AxoNN

Democratizing AI: Open-source Scalable LLM Training on GPU-based Supercomputers

Siddharth Singh, Prajwal Singhania, Aditya Ranjan, John Kirchenbauer, Jonas Geiping,Yuxin Wen, Neel Jain, Abhimanyu Hans, Manli Shu, Aditya Tomar, Tom Goldstein, Abhinav Bhatele

ACM Gordon Bell Prize '24 Finalist

PARALLEL SOFTWARE AND SYSTEMS GROUP

UNIVERSITY OF MARYLAND

SC24 Atlanta, GA | hpc creates.

ACM Gordon Bell Prize

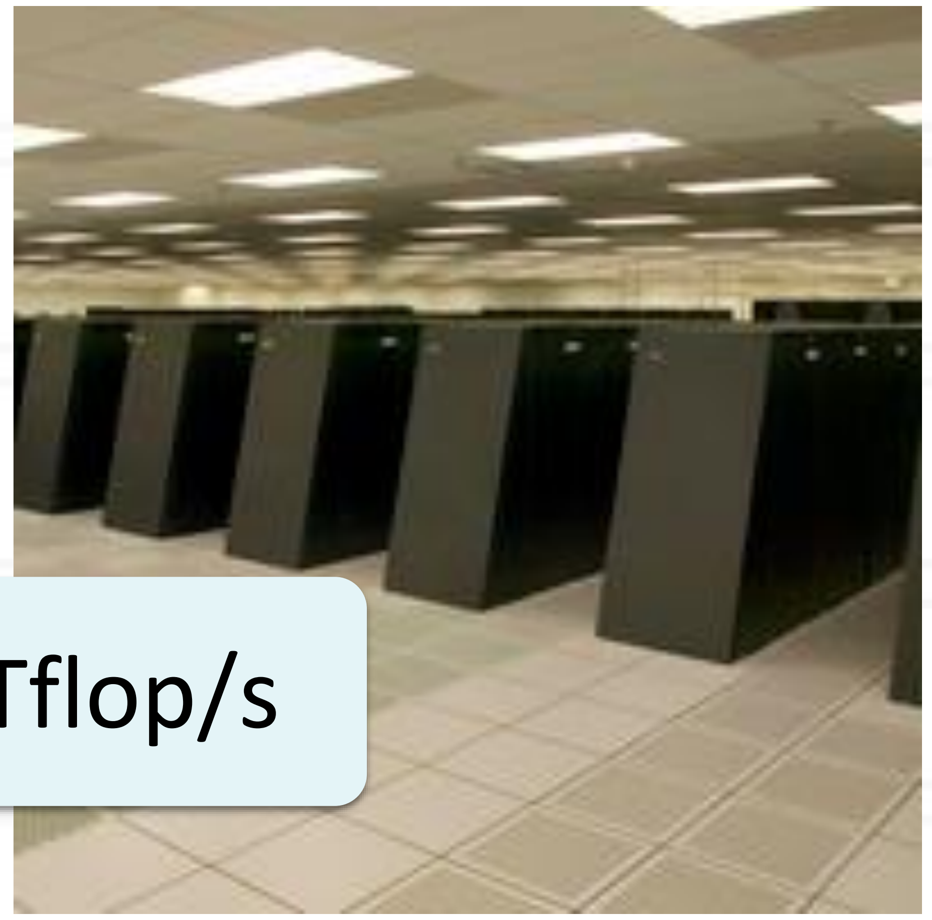
- Awarded annually by ACM to recognize major achievements in HPC

- Focus on innovations in scalability and potential real-world impact
  - Scalability – Use as many compute resources (GPUs/CPUs) as efficiently as possible [aka FLOPs-Maxxing]
  - Scientific impact – Contributions to a scientific domain

- Six Finalists at Supercomputing 24 - two of which focused on large scale LLM training.

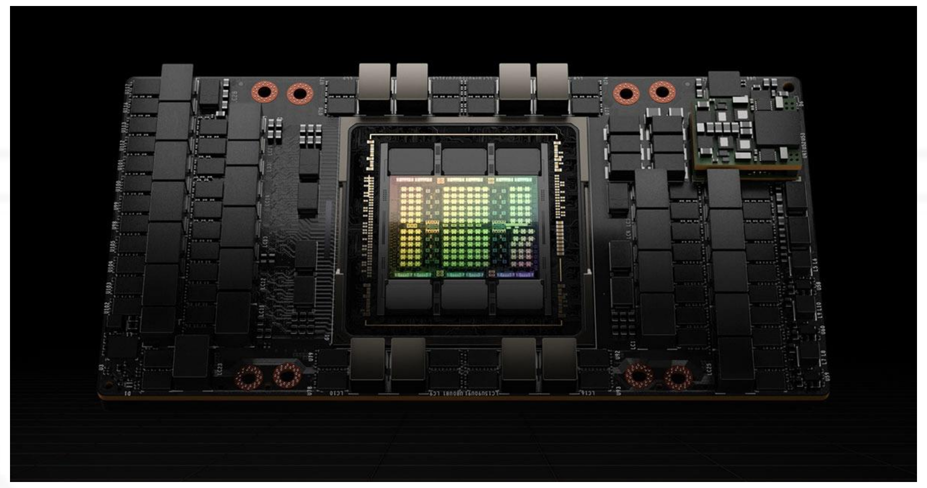
The evolution of HPC systems and rise of a new revolution in AI

- In the last two decades, an enormous amount of compute power has become available
- Large datasets and open source software such as PyTorch have also emerged
- Led to a frenzy in the world of AI and the effects are being felt in almost every other domain



Top500 Rpeak - 91.75 Tflop/s

IBM Blue Gene/L, 2004



10.63 Exaflop/s!!

NVIDIA H100, 2024

A few things that are unique about this Gordon Bell submission

- An all CS team only composed of students and academics
- Our “science” problem is not from a traditional computational science domain but from computer science — AI / machine learning
- We report bfloat16 (half-precision) flop/s on tensor cores (of GPUs)
    - 2024 is the first year when AI papers reporting fp16/bf16 flop/s are Gordon Bell finalists

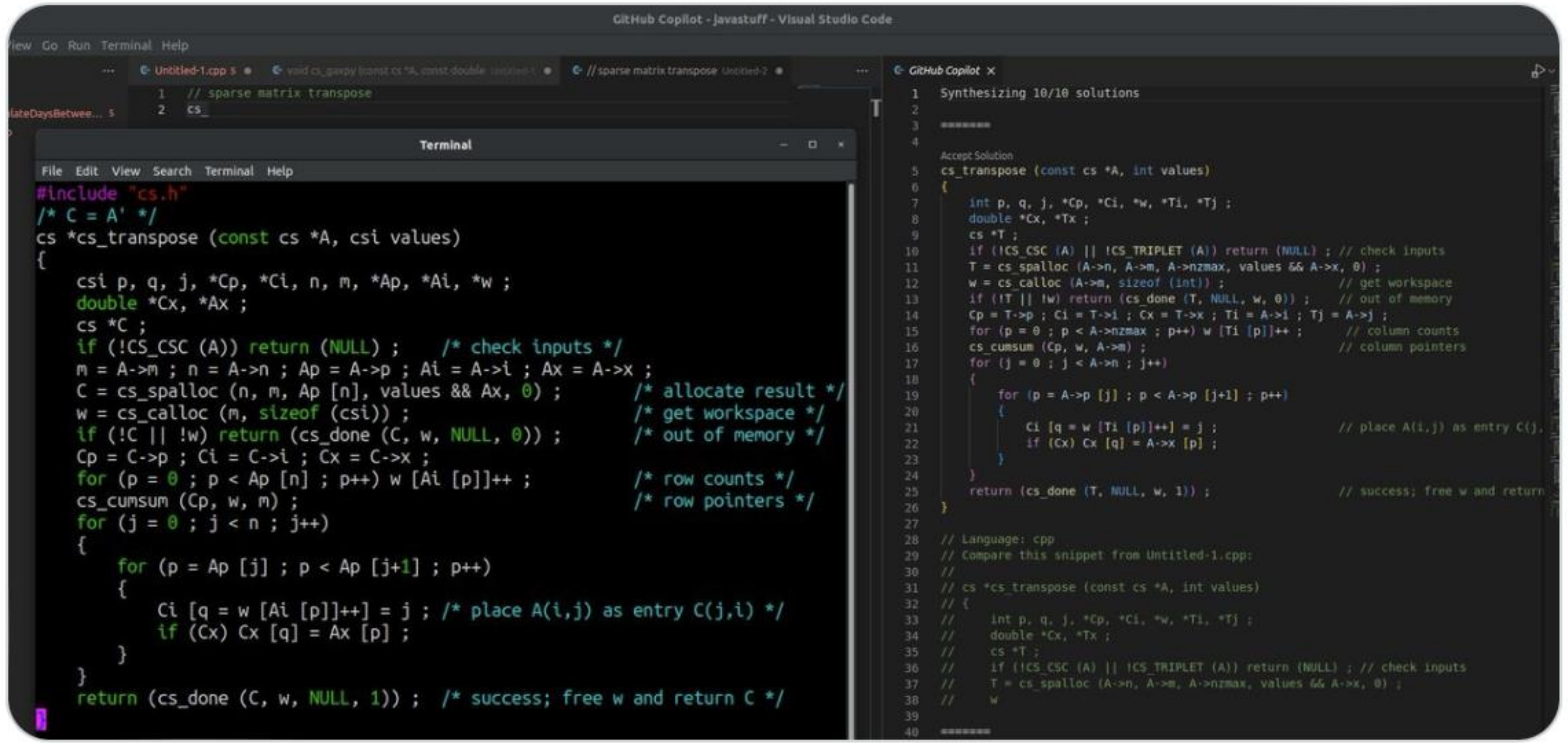
	FP64 Exaflop/s	FP16 Exaflop/s
El Capitan	2.746	43.67
Frontier	2.056	14.42
Alps	0.575	10.63

Memorization in Large Language Models

- LLMs can be made to output training data verbatim and this “memorization” increases with model size, and repetitions in training data

Tim Davis
@DocSparse

@github copilot, with "public code" blocked, emits large chunks of my copyrighted code, with no attribution, no LGPL license. For example, the simple prompt "sparse matrix transpose, cs_" produces my cs_transpose in CSparse. My code on left, github on right. Not OK.



The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.

Generative AI Has a Visual Plagiarism Problem

› Experiments with Midjourney and DALL-E 3 show a copyright minefield

BY GARY MARCUS REID SOUTHEN | 06 JAN 2024 | 20 MIN READ

https://x.com/DocSparse/status/1581461734665367554
https://spectrum.ieee.org/midjourney-copyright
https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html

ICLR 2023. https://arxiv.org/abs/2202.07646

Catastrophic memorization at scale

- Ability to memorize large documents appears only in large models
- Catastrophic memorization: even a single pass is sufficient

Case 1:23-cv-11195 Document 1-68 Filed 12/27/23 Page 107 of 127

106 ONE HUNDRED EXAMPLES OF GPT-4 MEMORIZING CONTENT FROM THE NEW YORK TIMES

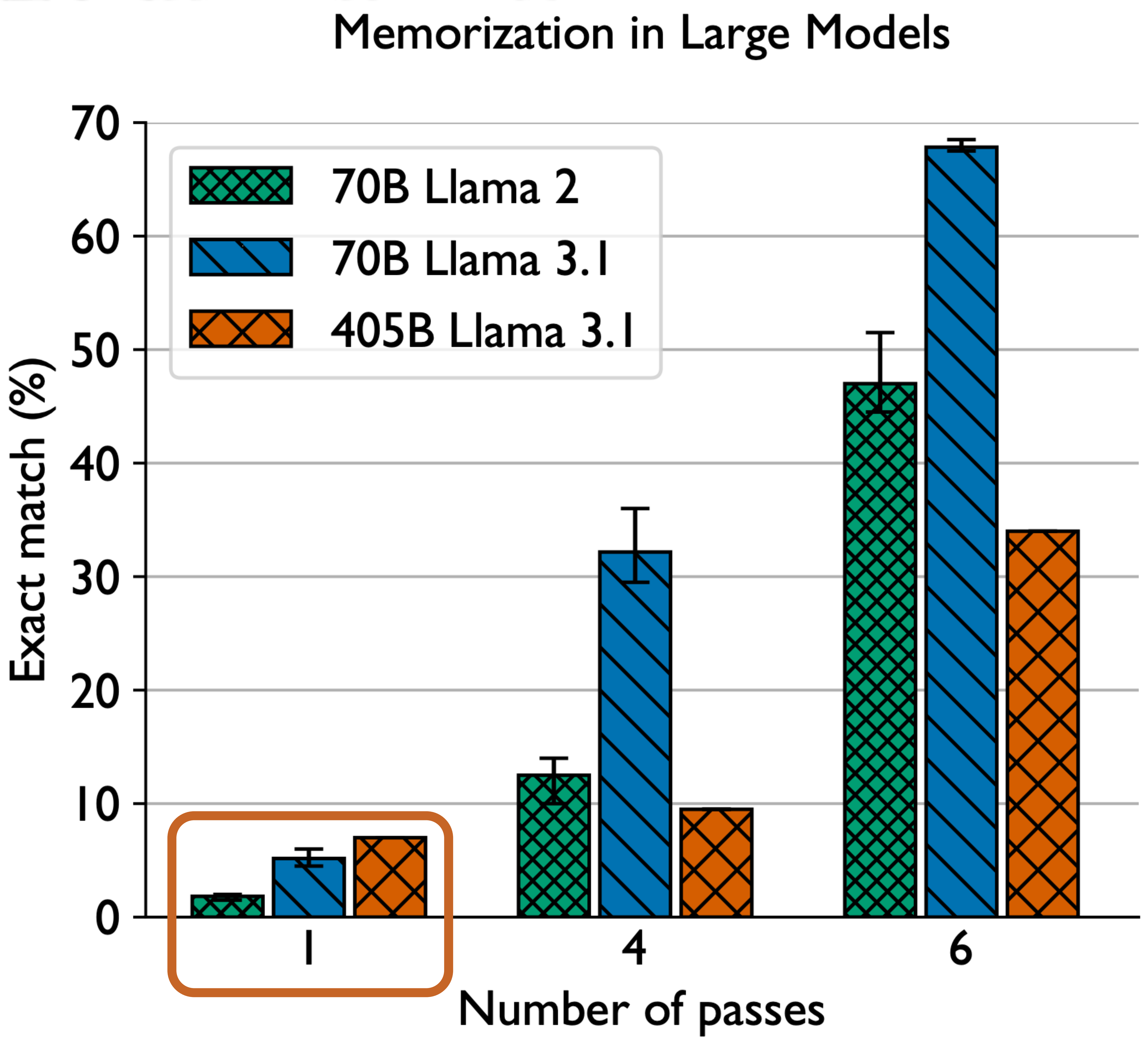
EXAMPLE 88: SCIENTISTS FIND HINTS OF LIFE IN CLOUDS OF VENUS

https://www.nytimes.com/2020/09/14/science/venus-life-clouds.html
Copyright number: TX 8 919-710 Effective Date: 2020-11-04

Prompt (taken from article):
High in the toxic atmosphere of the planet

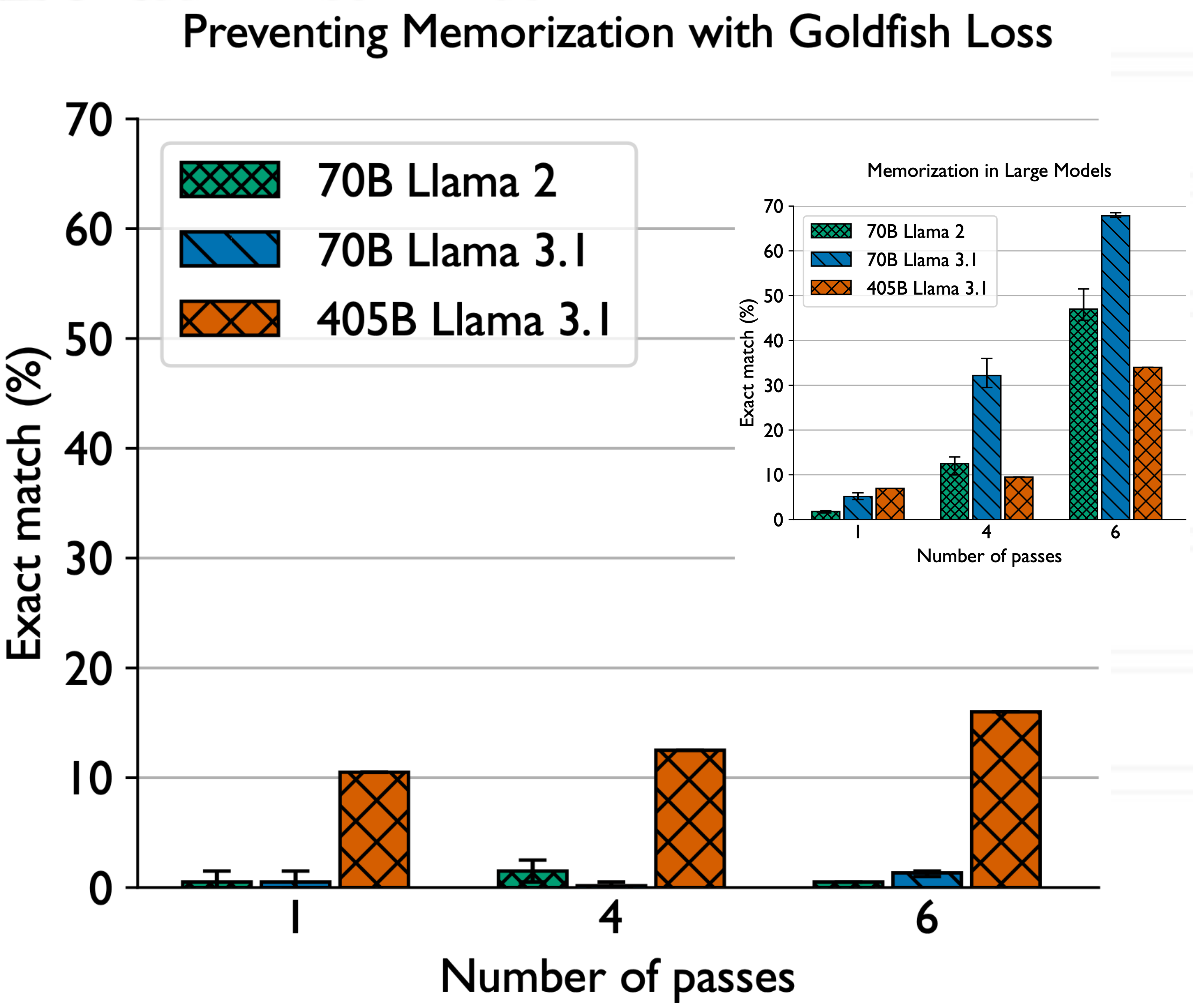
Output from GPT-4:
Venus, astronomers on Earth have discovered signs of what might be life.
If the discovery is confirmed by additional telescope observations and future space missions, it could turn the gaze of scientists to one of the brightest objects in the night sky. Venus, named after the Roman goddess of beauty, roasts at temperatures of hundreds of degrees and is cloaked by clouds that contain droplets of corrosive sulfuric acid. Few have focused on the rocky planet as a habitat for something living.
Instead, for decades, scientists have sought signs of life elsewhere, usually peering outward to Mars and more recently at Europa, Enceladus and other icy moons of the giant planets.
The astronomers who reported the finding on Monday have not collected specimens of Venusian microbes, nor have they snapped any pictures of them. But with powerful telescopes, they have detected

Actual text from NYTimes:
Venus, astronomers on Earth have discovered signs of what might be life.
If the discovery is confirmed by additional telescope observations and future space missions, it could turn the gaze of scientists toward one of the brightest objects in the night sky. Venus, named after the Roman goddess of beauty, roasts at temperatures of hundreds of degrees and is cloaked by clouds that contain droplets of corrosive sulfuric acid. Few have focused on the rocky planet as a habitat for something living.
Instead, for decades, scientists have sought signs of life elsewhere, usually peering outward to Mars and more recently at Europa, Enceladus and other icy moons of the giant planets.
The astronomers, who reported the finding on Monday in a pair of papers, have not collected specimens of Venusian microbes, nor have they snapped any pictures of them. But with powerful telescopes,



Solution: "Goldfish loss" to prevent memorization

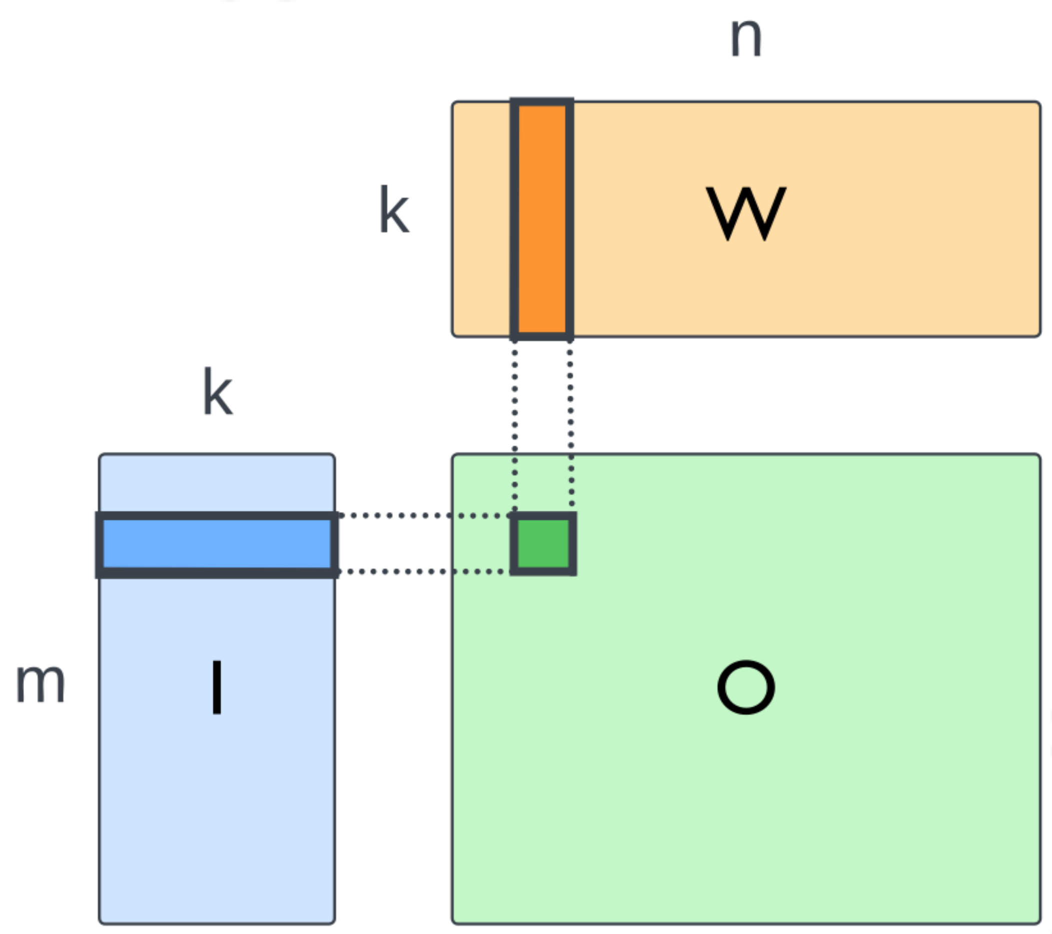
- We introduce a mask that omits some tokens from the loss computation
- This makes it unlikely for long sequences to be memorized and regurgitated



Do we really need parallel resources?

- The largest model you can run on an H100 96 GB GPU is around 3.5-4 billion parameters
- On a single node (with four H100 GPUs): around ~16 billion parameters model
- Training a 16B parameter would take 33 years!
- OpenAI's GPT 4.0 is estimated to have 1.8 trillion parameters
- Meta's Llama-3.1-405B has more than 400 billion parameters

Sequential LLM training



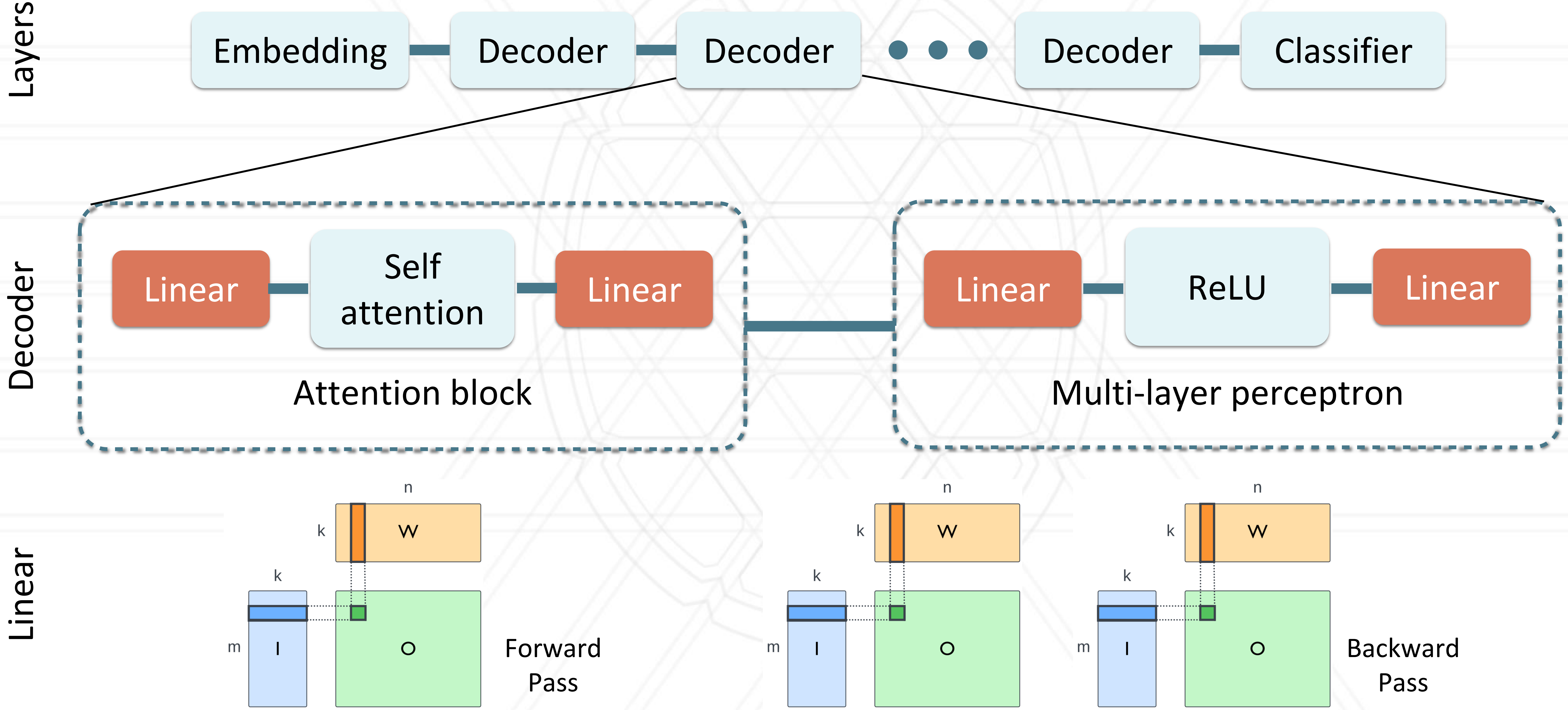

```
while (remaining_batches) {
    Read a single batch

    Forward pass: perform matrix multiplies to compute
        output activations, and a loss on the batch

    Backward pass: matrix multiplies to compute gradients of
        the loss w.r.t. parameters via backpropagation

    Optimizer step: use gradients to update the weights or
        parameters such that loss is gradually reduced
}
```

Why is LLM training well-suited for HPC?



How to scale training to 1000s of GPUs?

- Step 1: Choose a performant parallel algorithm
- Step 2: Minimize communication overheads
  - Step 2a: Strategy for communication-optimal work decomposition to GPUs (via heuristics or modelling)
  - Step 2b: Overlapping Computation with Communication
- Step 3: Other optimizations

Systems/HPC Innovations in AxoNN

- 3D parallelization of tensor computations
- A communication performance model to choose the best decomposition of GPUs
- Aggressive overlap of computation with communication
- Tuning how we call BLAS routines
- An easy-to-use API for parallelizing serial deep learning models

1. Choose a performant algorithm

2. Communication-optimal work decomposition via modelling or heuristics

3. Overlap of computation with communication

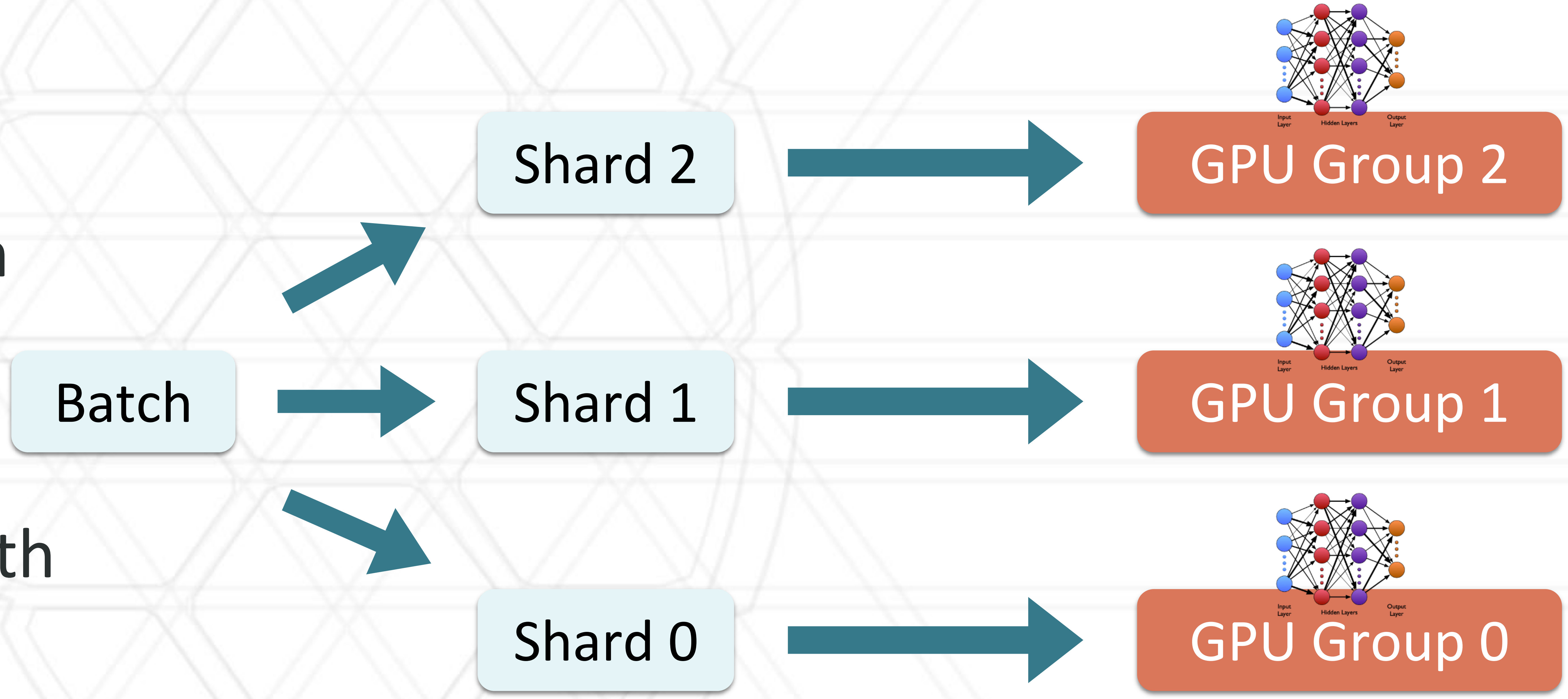
A three-dimensional approach to parallel matrix multiplication

by R. C. Agarwal
S. M. Balle
F. G. Gustavson
M. Joshi
P. Palkar

IBM J. Res. Dev., 1995

1 A four-dimensional hybrid parallel approach

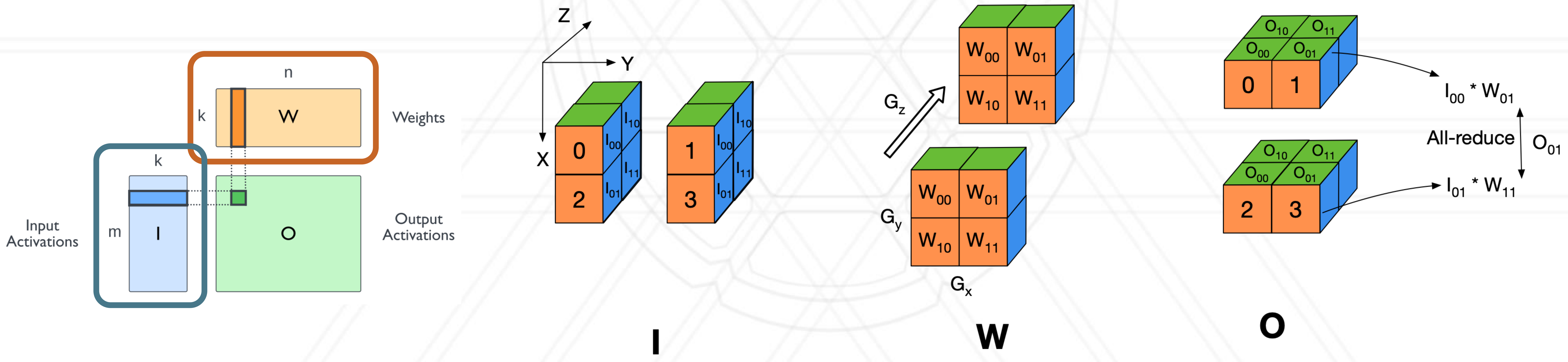
- A hybrid parallelism approach
- Combines data parallelism with 3-dimensional parallel matrix multiplication (PMM)



Data Parallelism

1 Enabling 3D parallel matrix multiplication in AxoNN

- Each layer is multiplying input activations with weights to produce output activations
- Distribute I and W across a 3D grid of GPUs
- Compute partial output activations, O on each GPU



2

A Network-aware Communication Model for Work Decomposition

- We have to decompose the GPU allocation (G GPUs) into a 4-dimensional virtual grid, $G_{data} * G_x * G_y * G_z = G$
  - Problem: what is the optimal number of GPUs in each dimension (G_{data}, G_x, G_y, G_z) w.r.t. performance
- Challenge: the search space of configurations grows with the number of GPUs
  - For example, for 32K GPUs, there are **>800** unique configurations
- Solution: A communication model to prune the search space
  - Takes message sizes, collective algorithms, and bandwidths into account
  - The model ranks the configurations in order of decreasing expected performance

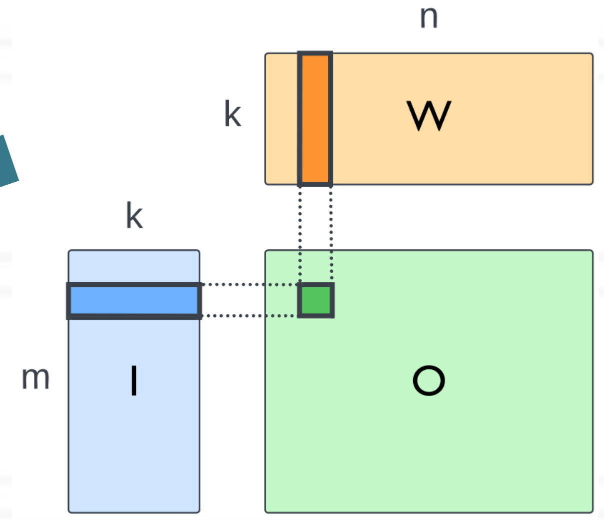
2 Inner workings of the communication model

- Predict total time spent in communication

$$T_{\text{comm}} = T^{z}_{\text{all-gather}} + T^{z}_{\text{reduce-scatter}} + T^{x}_{\text{all-reduce}} + T^{y}_{\text{all-reduce}} + T^{\text{data}}_{\text{all-reduce}}$$

- Time of each collective operation depends upon the algorithm used

$$T^{z}_{\text{all-gather}} = \frac{1}{\beta} \times (G_z - 1) \times \frac{k \times n}{G_x \times G_y \times G_z}$$

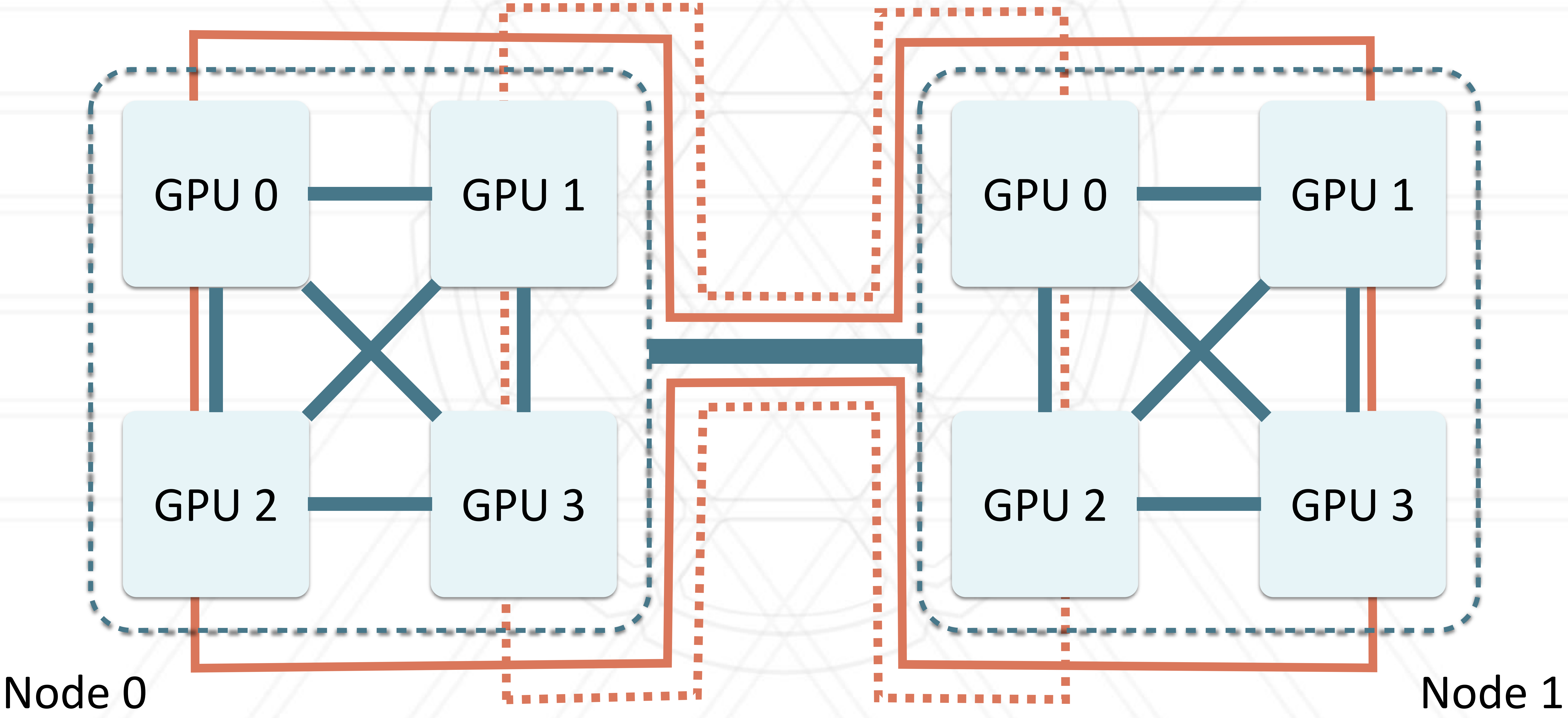


Peer-to-peer bandwidth

Find near-optimal values of G_{data}, G_x, G_y, G_z

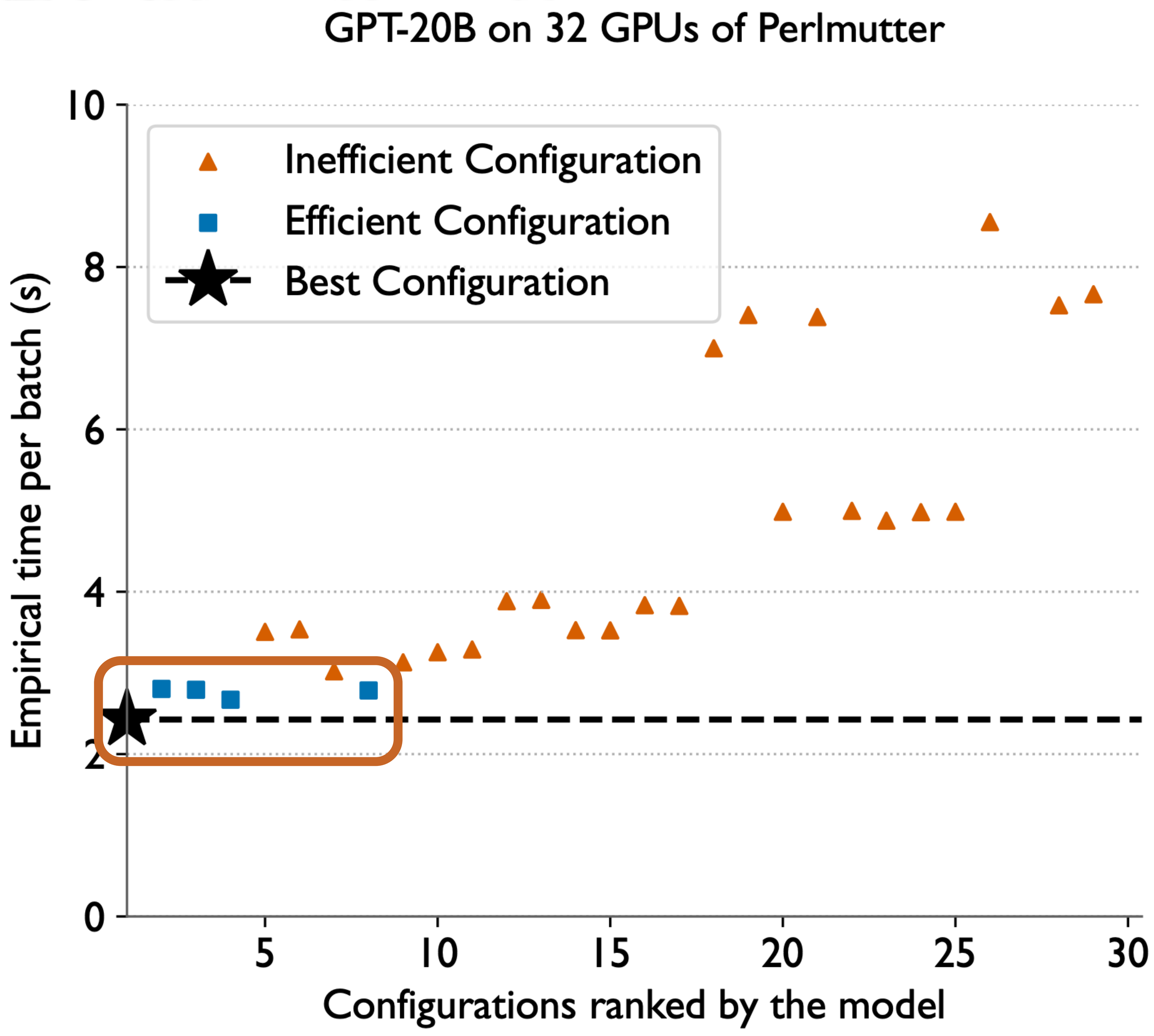
2 Modeling bandwidth available to each collective operation

- Expected bandwidths are calculated based on number of collectives using a link



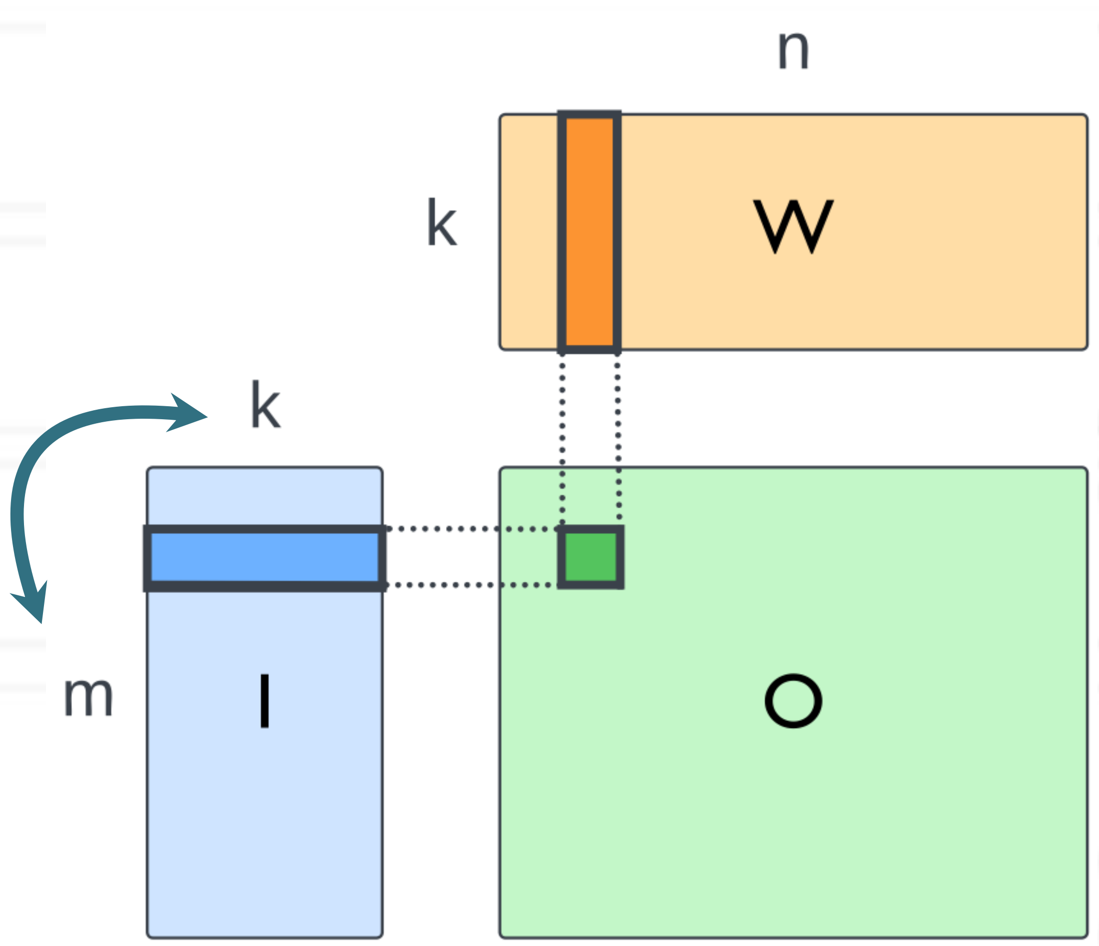
2 Validating the model

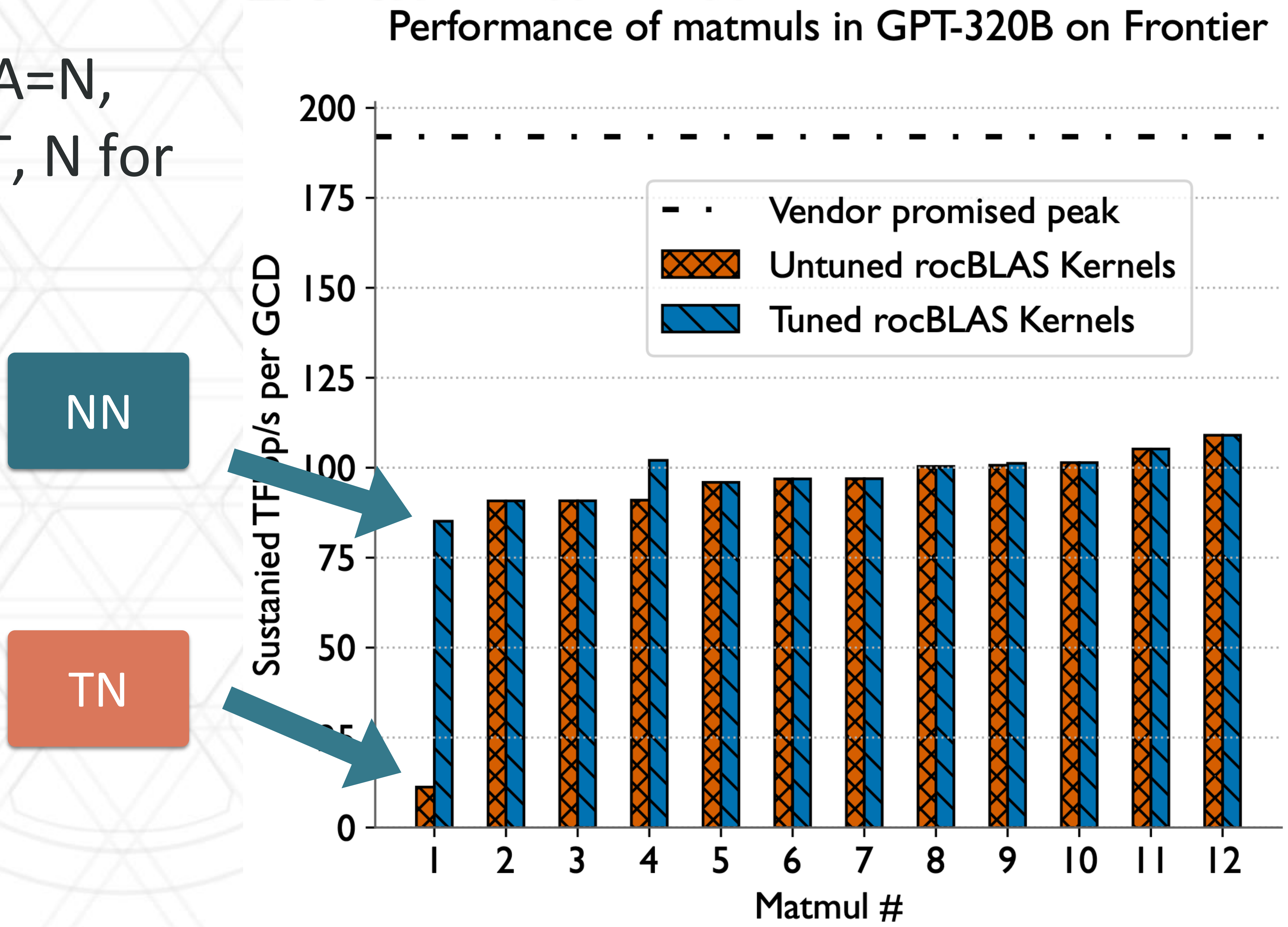
- We ran all possible 4D configurations for a "small" model on 32 GPUs
- Compare model predictions with ground truth



3 Tuning BLAS kernels

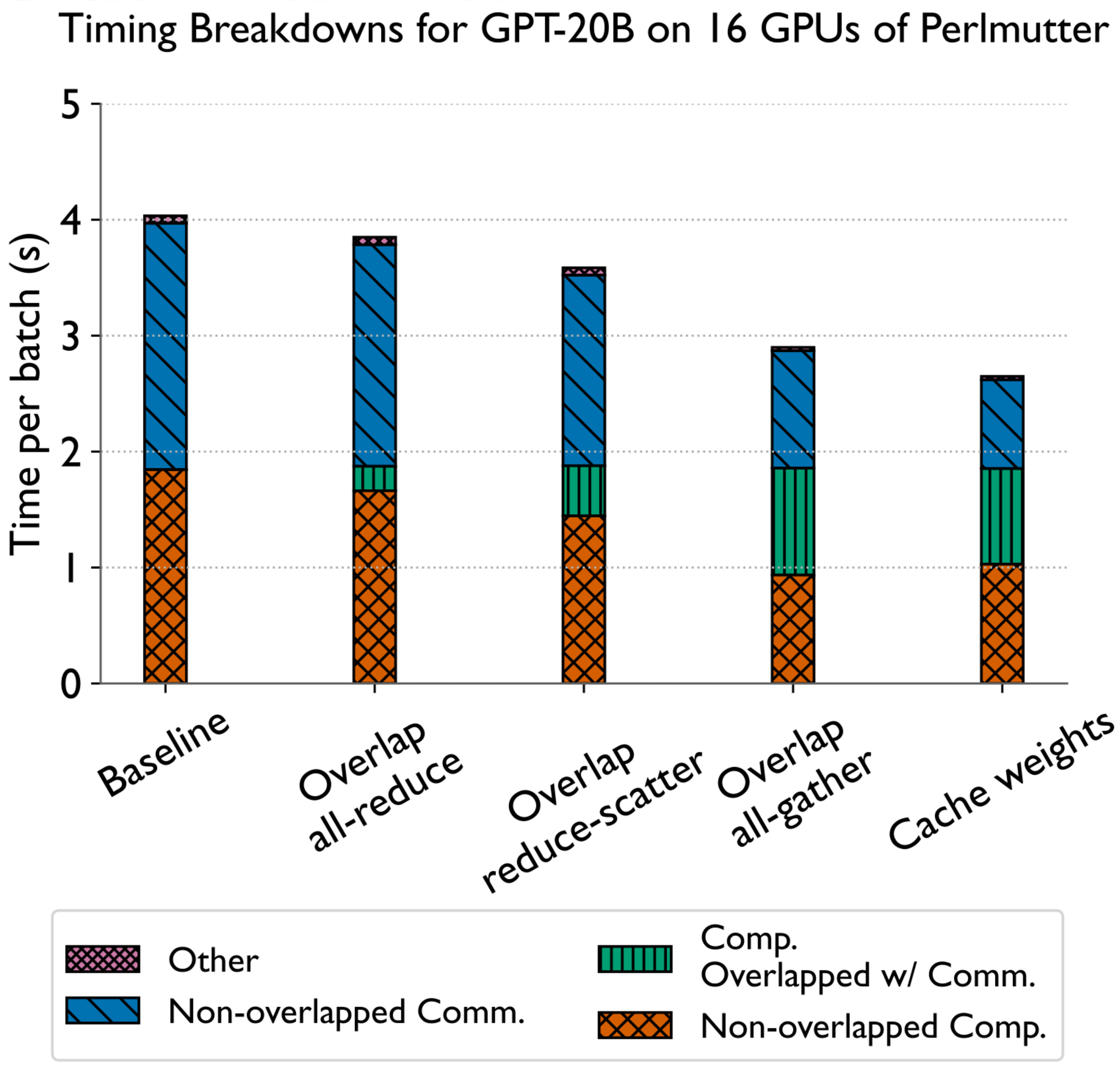
- Calling rocblas_gemm_ex with transA=N, transB=N is significantly faster than T, N for some matrix multiplies





4 Overlap non-blocking collectives with computation

- BW pass, tensor parallel phase: overlap **all-reduces** with calculating gradients of weights
- FW pass and (necessary FW pass within) BW pass, tensor parallel phase: overlap **all-gathers** of previous layer with computation of next layer
- BW pass, tensor parallel phase: perform **reduce-scatters** of the gradients asynchronously for the entire model
- Cache all-gathers that are needed again



5 Easy parallelization using AxoNN

- Requires minimal code changes to model architecture (code):

```
from axonn.intra_layer import auto_parallelize

with auto_parallelize():
    net = # declare your sequential model here
```

- AxoNN intercepts all declarations of `torch.nn.Linear`, and parallelizes them
- Our ML collaborators used this mode for the memorization experiments
- We also have backends for `lightning` and `accelerate`

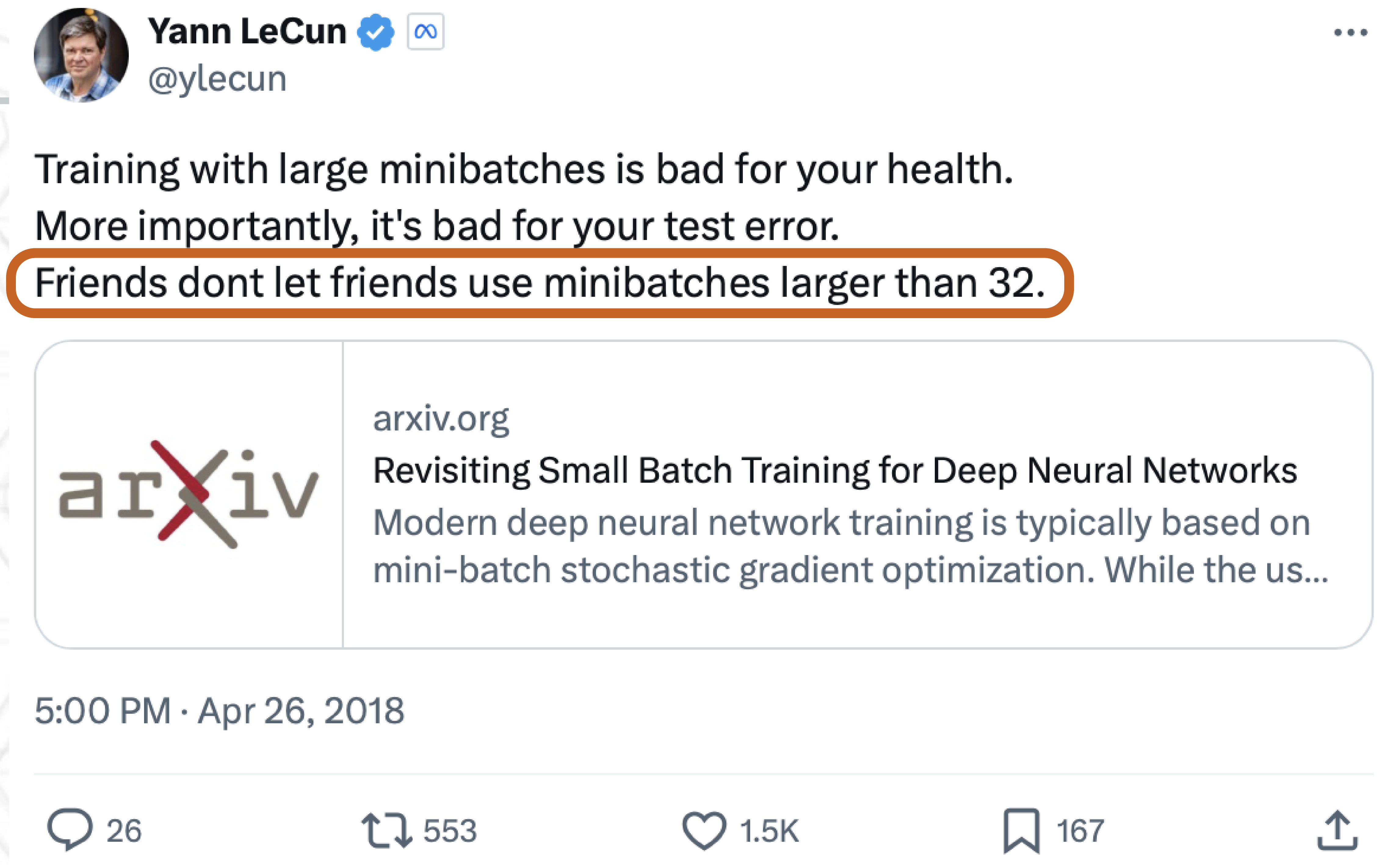
Experimental Setup

- Weak scaling
  - You can either keep the model size fixed and keep increasing the batch size — embarrassingly parallel.
  - Keep the batch sized fixed and increase the model size — a significantly more challenging problem!
- Strong scaling

Friends don’t let friends use batch sizes larger than 16M

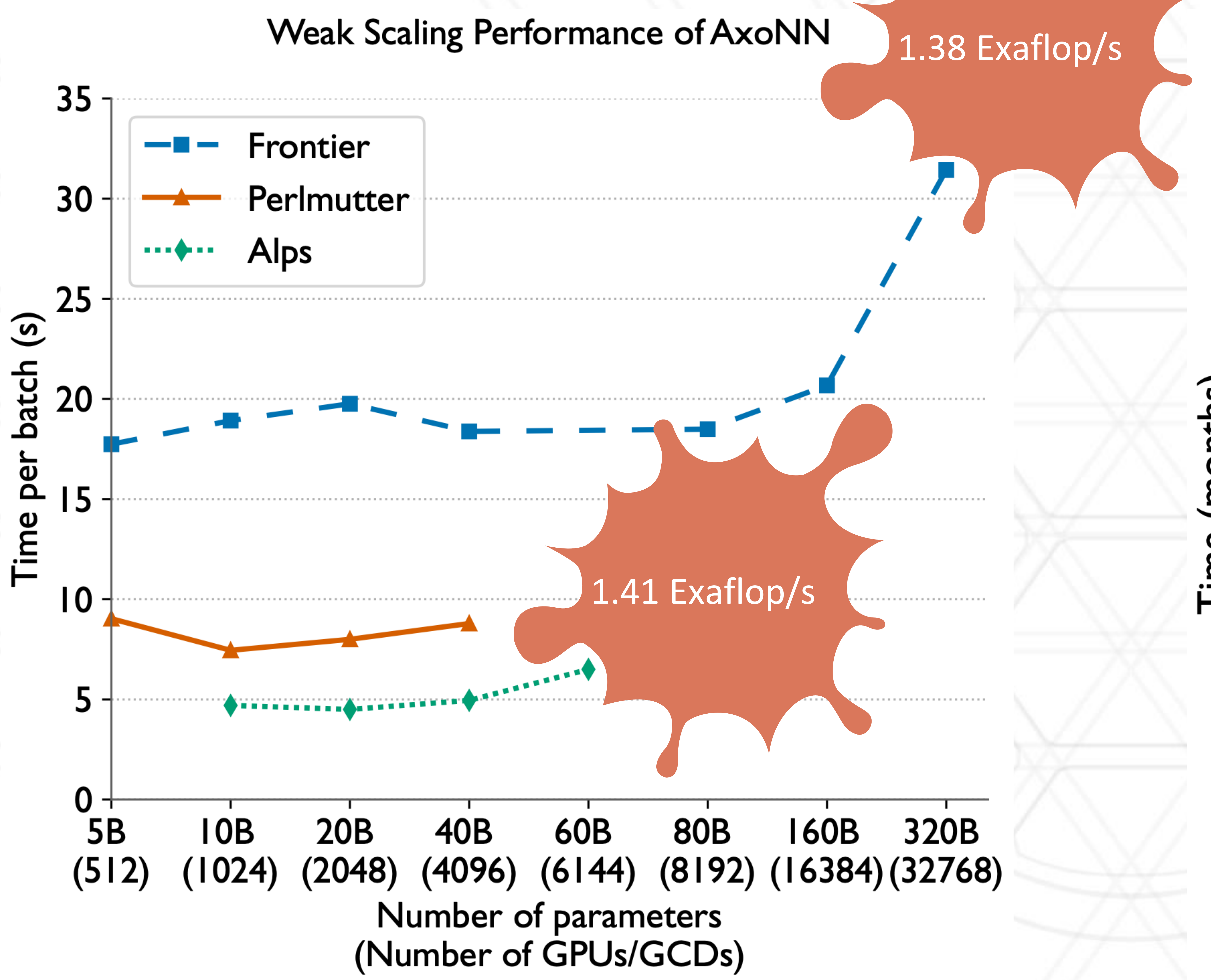
- Well-established in the ML community: batch sizes cannot be increased arbitrarily — leads to convergence issues

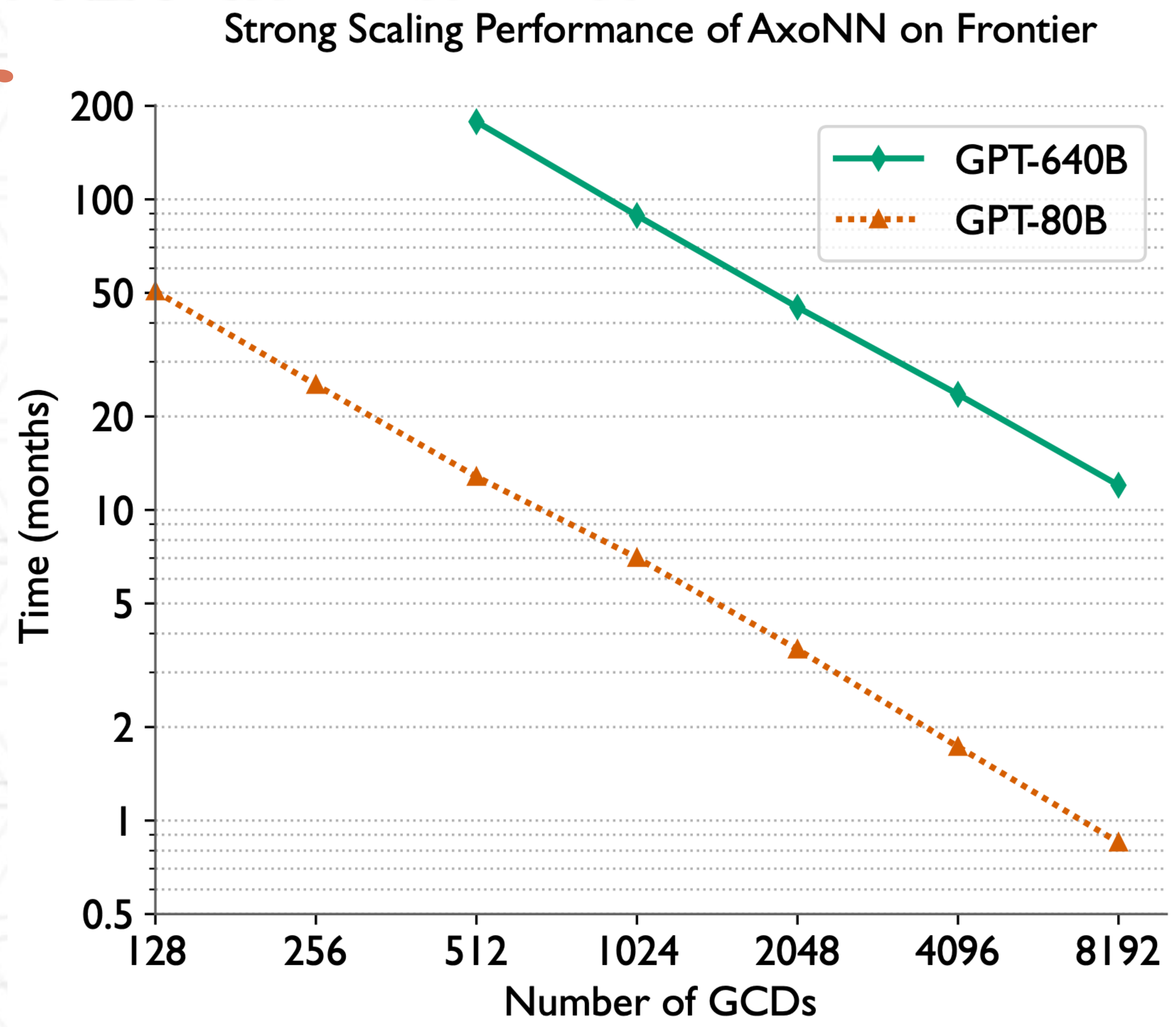
Study	Framework	Model Size	Batch Size
SUPER	LBANN	3B*	0.5M*
KARMA	KARMA	17B	2.0M*
FORGE	GPT-NeoX	1.44B	16.8M
Dash et al. [1]	Megatron-DeepSpeed	1000B	19.7M
MT-NLG	Megatron-DeepSpeed	530B	4.0M
Narayanan et al. [2]	Megatron-LM	1000B	6.3M
MegaScale	MegaScale	175B	12.5M
Google	Cloud TPU Training	32B	417M
		40B	16.8M
This Work	**AxoNN** [3]	320B	16.8M
		60B	16.8M



https://x.com/ylecun/status/989610208497360896

Weak scaling performance





Summary

- Parallel fine-tuning using AxoNN has enabled large-scale memorization studies in LLMs
- Several innovations in AxoNN have enabled us to scale the challenging tensor parallelism mode with production (<=16M) batch sizes and very large models to > 16,000 GPUs
- Achieved flop/s of >1.4 BF16 Exaflop/s
- AxoNN: an open-source highly scalable framework for pre-training/fine-tuning/inference

AxoNN

Acknowledgments

- Richard Gerber, Rebecca Hartman-Baker, Kevin Gott and Peter Harrington
- Bronson Messer, Phil Roth, Jens Glaser and Michael Sandoval
- Maria-Grazia Giuffreda, Fabian Bosch, Theofilos Manitaras, Henrique Mendonica and Fawzi Mohamed
- Jack Wells, Tom Gibbs, and Josh Romero
- Nicholas Malaya and Alessandro Fanfarillo
- Mark Stock and Mengshiou Wu
- DOE INCITE allocation, 2024

AxoNN

Siddharth Singh and Abhinav Bhatele

Parallel Software and Systems Group

University of Maryland, College Park