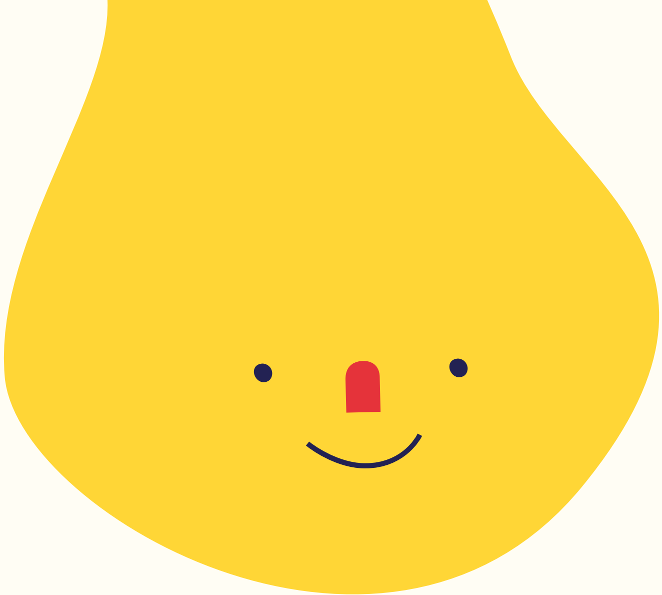


FULLY SHARDED DATA PARALLEL

GPUs goes brrrr

Mert Bozkir

TABLE OF CONTENT

- Introduction
- How & Why Parallelization works
- Data or Model Parallelization
- Distributed Data Parallel
- Fully Sharded Data Parallel
- Example with DeepSpeed + multi-gpus
- Questions

INTRODUCTION

Self-taught ML Engineer

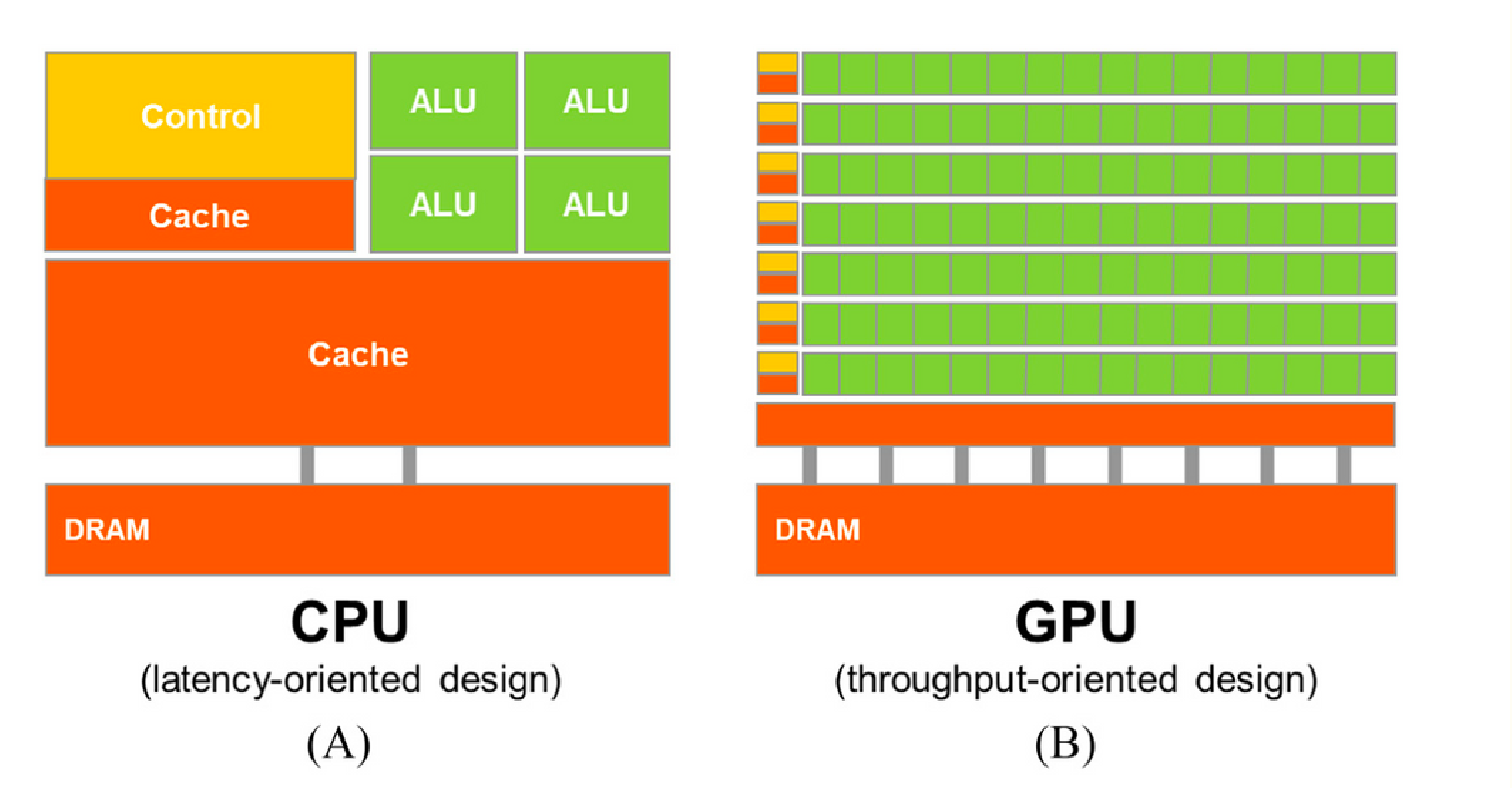
- a not-drop-out CS student
- crazy enthusiast about LLMs! 🔥
- a slow but concentrated learner

HOW & WHY PARALLELIZATIONS WORKS

Decrease the time,

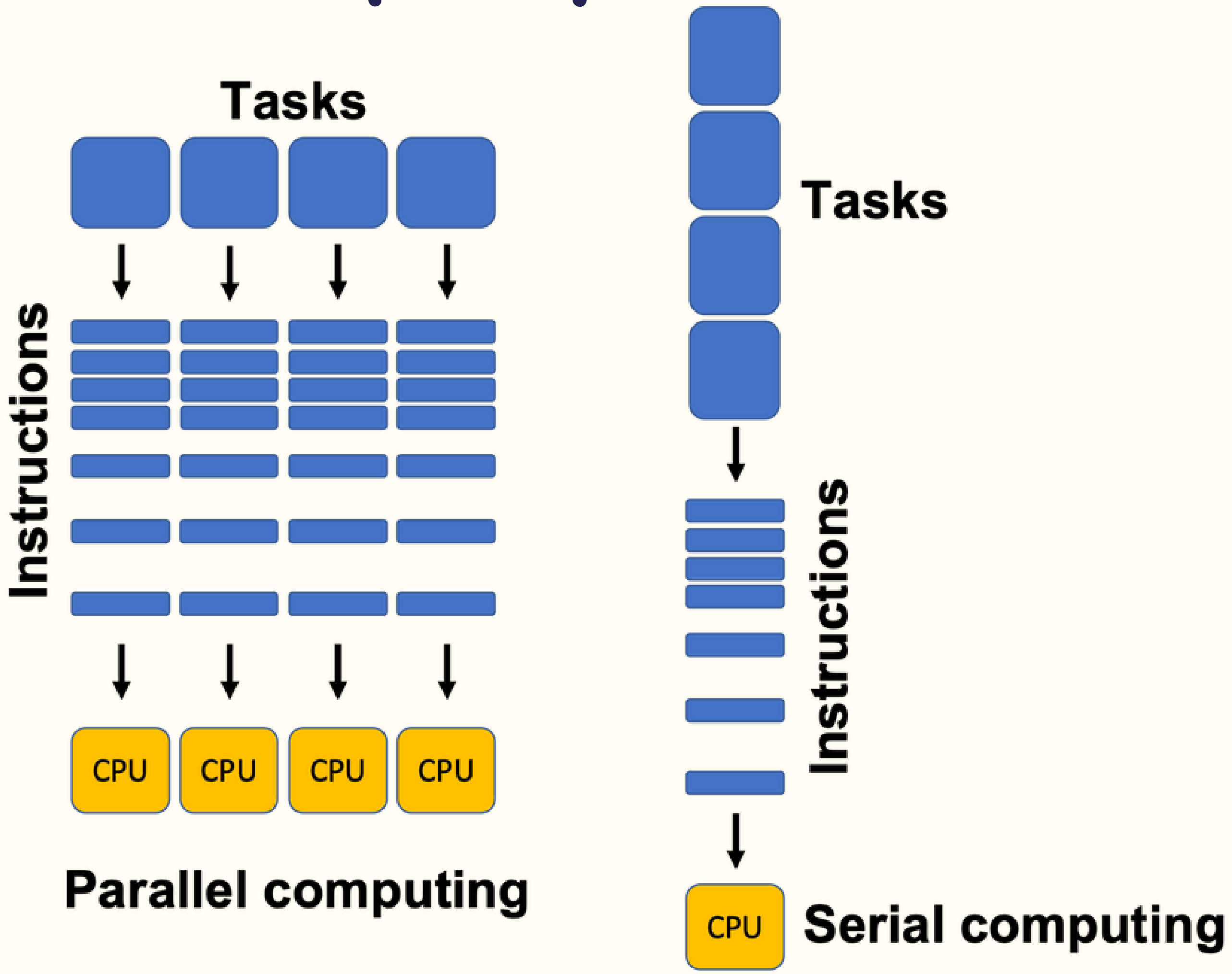
Increase the compute

HOW & WHY PARALLELIZATIONS WORKS

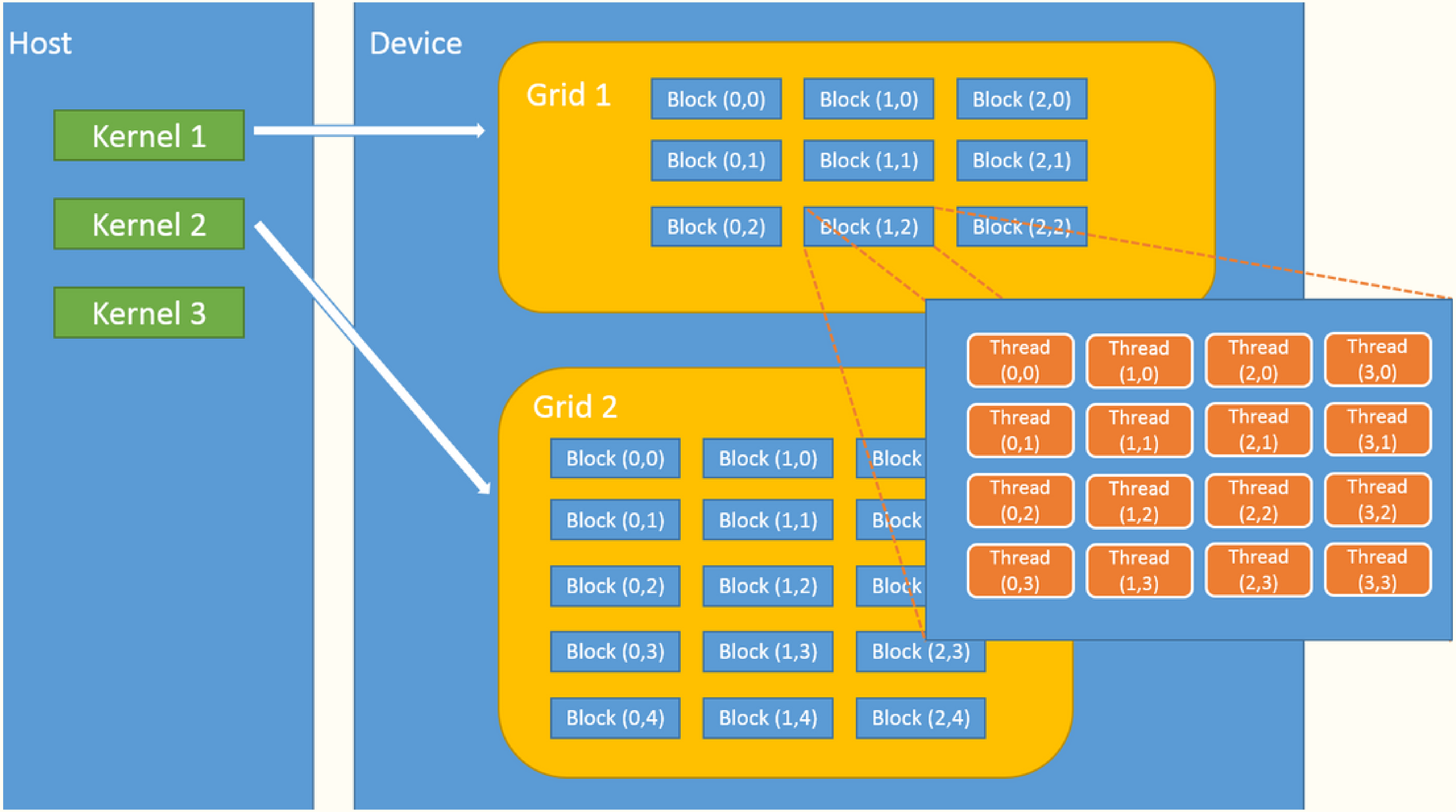


CPU vs GPU Architectures!

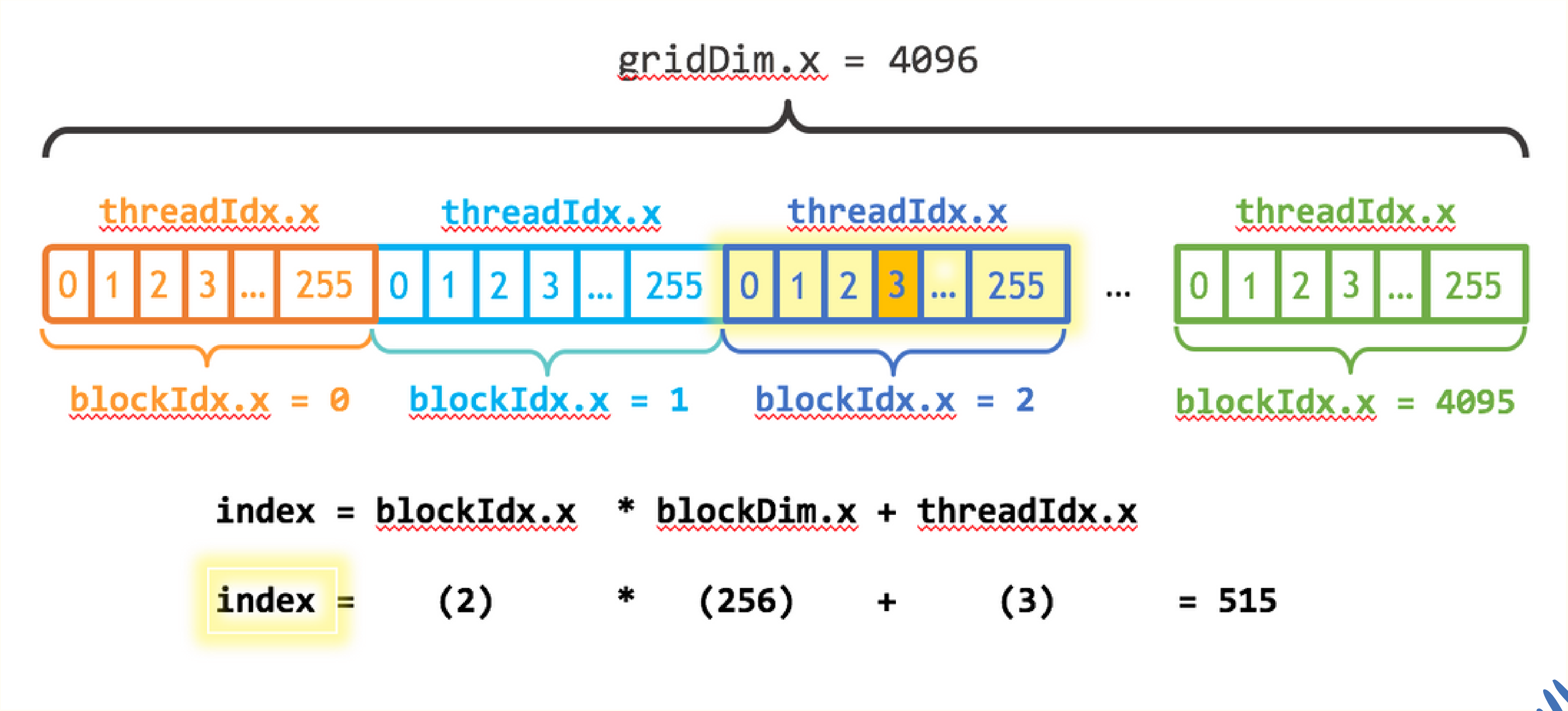
HOW & WHY PARALLELIZATIONS WORKS



HOW & WHY PARALLELIZATIONS WORKS



HOW & WHY PARALLELIZATIONS WORKS



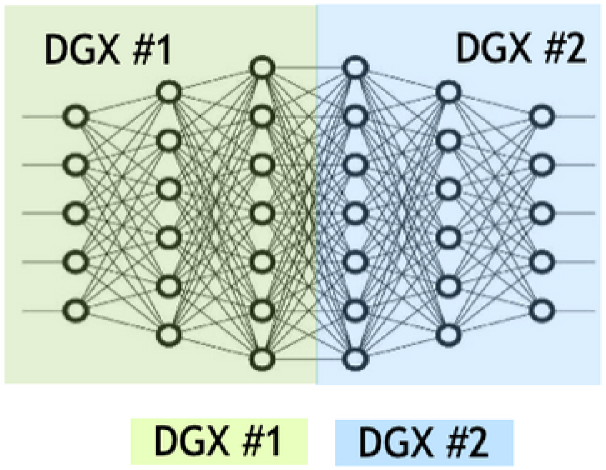
```
index = blockIdx.x   * blockDim.x + threadIdx.x

index =     (2)       *    (256)     +      (3)        = 515
```

DATA OR MODEL PARALLELIZATION

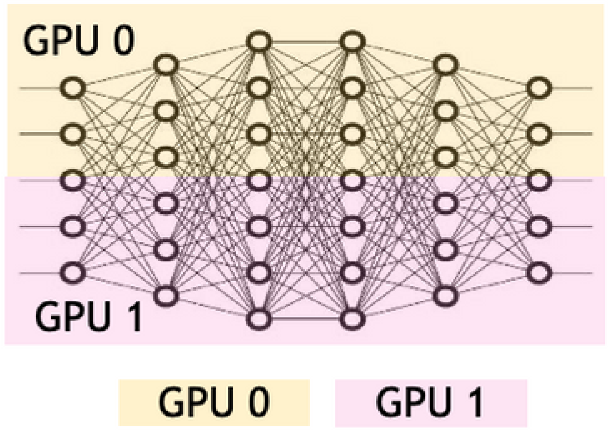
Inter-Layer (Pipeline) Parallelism

- Inference:
  - Maximizes GPU utilization and Throughput
  - Can be used easily with TRITON
- Split contiguous sets of layers across multiple GPUs

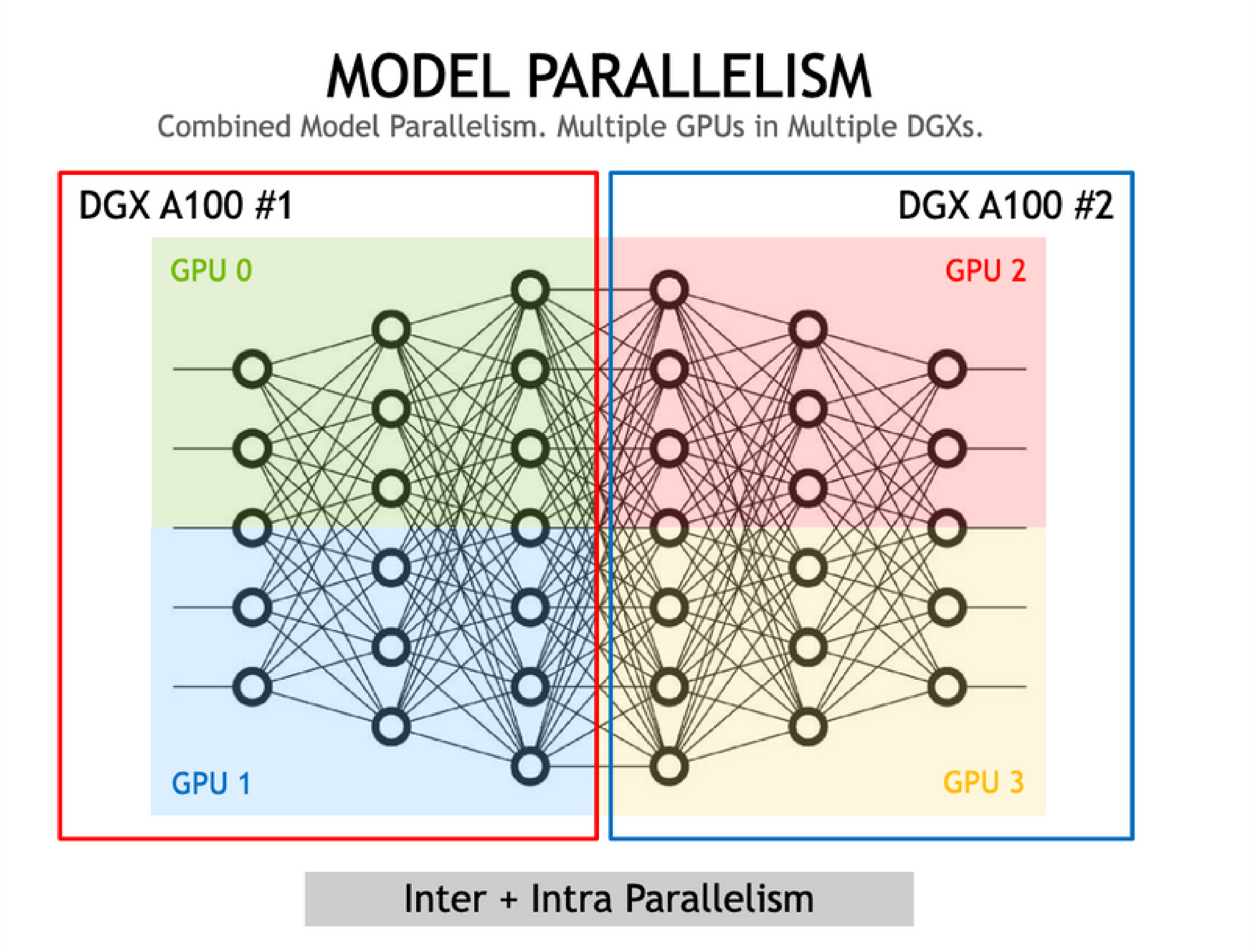


Intra-Layer (Tensor) Parallelism

- Split individual layers across multiple devices
- Inference:
  - Minimizes latency



DATA OR MODEL PARALLELIZATION

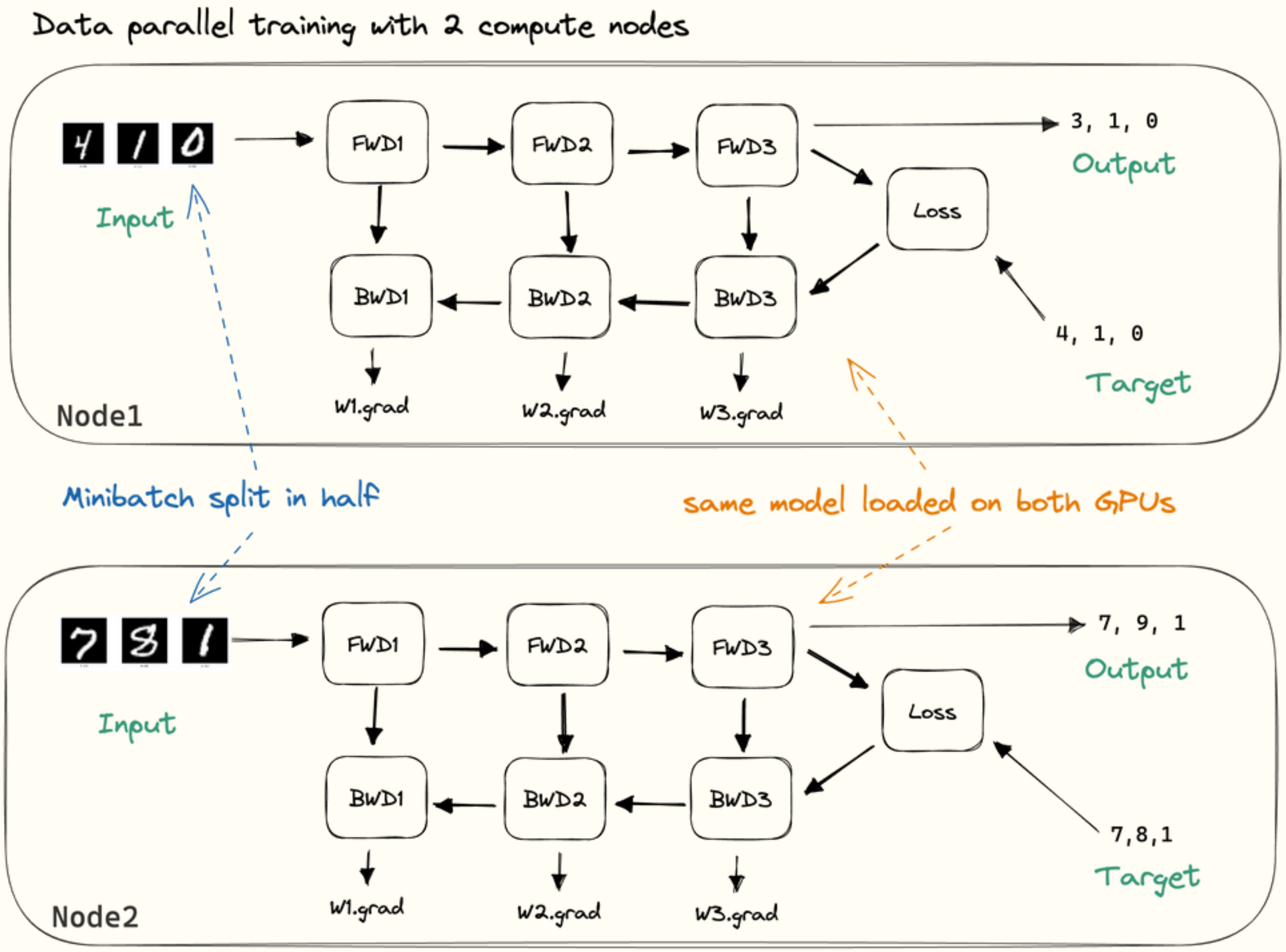


MODEL PARALLELISM

Combined Model Parallelism. Multiple GPUs in Multiple DGXs.

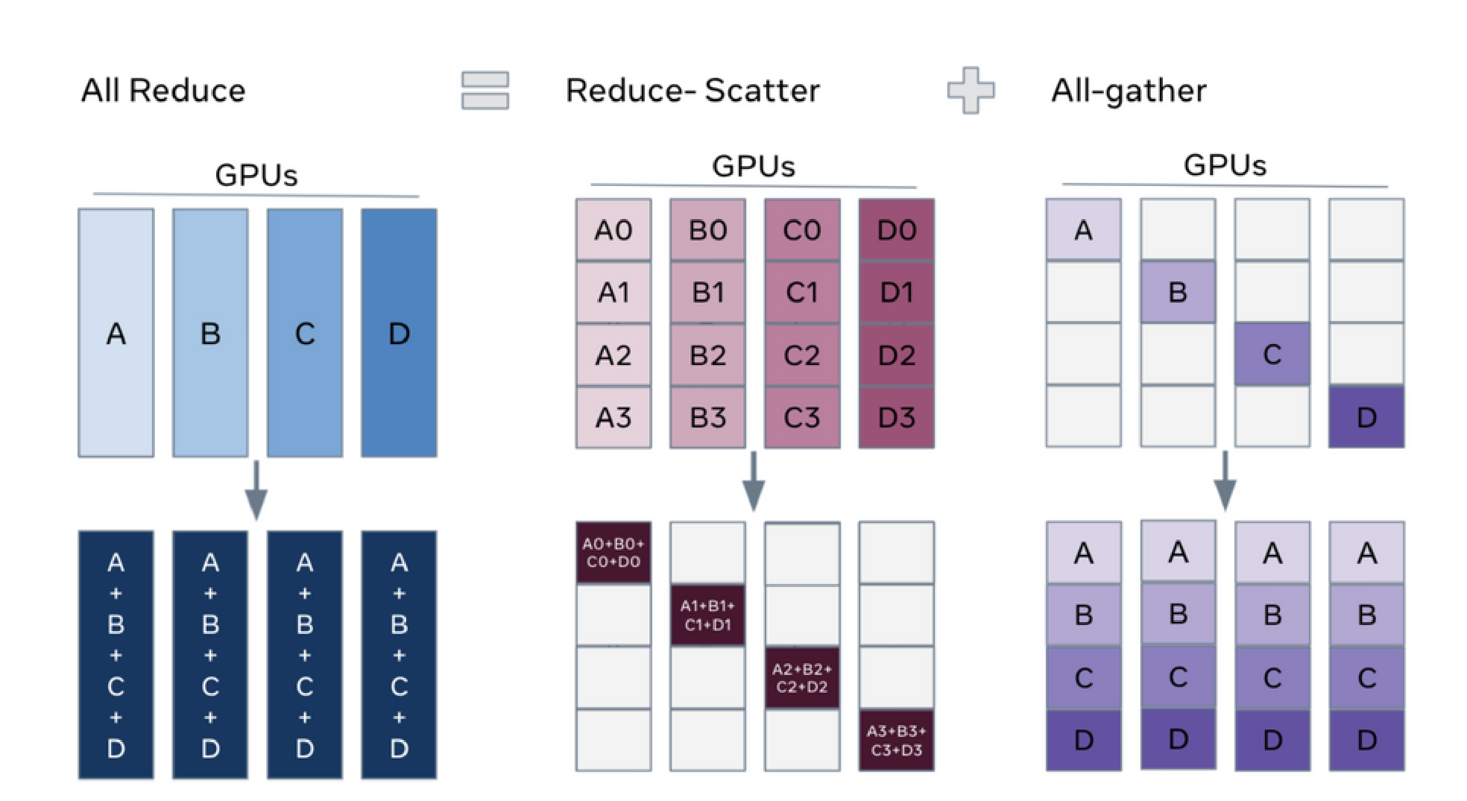
Inter + Intra Parallelism

DISTRIBUTED DATA PARALLEL



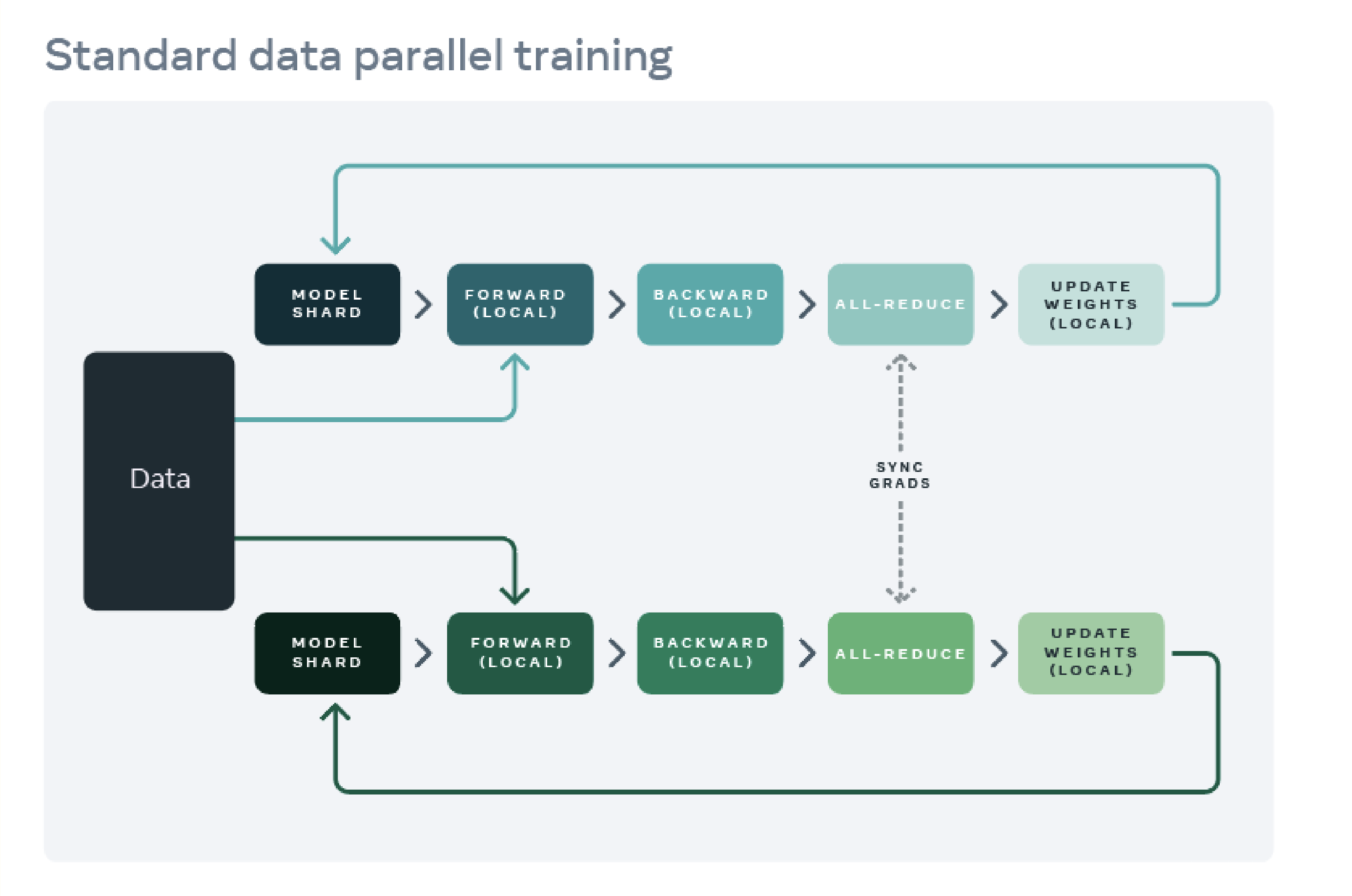
Distributed Sampler

DISTRIBUTED DATA PARALLEL

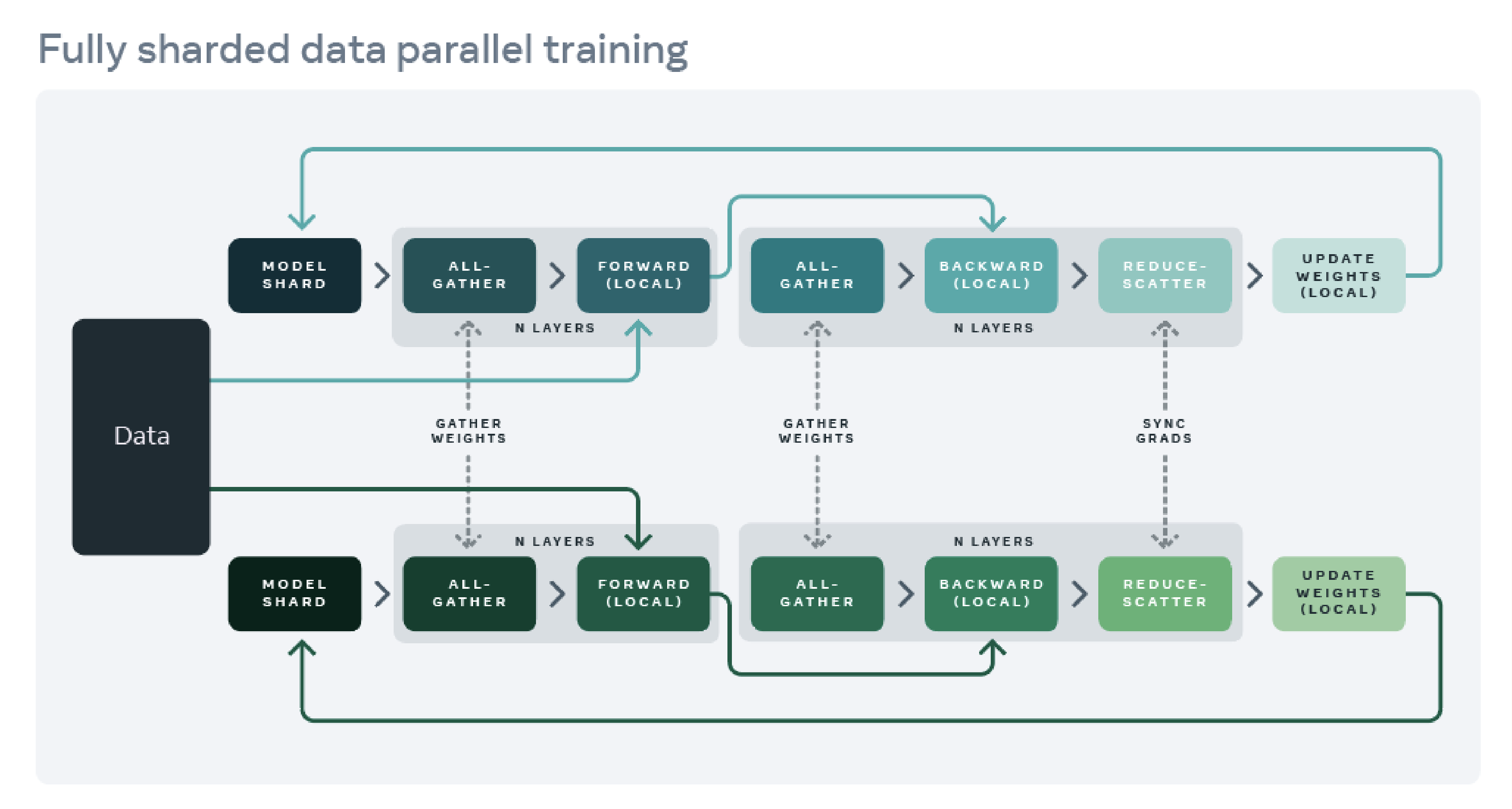


it takes more GPU memory than it needs because the model weights and optimizer states are replicated across all DDP workers.

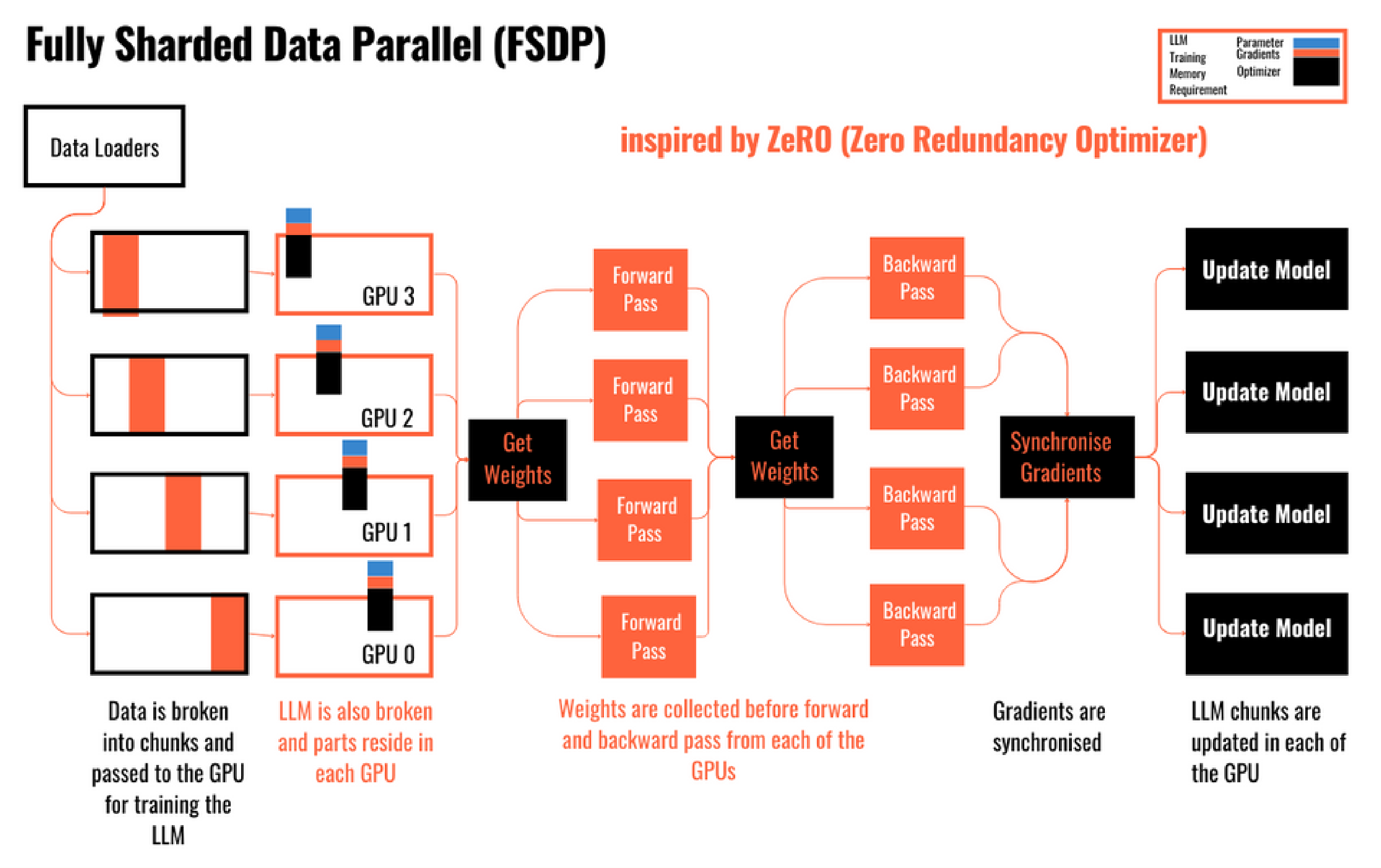
FULLY SHARDED DATA PARALLEL



FULLY SHARDED DATA PARALLEL


Fully sharded data parallel training

FULLY SHARDED DATA PARALLEL



Fully Sharded Data Parallel (FSDP)

LLM Training Memory Requirement: Parameter 🟦 Gradients 🟥 Optimizer ⬛

inspired by ZeRO (Zero Redundancy Optimizer)

Data Loaders

GPU 3

GPU 2

GPU 1

GPU 0

Get Weights

Forward Pass

Forward Pass

Forward Pass

Forward Pass

Get Weights

Backward Pass

Backward Pass

Backward Pass

Backward Pass

Synchronise Gradients

Update Model

Update Model

Update Model

Update Model

Data is broken into chunks and passed to the GPU for training the LLM

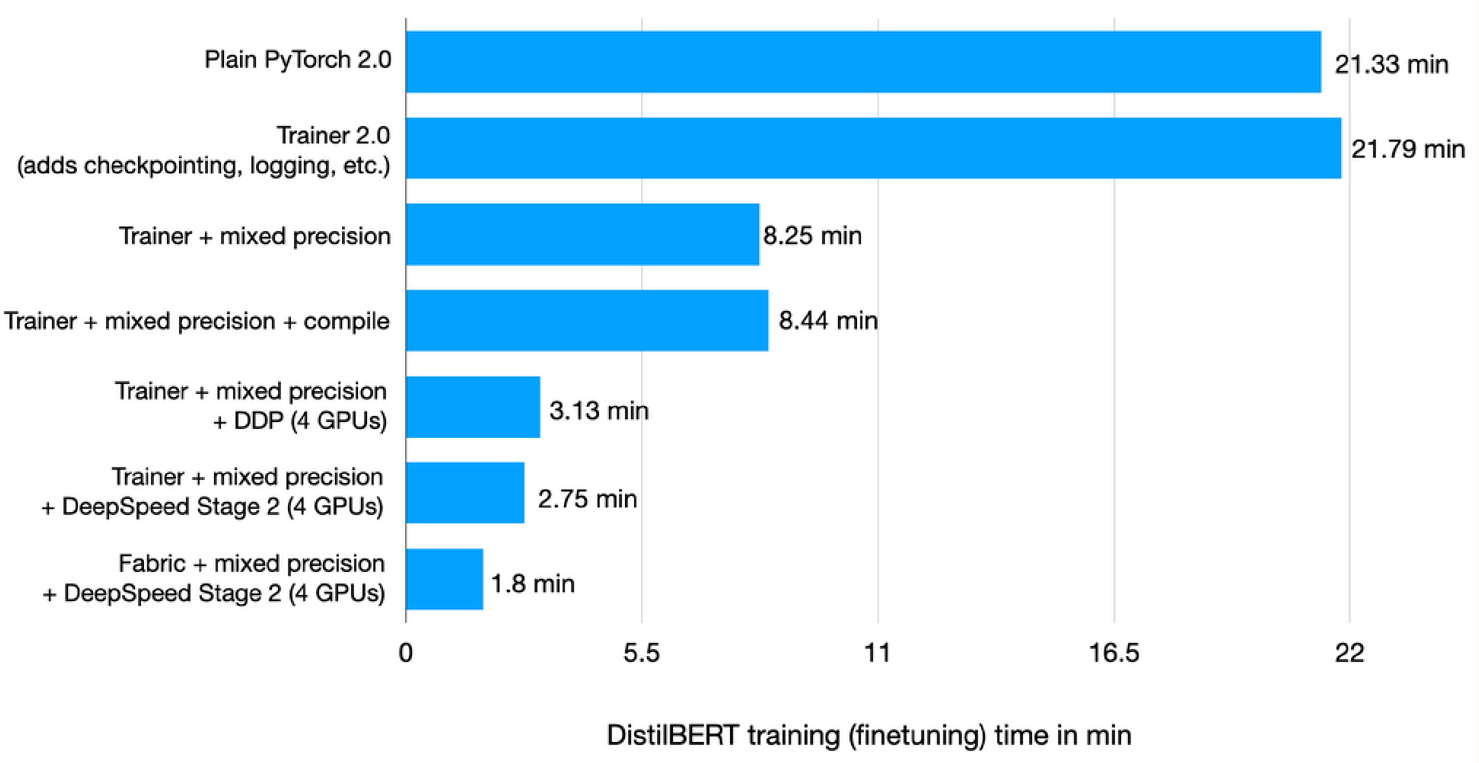
LLM is also broken and parts reside in each GPU

Weights are collected before forward and backward pass from each of the GPUs

Gradients are synchronised

LLM chunks are updated in each of the GPU

EXAMPLE WITH DEEPSPEED + MULTI-GPUS



University of Amsterdam: https://uvadlc-notebooks.readthedocs.io

RESOURCES

- Fully Sharded Data Parallel: faster AI training with fewer GPUs
- Getting Started with Fully Sharded Data Parallel (FSDP)
- Introducing PyTorch Fully Sharded Data Parallel (FSDP) API
- Multi GPU Fine tuning with DDP and FSDP

LET'S GET JAMMIN'

For questions, comments, and feedbacks

Email

mert.bozkirr@gmail.com

Website

www.mertbozkir.com

Linkedin

https://linkedin.com/in/mertbozkir

THANK YOU

for listening