Exploiting Sparsity in Pruned Neural Networks to Optimize Large Model Training

Siddharth Singh and Abhinav Bhatele
Department of Computer Science
UNIVERSITY OF MARYLAND

Neural Networks

- Neural Networks (NNs): ‘Parameterized’ function approximators
- Can work with very high dimensional data.

Input (x)

Neural Network

“A sitting dog”

Output $f(x;\theta)$

Parameters(θ)

‘Learn’ θ from data

Mostly composed of matrix multiplications.

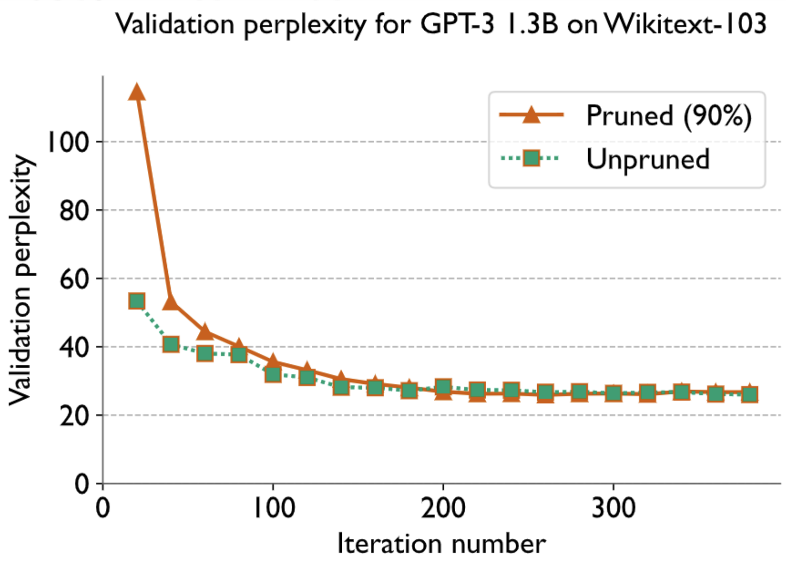
Stochastic Gradient Descent

- Repeat until loss (L) has been minimized sufficiently:
  - Read in a batch of training data
  - Forward Pass : Calculate output $f(x;\theta)$ and the loss (L) on the batch.
  - Backward Pass : Calculate gradients of the loss wrt the parameters ($\frac{\partial L}{\partial \theta}$).
  - Optimizer Step : Use $\frac{\partial L}{\partial \theta}$ to update θ.

Neural Network Pruning

- Zeroing parameters with small magnitudes permanently mid-training.

- DL pruning algorithms can prune as many as 80-90% of the parameters without affecting model quality.



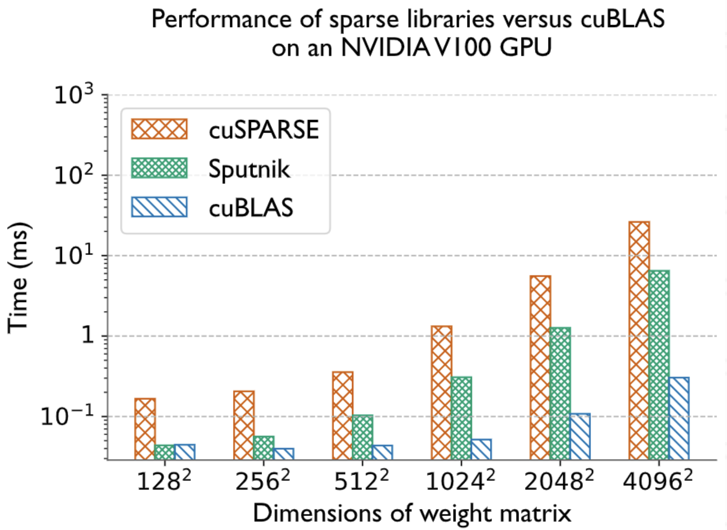
GPT-3 1.3B pruned to 90% sparsity using [1].

Can we exploit pruning in large models to improve performance of parallel training on multi-GPU clusters?

Sparse matrix multiplication?

- Most compute in a pruned NNs is sparse matrix multiplication.
- Can we use optimized implementations of SpMM?

Instead, we focus on optimizing communication volume in parallel training of NNs.



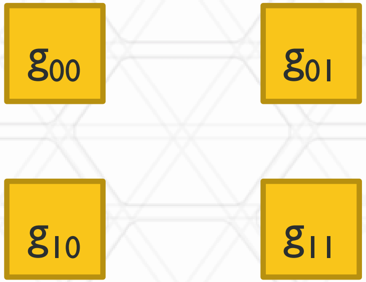
Comparison of CuBLAS with sparse libraries on a 90% pruned FC layer.

Background on AxoNN

- In this work we used AxoNN [2] as our parallel DL framework of choice.
- AxoNN implements a hybrid parallel algorithm of data and inter-layer parallelism.

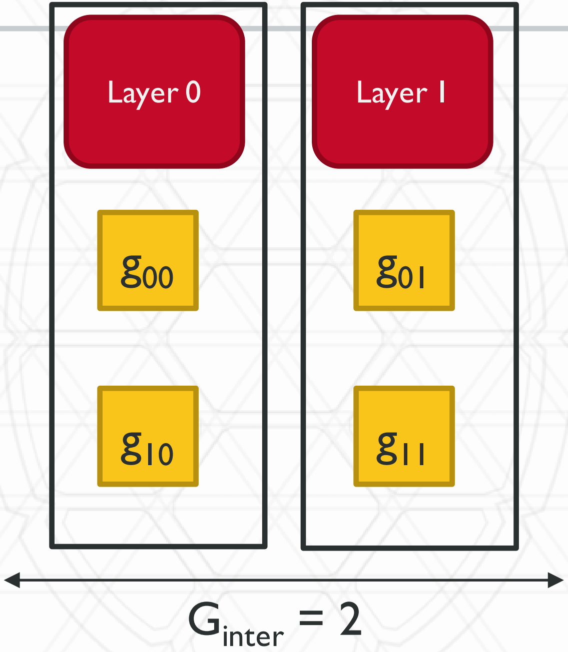
A two layer neural network on 4 GPUs.

Distribution of Compute in AxoNN



Organize GPUs in a 2D grid

Distribution of Compute in AxoNN

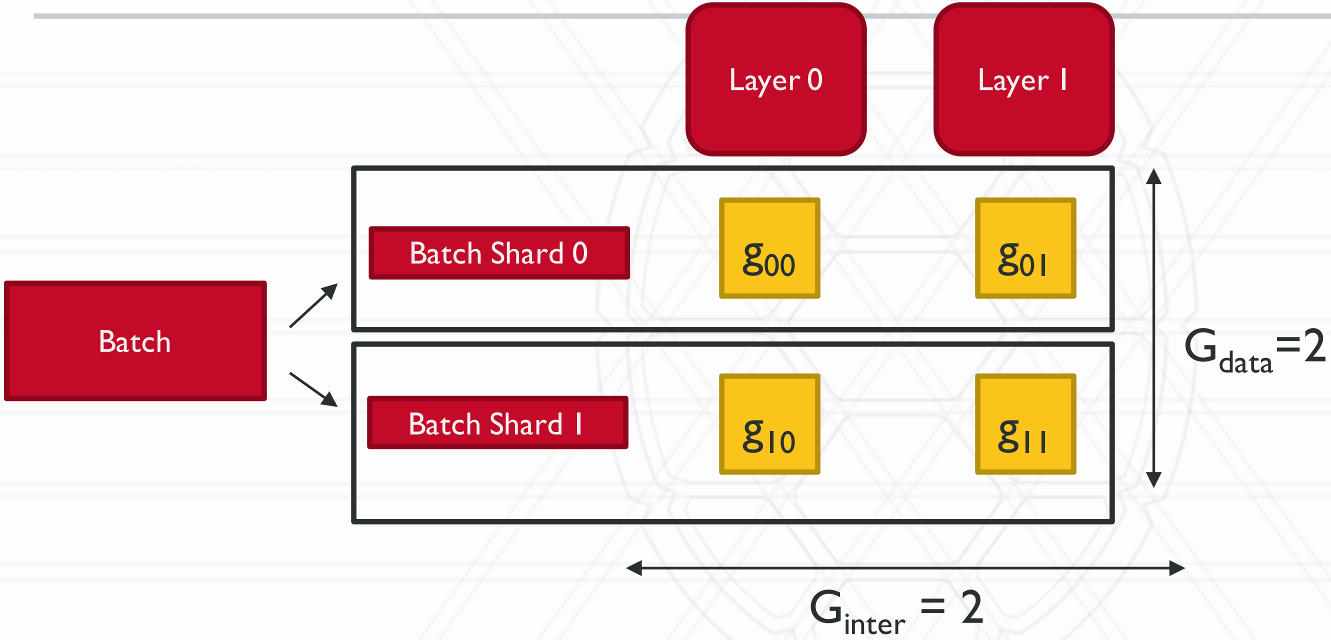


Inter-Layer Parallelism

Partition layers equally across columns

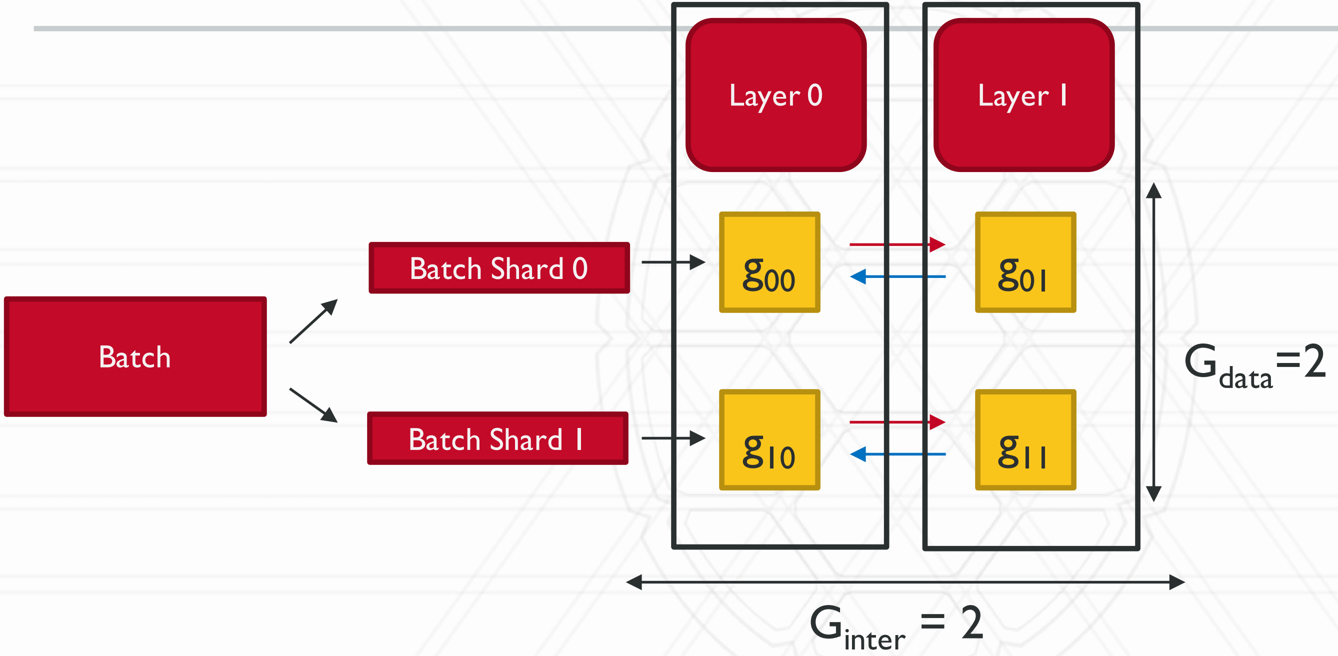
Distribution of Compute in AxoNN

Data Parallelism



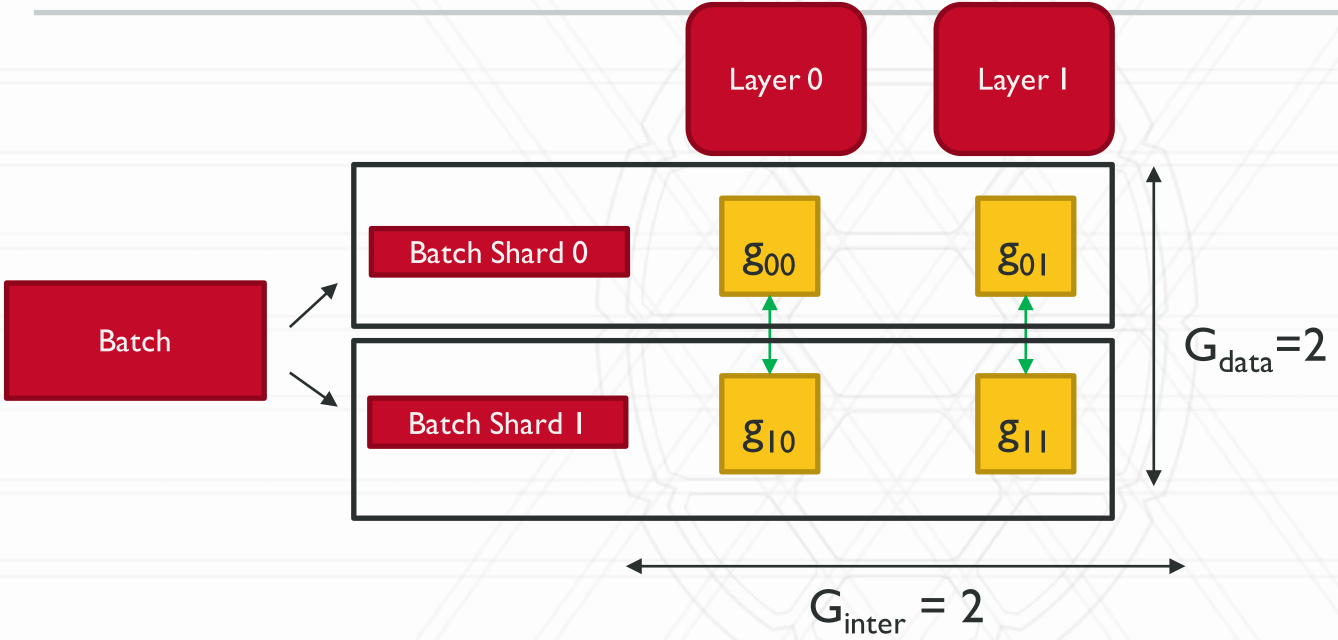
Partition batch equally across rows

Communication in Inter-Layer Parallelism



Point-to-point communication of activations (FW pass) and their gradients (BW pass)

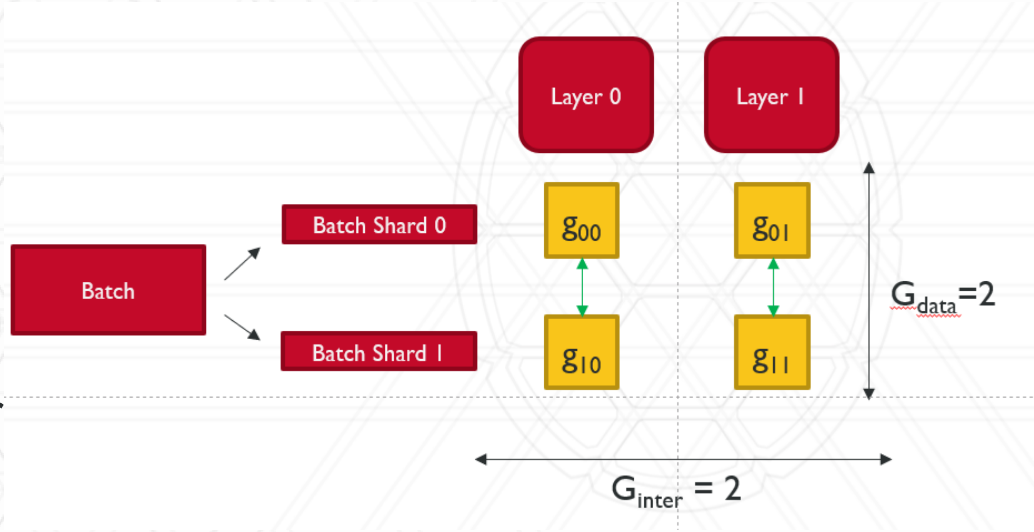
Communication in Data Parallelism



All-reduce to synchronize gradients after BW pass.

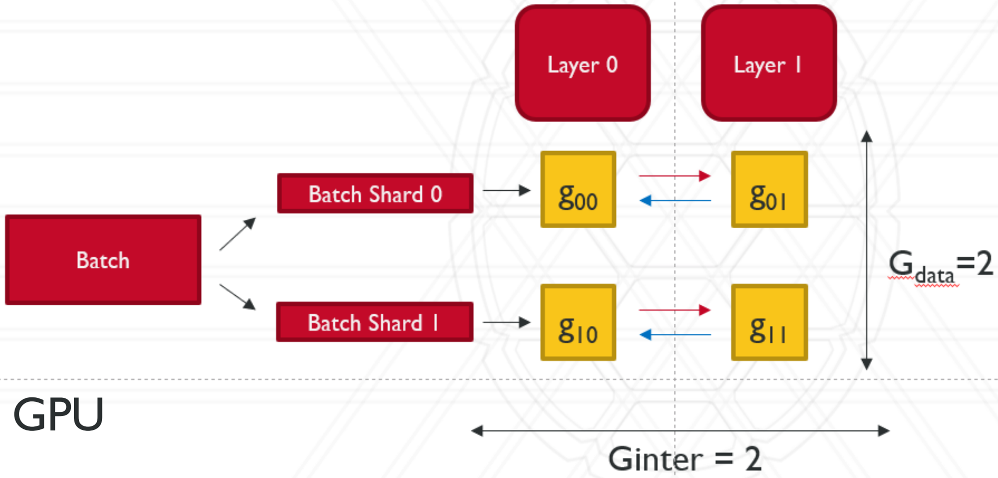
Optimizing Communication in Pruned NNs

- Data Parallelism
  - Communication – All-reduce on gradients.
  - Volume $\propto |\theta|$
  - Simple! – Only communicate gradients of unpruned parameter



Optimizing Communication in Pruned NNs

- Inter-Layer Parallelism
  - Communication – P2P comm. of activations and their gradients
  - Messages aren't sparse
  - Volume $\propto G_{inter}$ (proof in paper)
  - Decrease G_{inter} → More layers per GPU



Need to optimize memory consumption by exploiting pruning.

Sparsity Aware Memory Optimization (SAMO)

- Selective compression of model states after pruning.

θ

Parameters

$\frac{\partial L}{\partial \theta}$

Gradients

s_{opt}

Optimizer data

Do not compress

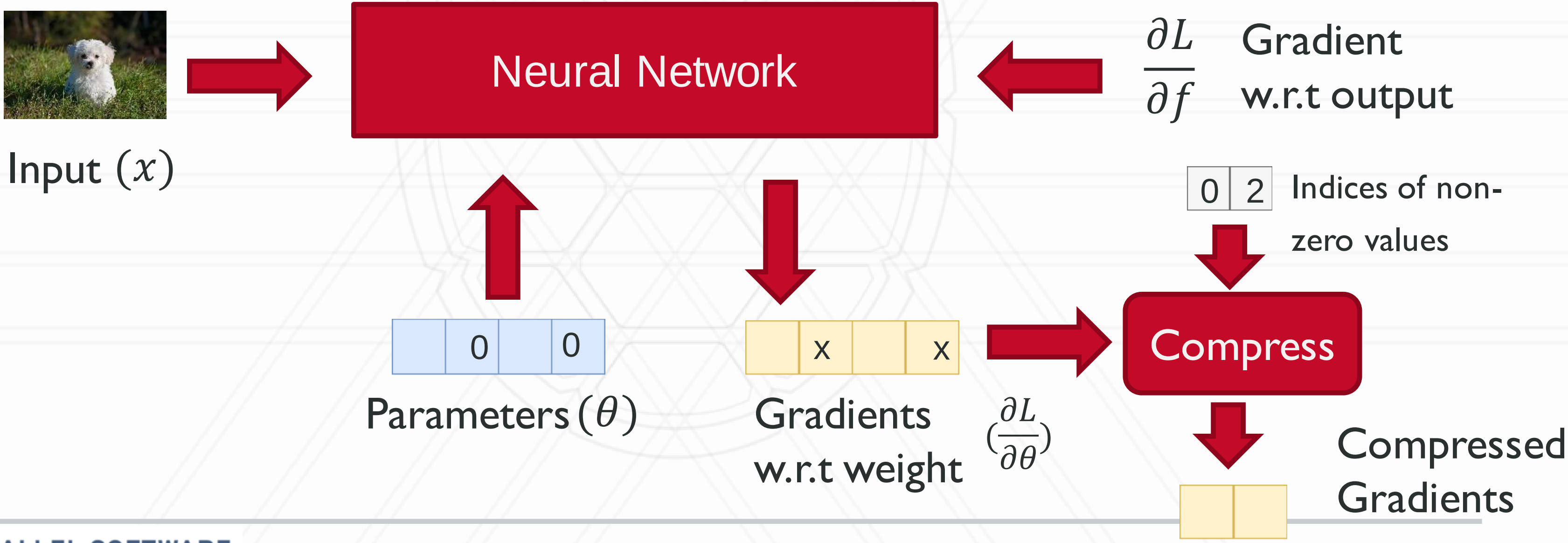
- Store in dense with 0s explicitly filled out.
- Invoke efficient dense CuBLAS kernels for matrix mult.

Compress

- Store in a 1D sparse COO format.
- Common index vector of non-zero elements.

Overheads in SAMO

- Backward pass – Compute gradients with **dense computation kernels and then compress**

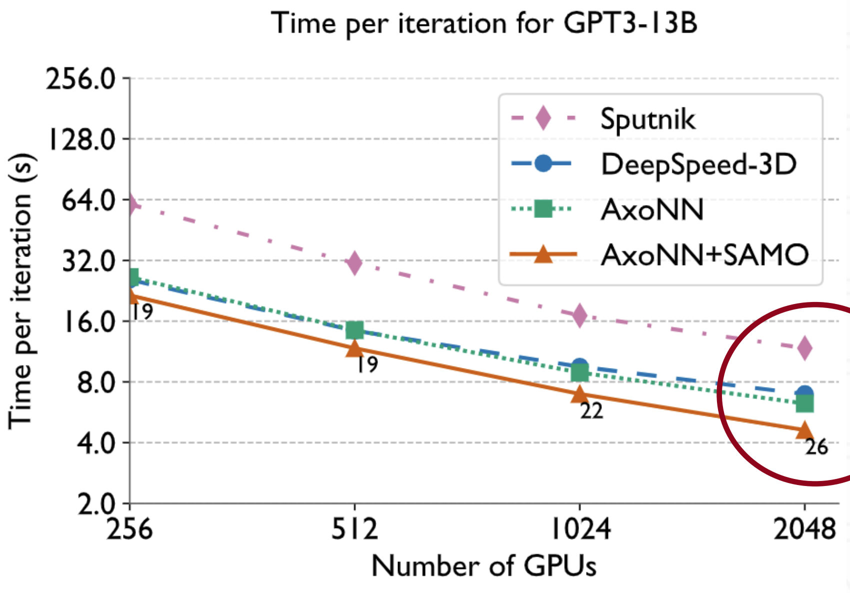


Sparsity Aware Memory Optimization (SAMO)

- Assuming mixed precision and the Adam Optimizer, we prove that our method **saves 66-78% memory** for an 80-90% pruned NN.

- Exploit the saved memory to decrease G_{inter} and decrease point-to-point communication.

Results



31% of peak flop/s

Strong Scaling of GPT3-13B on Summit. We prune to 90% sparsity using [1]. We annotate AxoNN+SAMO's line with its percentage speedup over AxoNN.

Conclusion

- Developed a novel method that exploits neural network pruning algorithm in large models to improve performance of parallel training.
- Presented Sparsity-Aware Memory Optimization (SAMO) to significantly reduce memory consumption while not sacrificing performance.
- Demonstrated how the memory saved can be used to optimize communication in data and inter-layer parallelism.

Future Work

- Training pruned large language models.
  - Imagine a ChatGPT like model that fits on your laptop.
- Accelerating inference tasks via pruning.
- Experimenting with other forms of parallelism like tensor parallelism.

Bibloliography

[1] Drawing Early-Bird Tickets: Toward More Efficient Training of Deep Networks, You et al., ICLR 2020, https://openreview.net/forum?id=BJxsrgStvr

[2] AxoNN: An asynchronous, message-driven parallel framework for extreme-scale deep learning, Siddharth Singh and Abhinav Bhatele, IPDPS 2022, https://arxiv.org/abs/2110.13005

UNIVERSITY OF MARYLAND

Siddharth Singh
ssingh37@umd.edu