPU $g_{i,j}$ computes on the i^{th} batch shard and j^{th} neural network shard.

3. Data Parallelism – Each row of GPUs computes on an equally sized shard of the input batch.

4. Model Parallelism – Each column of GPUs computes on an equally sized shard of the neural network. Two types – inter-layer and tensor parallelism.

Inter-Layer Parallelism

1. Distribute neural network layers equally within model parallel GPUs.

2. Divide batch shard into microbatches and execute them in a pipelined fashion.

3. An asynchronous, message-driven communication backend to effectively overlap communication with computation [1].

4. An efficient memory optimization algorithm that moves optimizer data to the CPU and saves 4x memory [1].

5. We then exploit the saved memory to greatly reduce point-to-point communication volume [1].

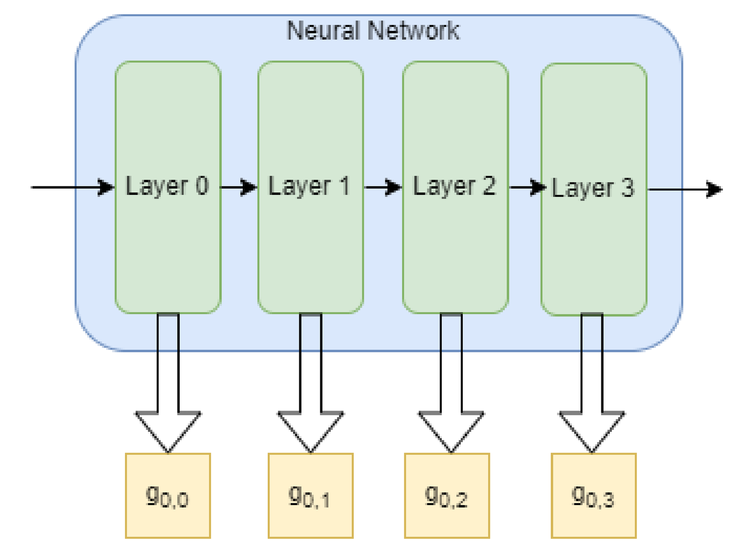

Fig 2: Distribution of neural network compute under inter-layer parallelism. across GPUs in the first row of Figure 1.

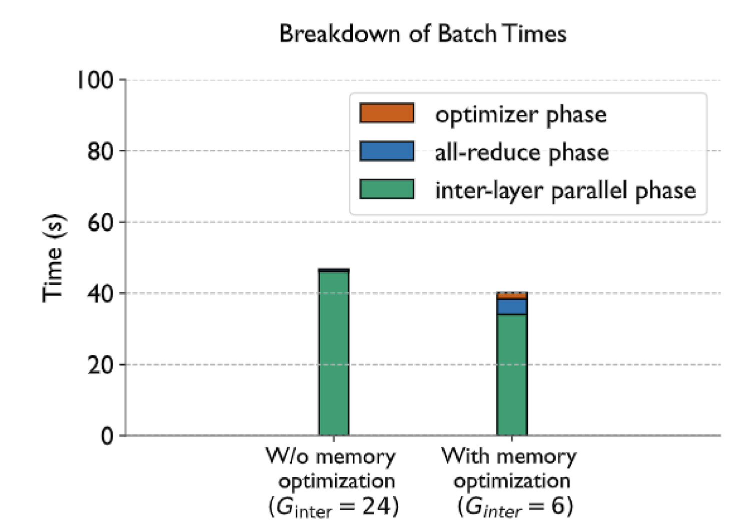

Fig 3: Batch time breakdown for a 12B parameter GPT on 48 GPUs of Summit.

Tensor Parallelism

1. A novel asynchronous two-dimensional (2D) algorithm for parallelizing the computation of every layer of the neural network across model parallel GPUs.

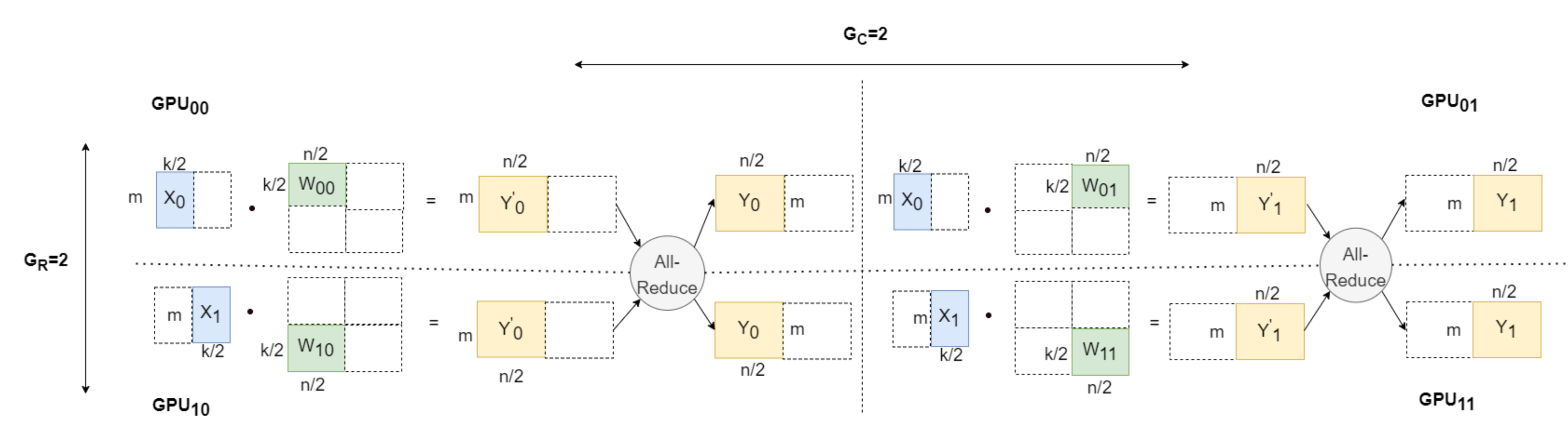

Fig 4: Computing an FC layer with our tensor 2D tensor parallel algorithm on 4 GPUs.

2. Communication models to derive communication-optimal configurations for arbitrary models.

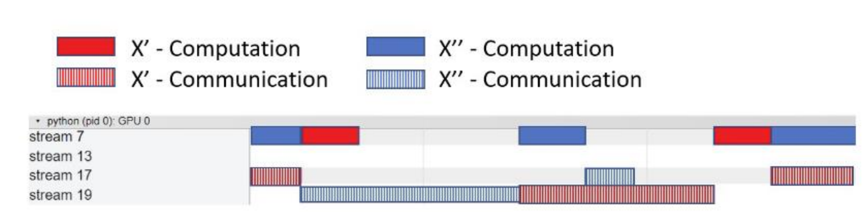

Fig 5: Trace of our tensor parallel algorithm for a 10B parameter GPT on 8 A100 GPUs.

Results

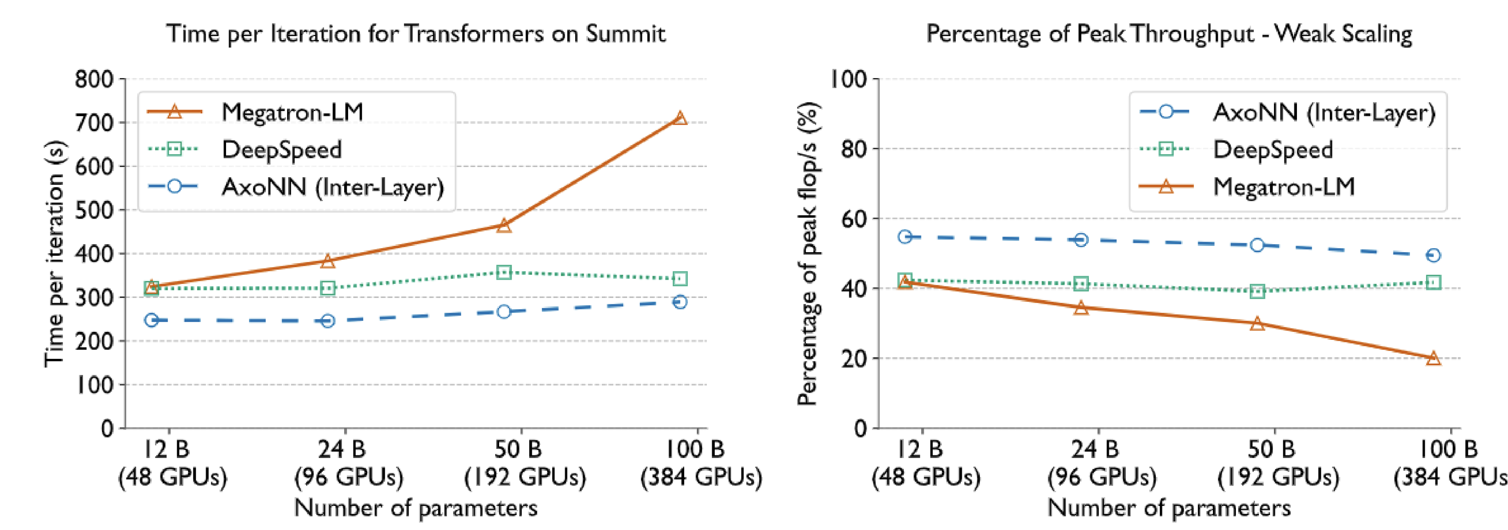

Fig 6: Comparing the weak scaling performance of AxoNN's inter-layer parallelism with other frameworks on GPT neural networks on Summit. For the 100 B model, AxoNN is faster than DeepSpeed by 1.18× and Megatron-LM by 2.46×.

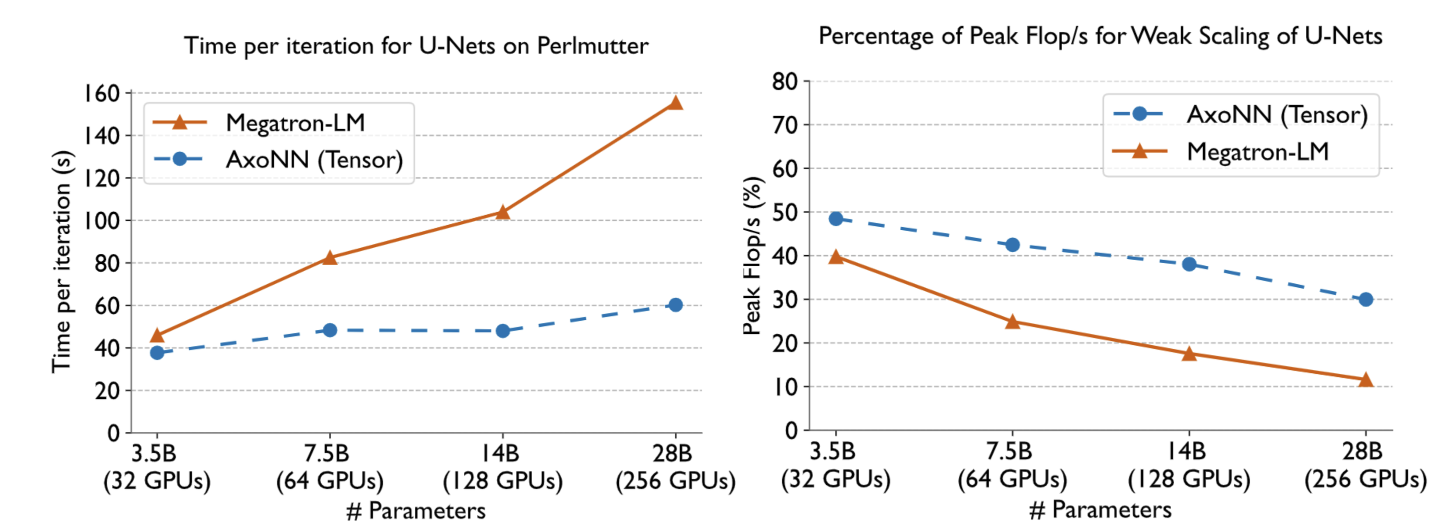

Fig 7: Comparing the weak scaling performance of AxoNN's tensor parallelism with Megatron-LM on U-Nets, on Perlmutter. For the 28 B model, AxoNN is 2.5x faster than Megatron-LM.

Conclusion and Future Work

1. Presented AxoNN, an asynchronous hybrid parallel framework for parallel deep learning.

2. Developed highly optimized implementations of inter-layer and tensor parallelism with a focus on minimizing communication time.

3. Future work involves combining inter-layer and pipeline parallelism and developing methods to autotune configuration parameters.

References

[1] Singh et al., AxoNN: An asynchronous, message-driven parallel framework for extreme-scale deep learning, IPDPS 2022

Acknowledgements

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. This research also used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC award DDR-ERCAP0025593.