Machine Learning Systems Design

Modeling Pipeline

Lecture 15: High-performance Modeling

CE 40959 Spring 2023
Ali Zarezade
SharifMLSD.github.io

Ways a model can scale

1. In complexity: architecture, number of parameters

Ways a model can scale

1. In complexity: architecture, number of parameters
2. In prediction traffic

Ways a model can scale

1. In complexity: architecture, number of parameters
2. In prediction traffic
3. In number of models

Rise of Incredibly Large DL Models

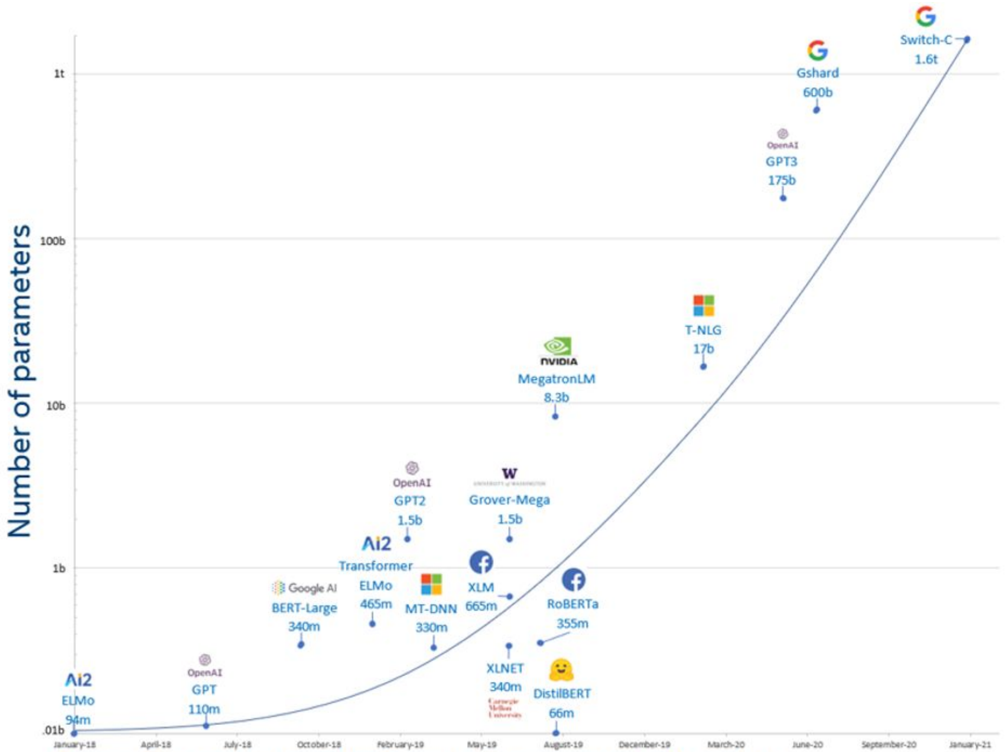
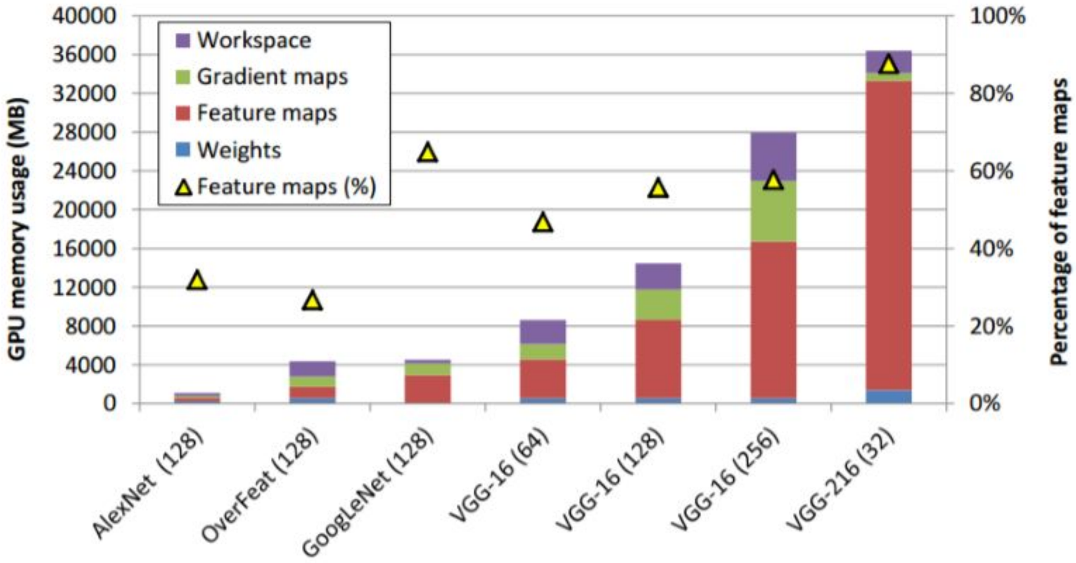


Figure 1: Exponential growth of number of parameters in DL models

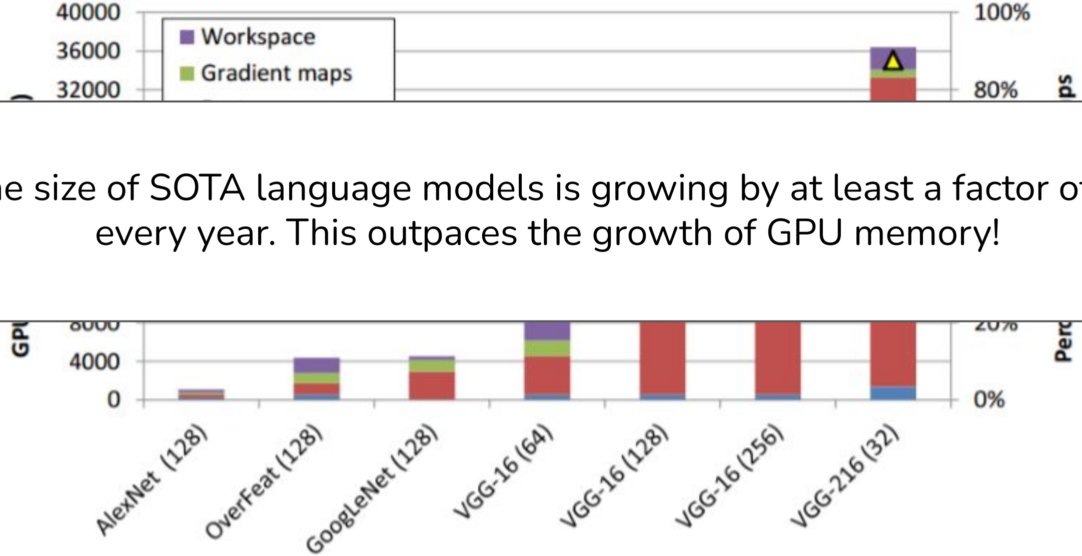
Illustration by [Towards Data Science]

GPU Usage



Rhu, M., et al, [vDNN: Virtualized Deep Neural Networks for Scalable, Memory-Efficient Neural Network Design]()

GPU Usage



The size of SOTA language models is growing by at least a factor of 10 every year. This outpaces the growth of GPU memory!

Rhu, M., et al, [vDNN: Virtualized Deep Neural Networks for Scalable, Memory-Efficient Neural Network Design]

Issues

- A smaller batch size can lead to
  - More iterations necessary to converge
  - Decreased stability

-> What about when the model itself doesn’t fit into GPU memory? Or when even a single data sample doesn’t fit into GPU memory?

1 https://developer.nvidia.com/gtc/2020/video/s21496-vid
2 https://lambdalabs.com/blog/demystifying-gpt-3/

Distributed Training

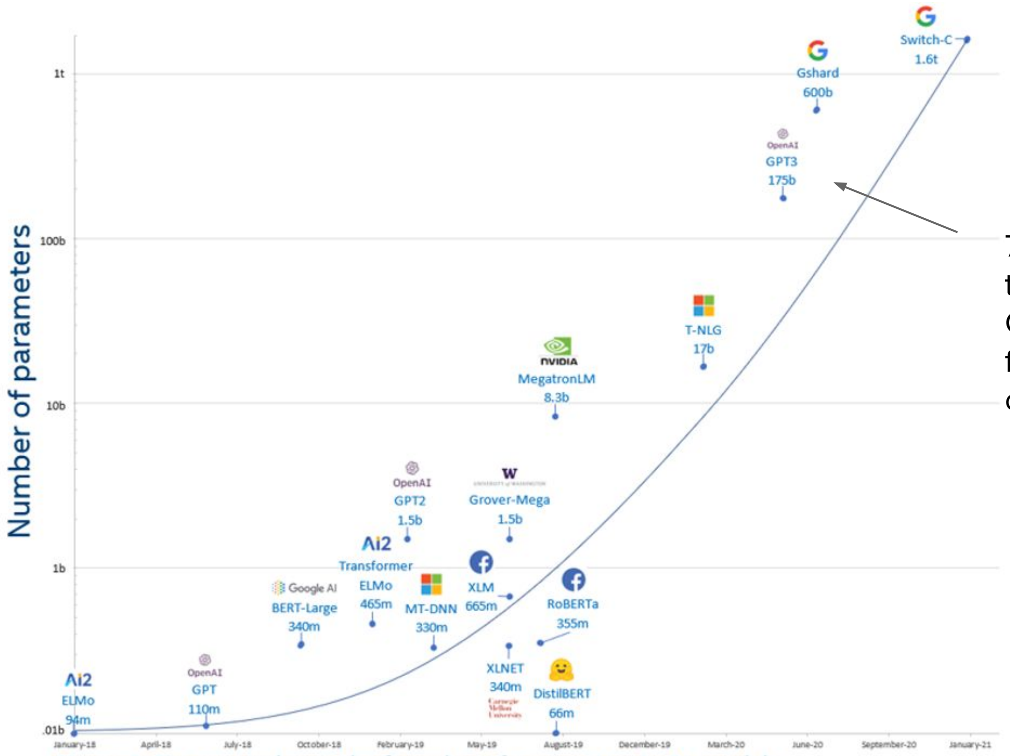


Figure 1: Exponential growth of number of parameters in DL models

700GB memory to store the parameters; 355 GPU-years and $4.6M for a single training run on NVIDIA V100 GPUs[2]

Illustration by Towards Data Science

1 https://developer.nvidia.com/gtc/2020/video/s21496-vid
2 https://lambdalabs.com/blog/demystifying-gpt-3/

Distributed Training

8-way model parallelism, 64-way data parallelism on 512 GPUs[1]

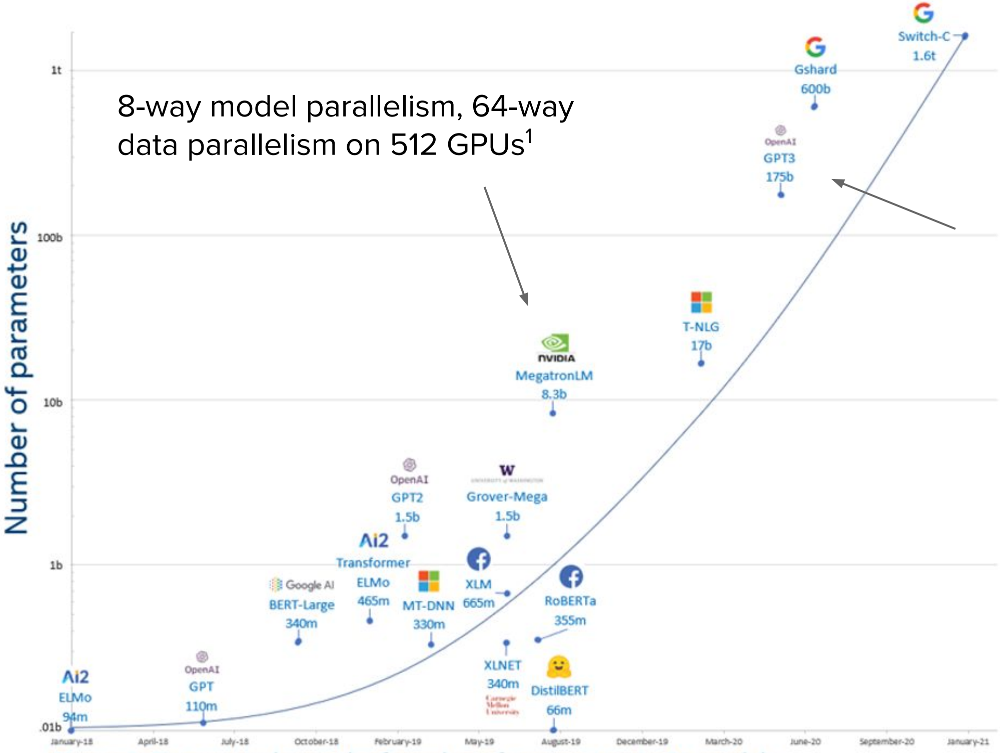


Figure 1: Exponential growth of number of parameters in DL models

700GB memory to store the parameters; 355 GPU-years and $4.6M for a single training run on NVIDIA V100 GPUs[2]

Illustration by [Towards Data Science]()

Distributed Training

Data parallelism

Model parallelism

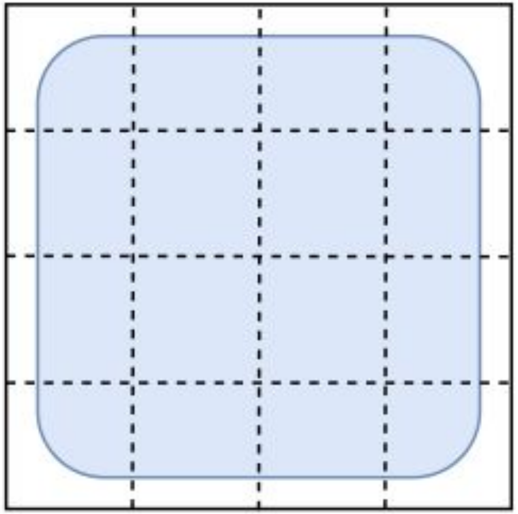
Data Parallelism for Large Batch Training

Split the data across devices

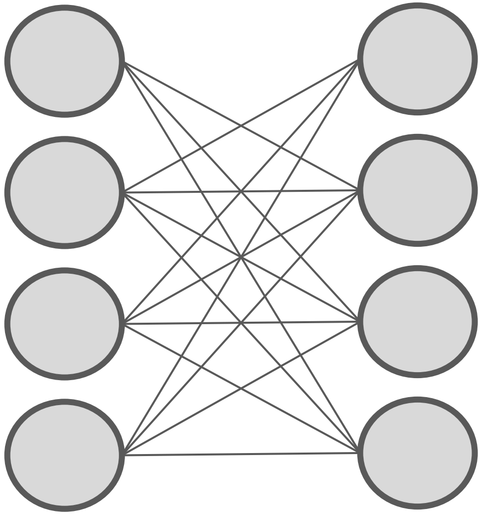
Each device sees a fraction of the batch

Each device replicates the model

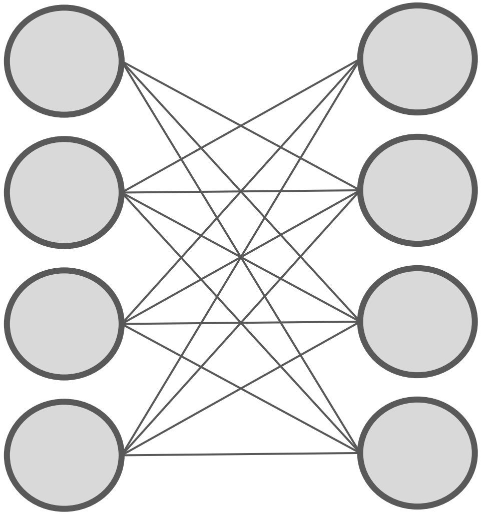
Each device replicates the optimizer



Replicate model across devices



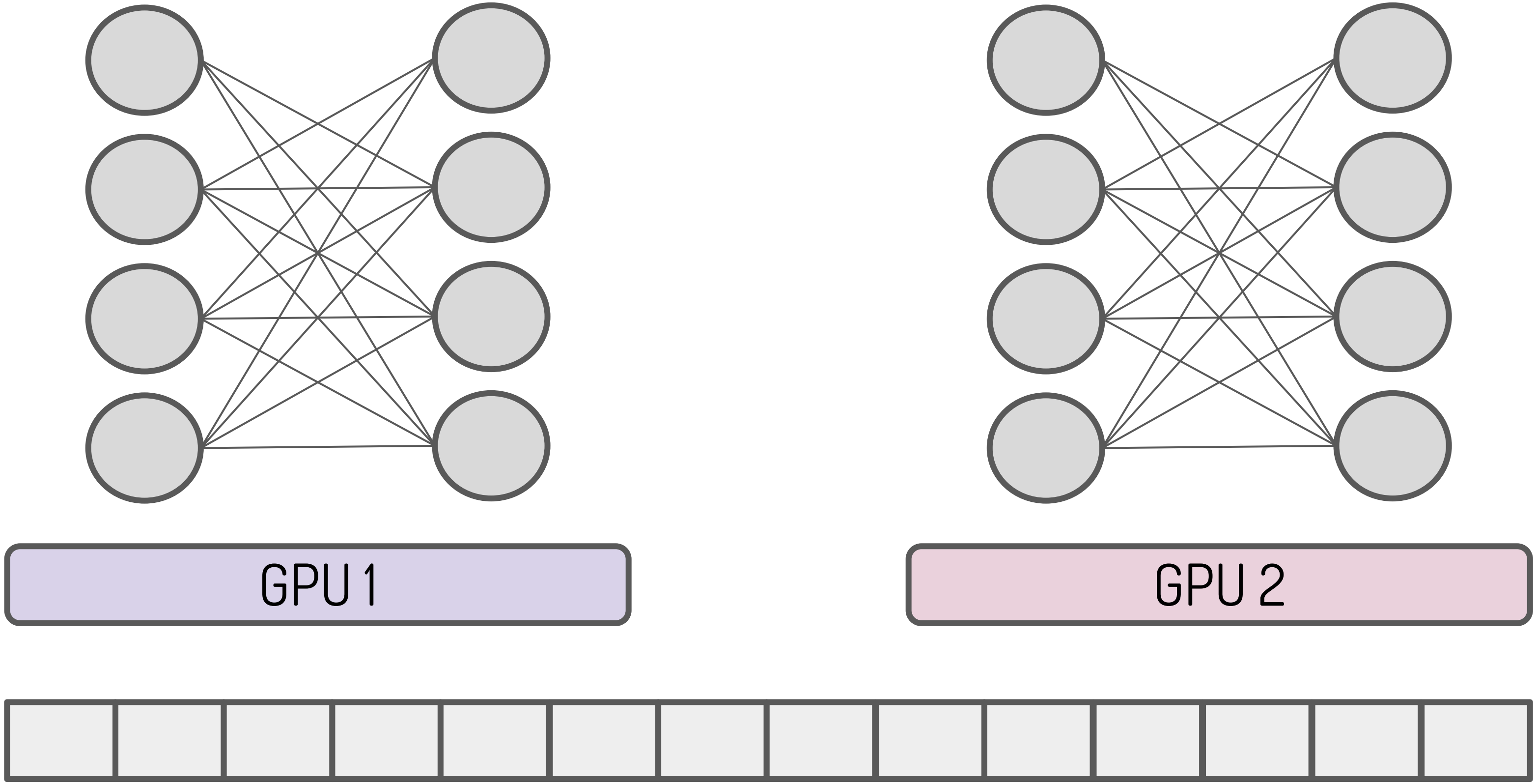
GPU 1



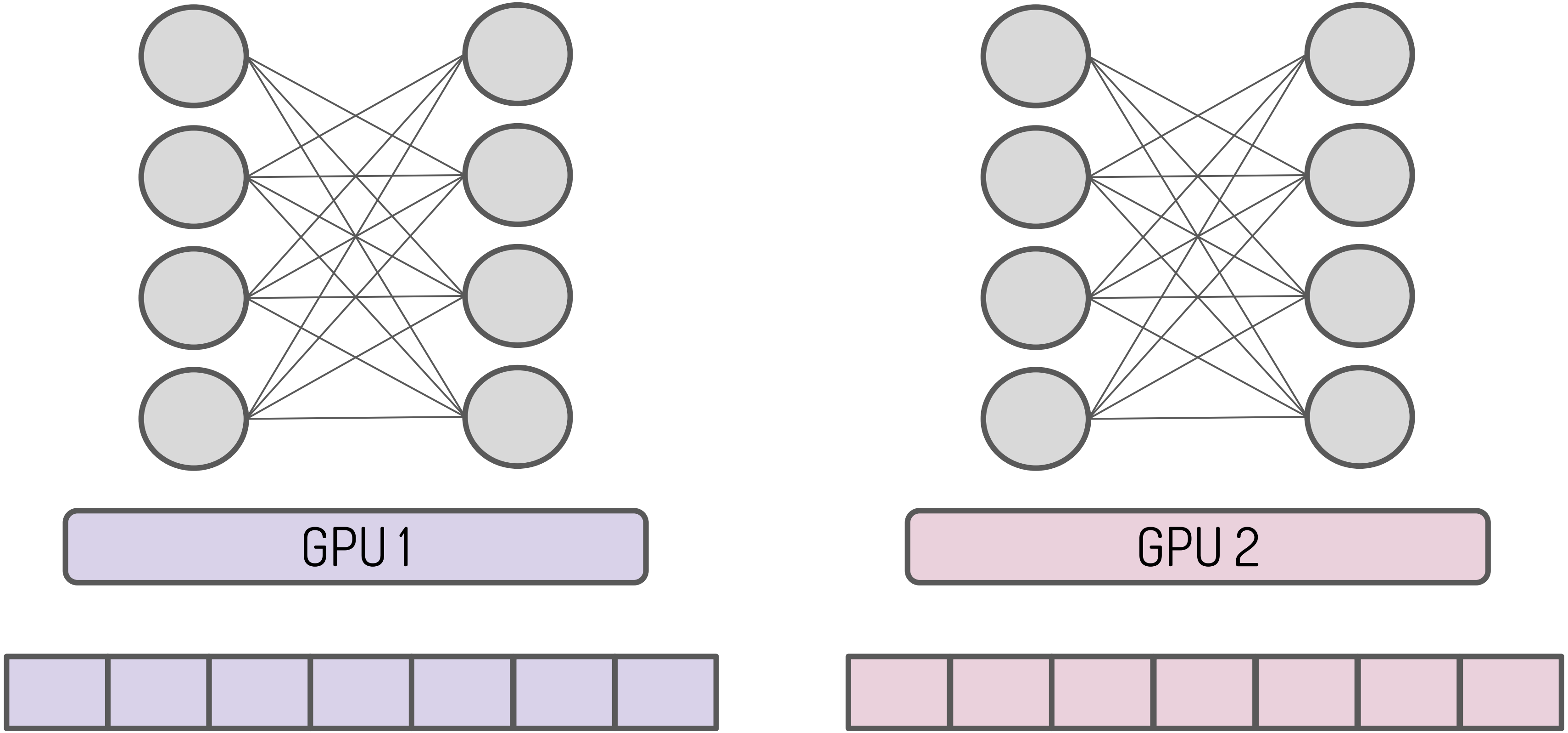
GPU 2

GPUs could be on same or multiple nodes

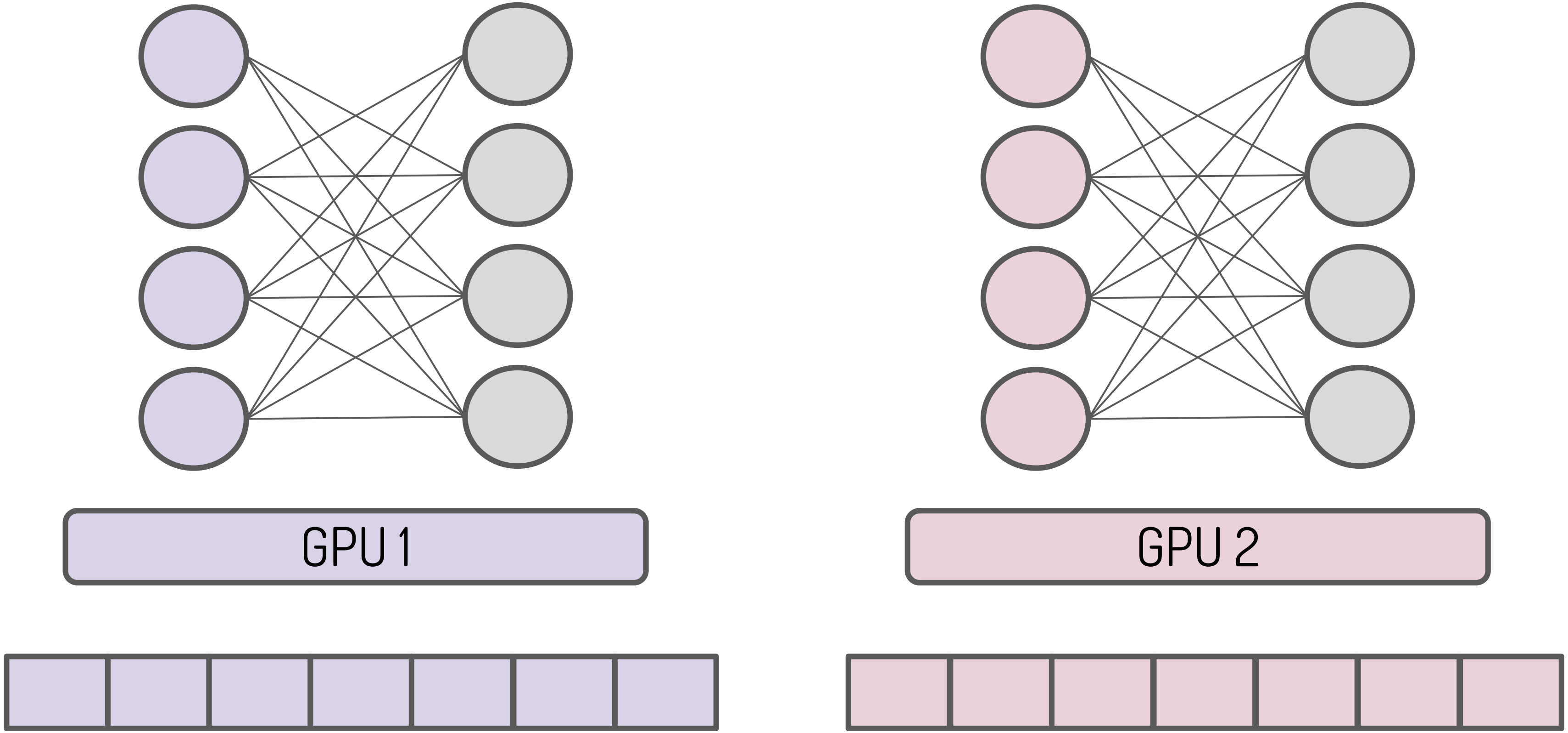
To push in a batch of data



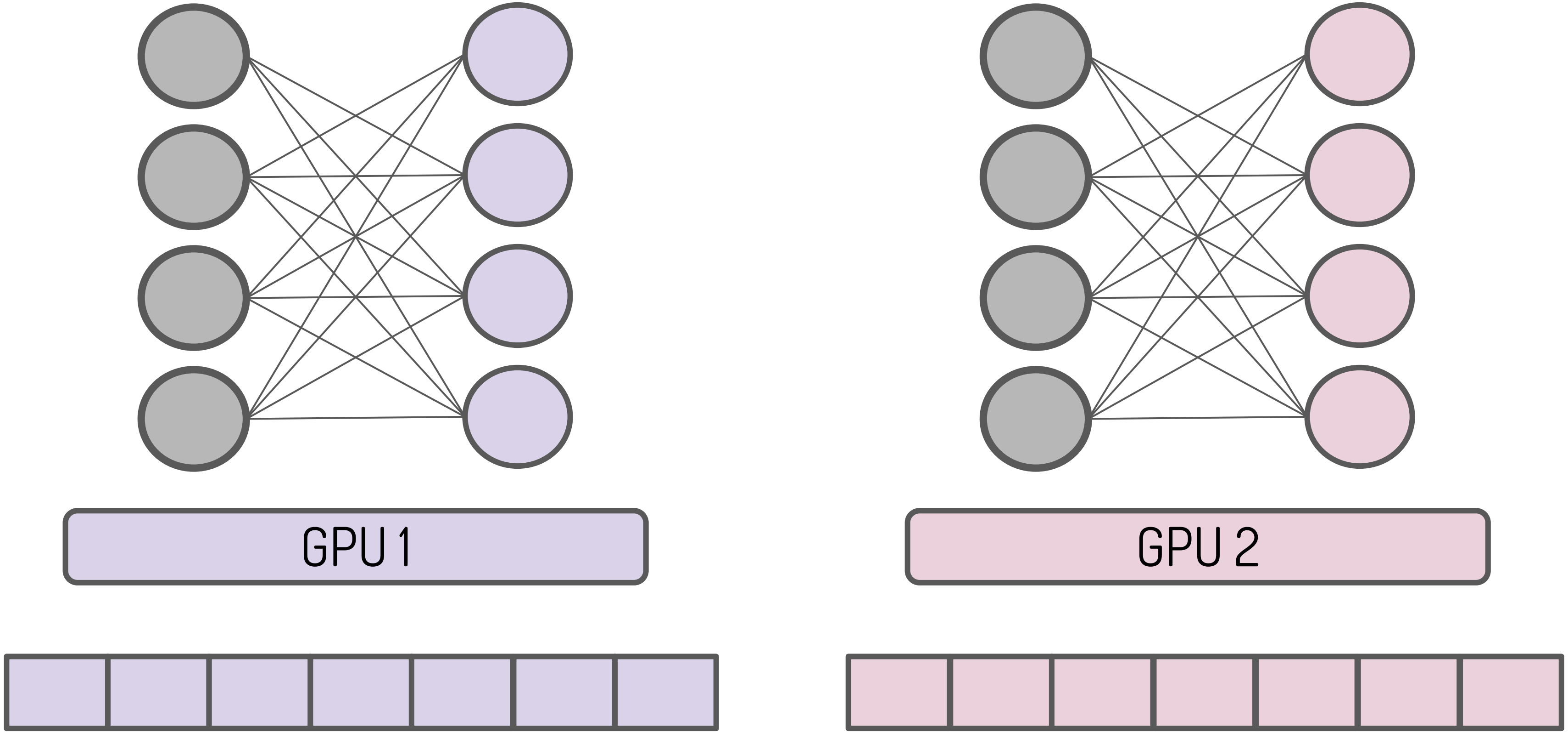
Split batch across devices

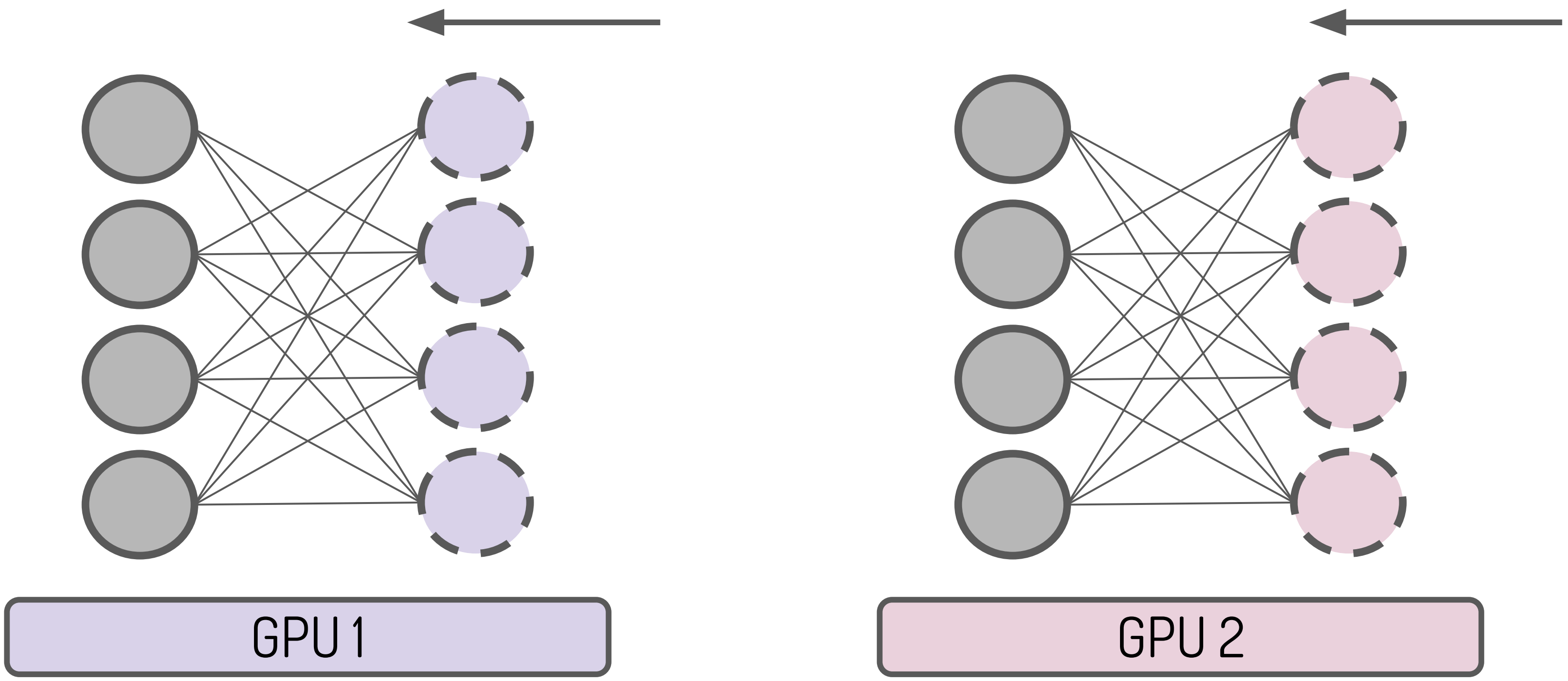


Parallel forward passes

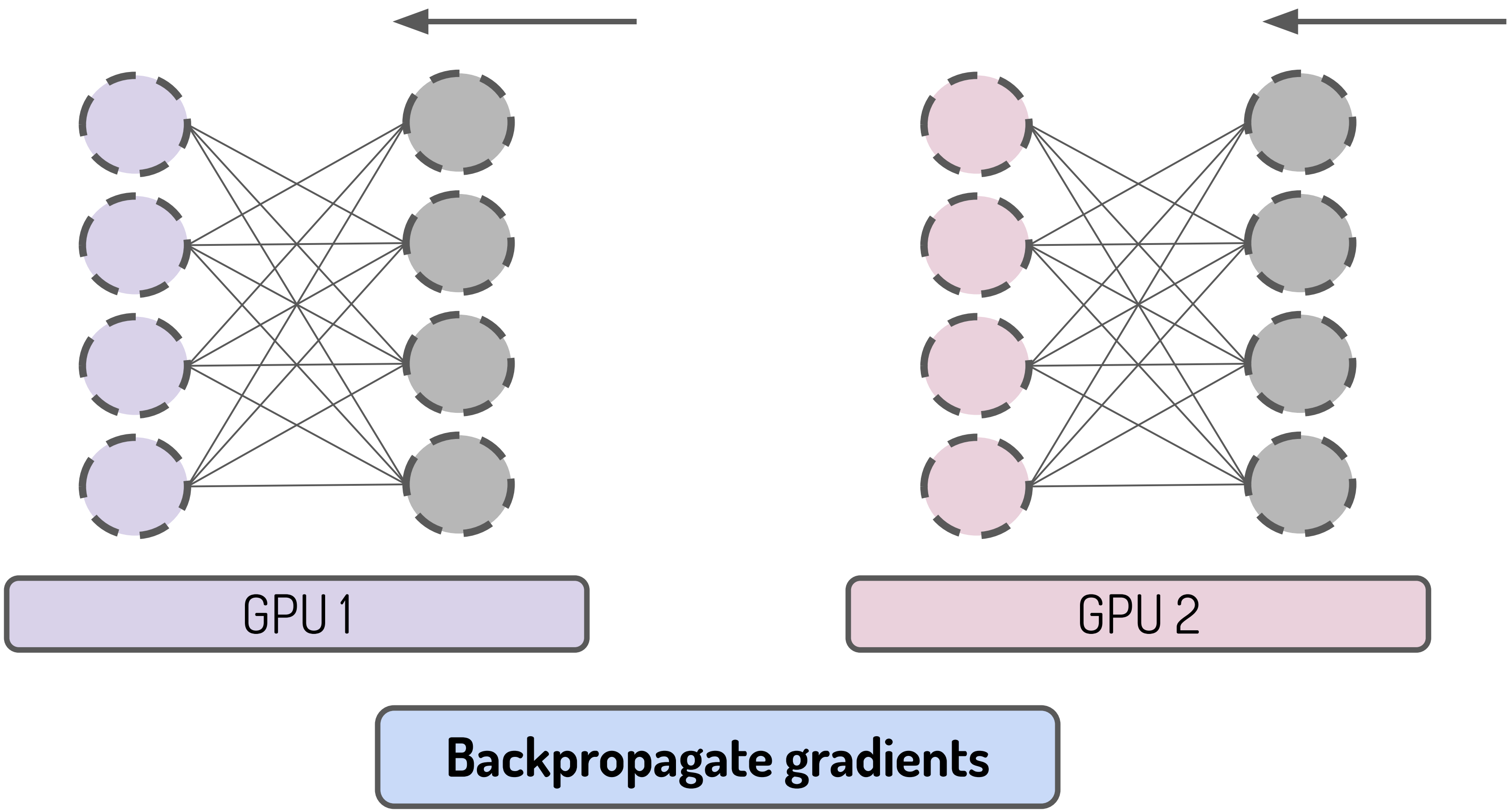


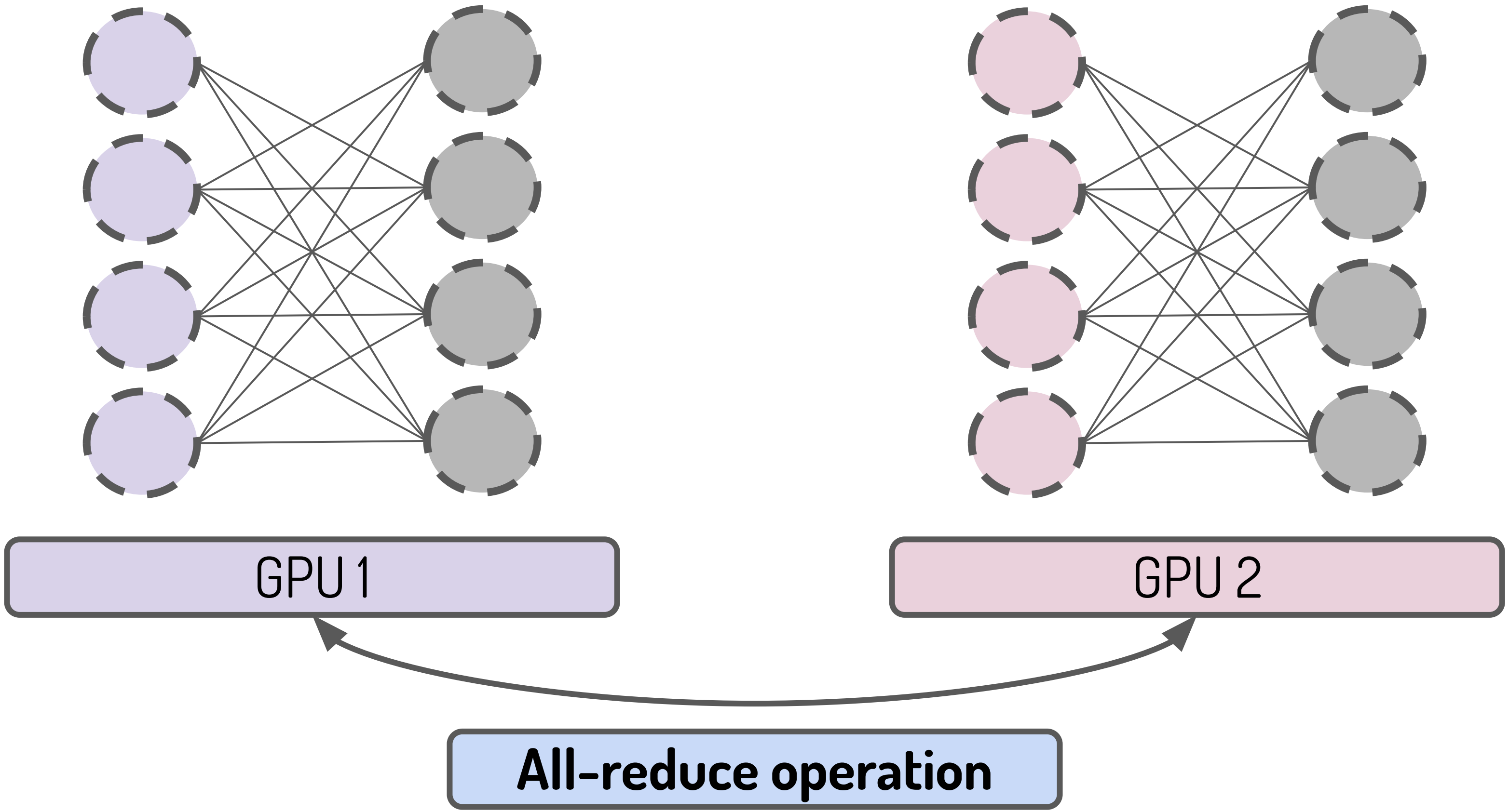
Parallel forward passes



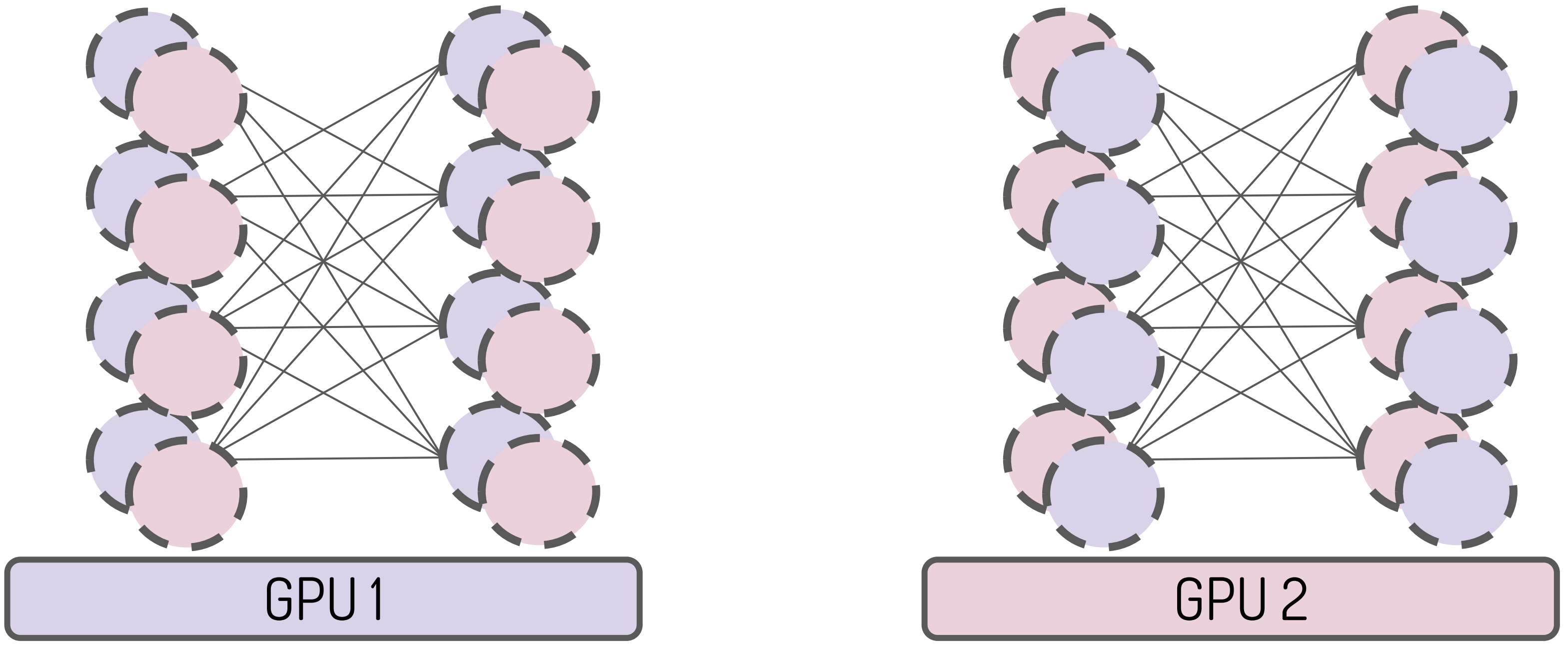


Backpropagate gradients

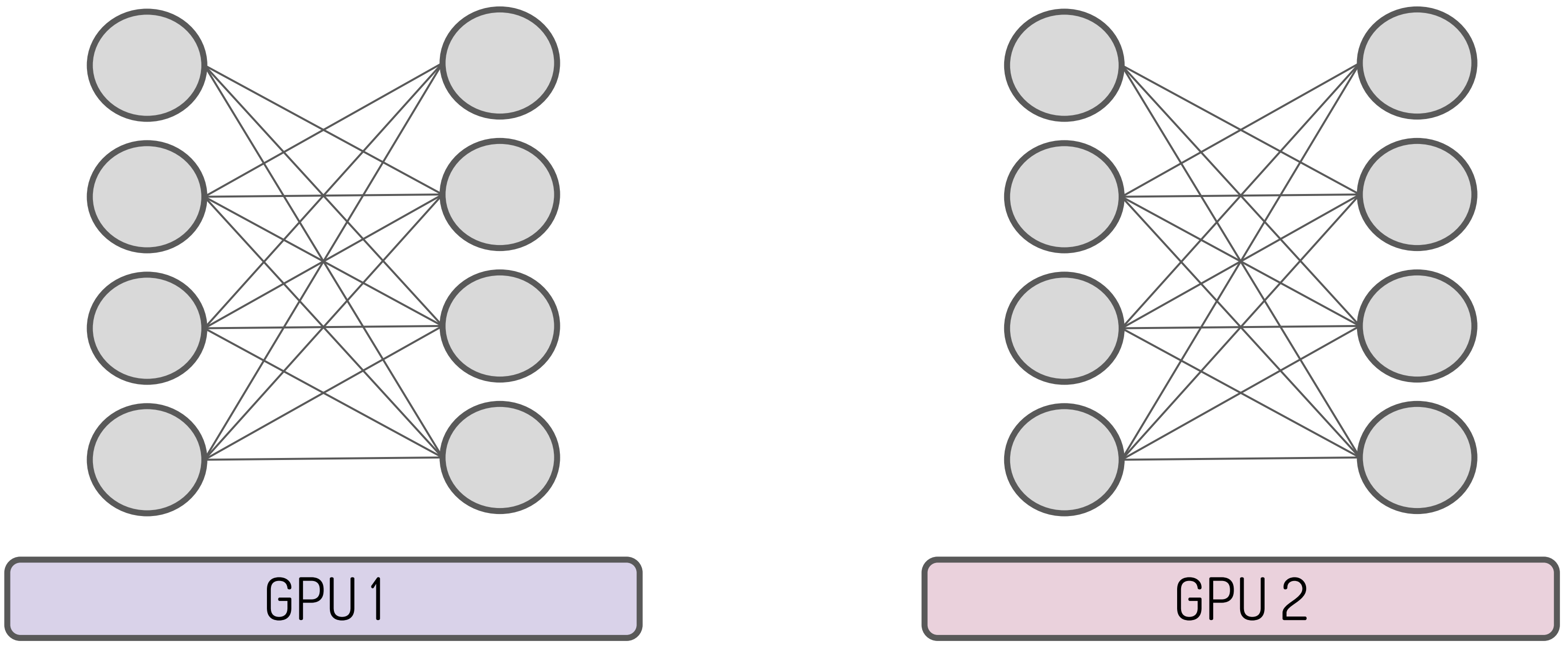

GPU 1
GPU 2
Backpropagate gradients



GPU 1
GPU 2
All-reduce operation



All devices do the same gradient updates



All parameters stay synchronized!

Data Parallelism

Split the data across devices

Each device sees a fraction of the batch

Each device replicates the model

Each device replicates the optimizer

GPT-3: 3.2M batch size

1M samples

- 1000 samples/batch/machine
- 1 machine: 1000 batches
- 100 machines: **10 batches**

[Scaling SGD Batch Size to 32K for ImageNet Training](#) (You et al., 2017)

Data Parallelism

Split the data across devices

Each device sees a fraction of the batch

Each device replicates the model

Each device replicates the optimizer

GPT-3: 3.2M batch size

Challenge 1: Learning rate

- Too small -> too long to converge
- Too large -> unstable learning

Scaling SGD Batch Size to 32K for ImageNet Training (You et al., 2017)

Data Parallelism: LR Scaling

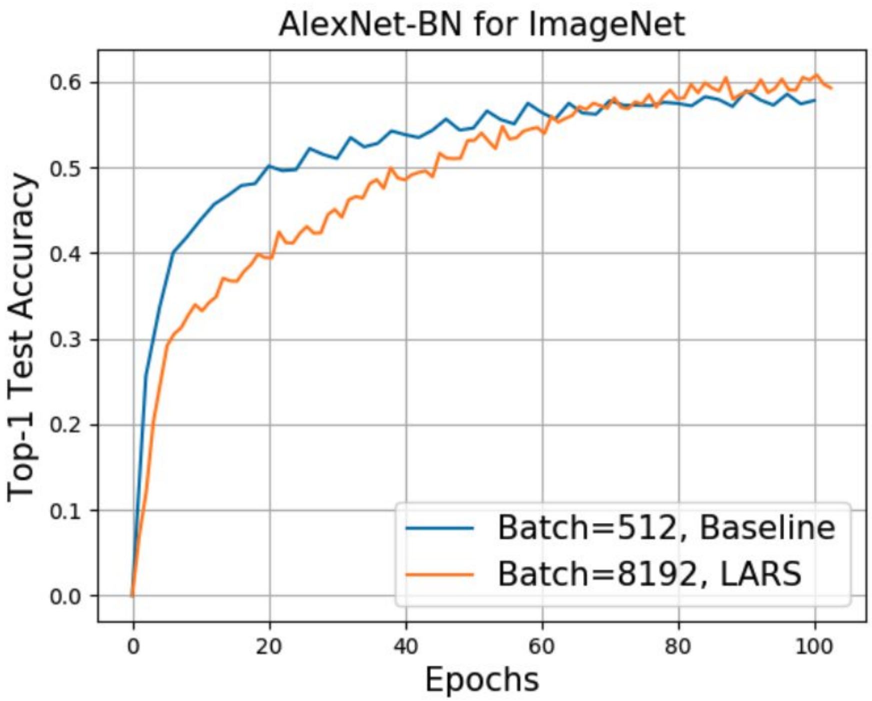
Split the data across devices

Each device sees a fraction of the batch

Each device replicates the model

Each device replicates the optimizer

GPT-3: 3.2M batch size

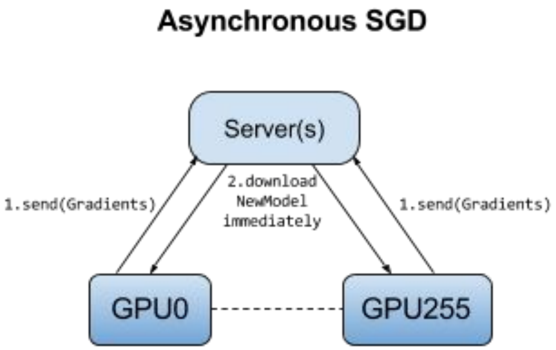


Scaling SGD Batch Size to 32K for ImageNet Training (You et al., 2017)

Data Parallelism: Gradient Updates

Challenge 2: How to aggregate gradient updates?

- Synchronous: have to wait for stragglers
- Asynch: gradients become stale



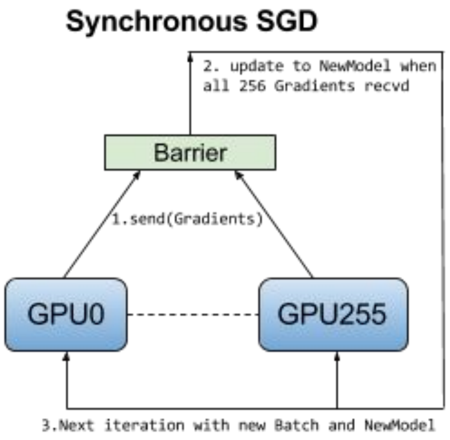
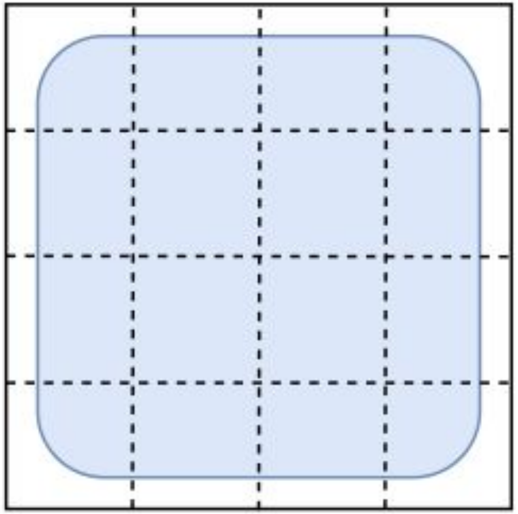


Image from Distributed TensorFlow (Jim Dowling, O'Reilly 2017)

Solution: Model Parallelism for Large Model Training

Split the model across devices

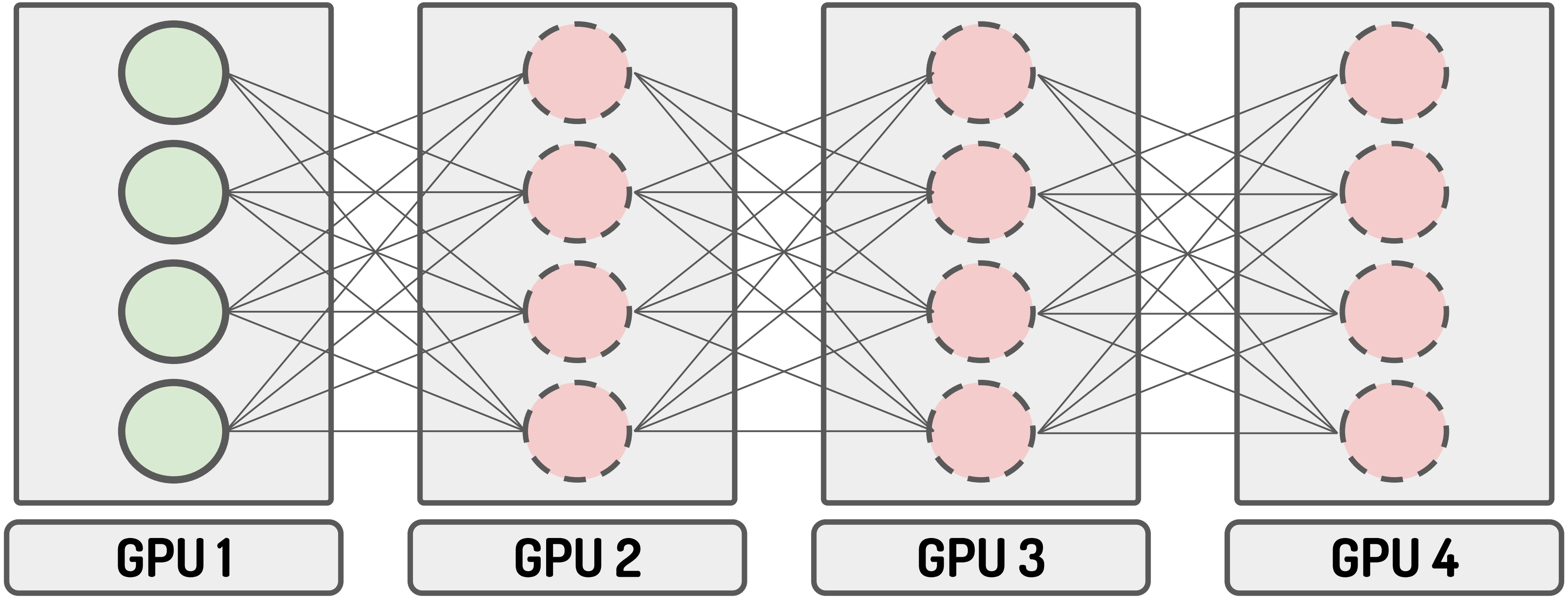
Each device runs a fragment of the model



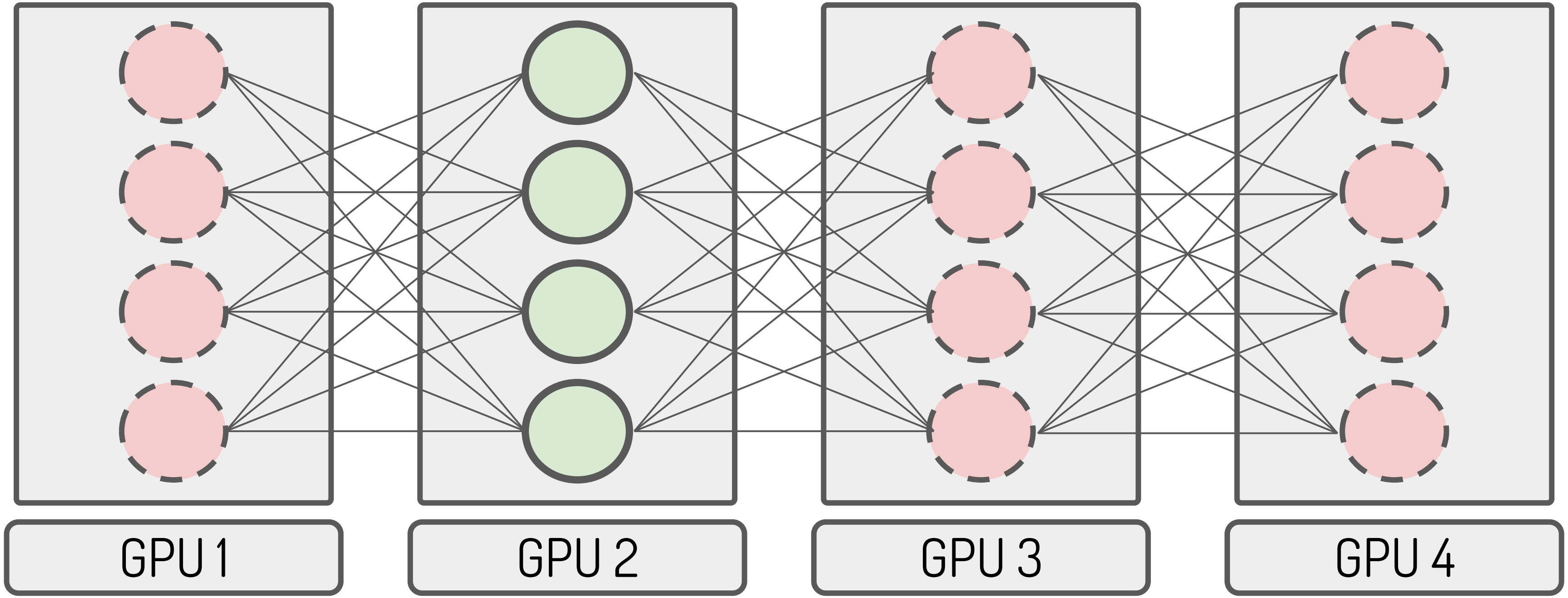
Credit: Fedus et al. (Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity

Model Parallelism: Naive

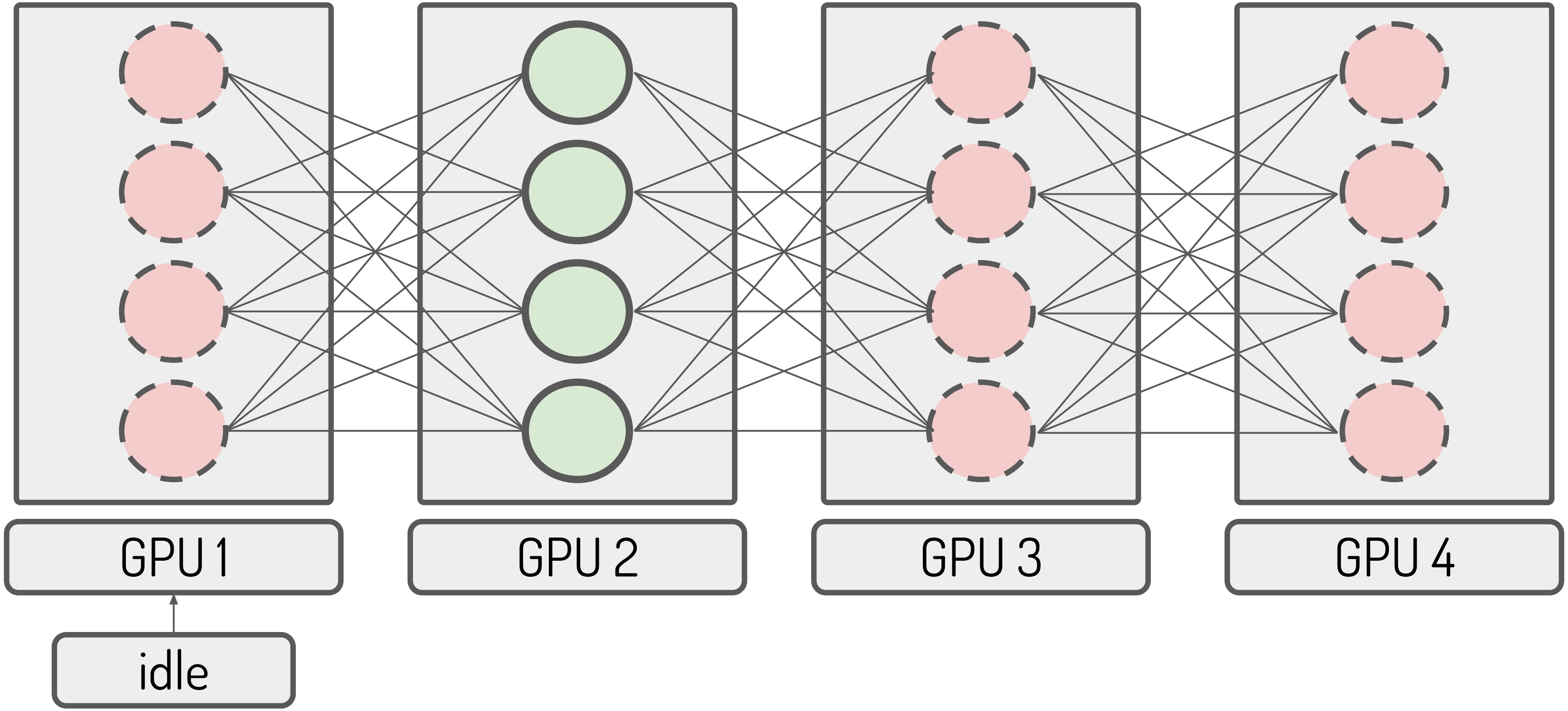
Model Parallelism: Naive



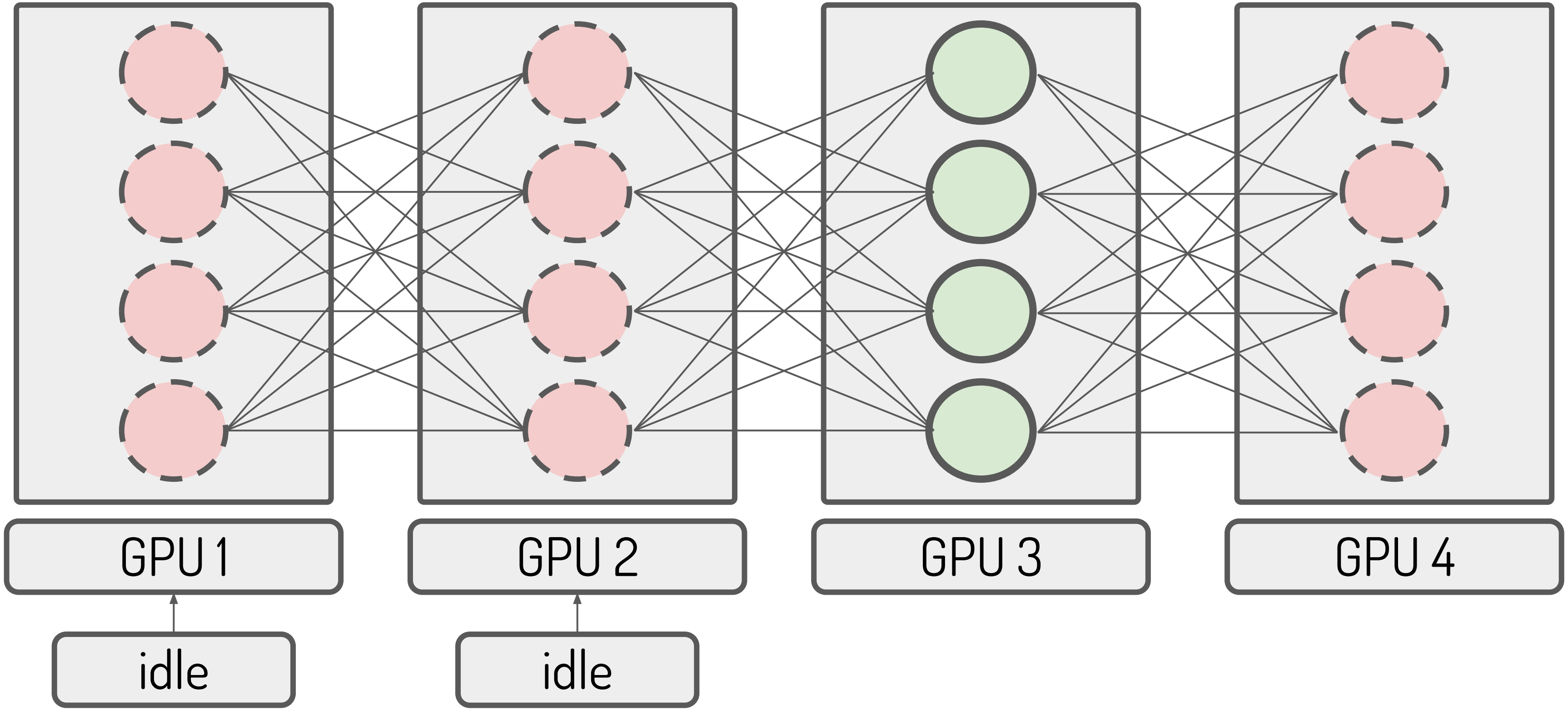
Model Parallelism: Naive



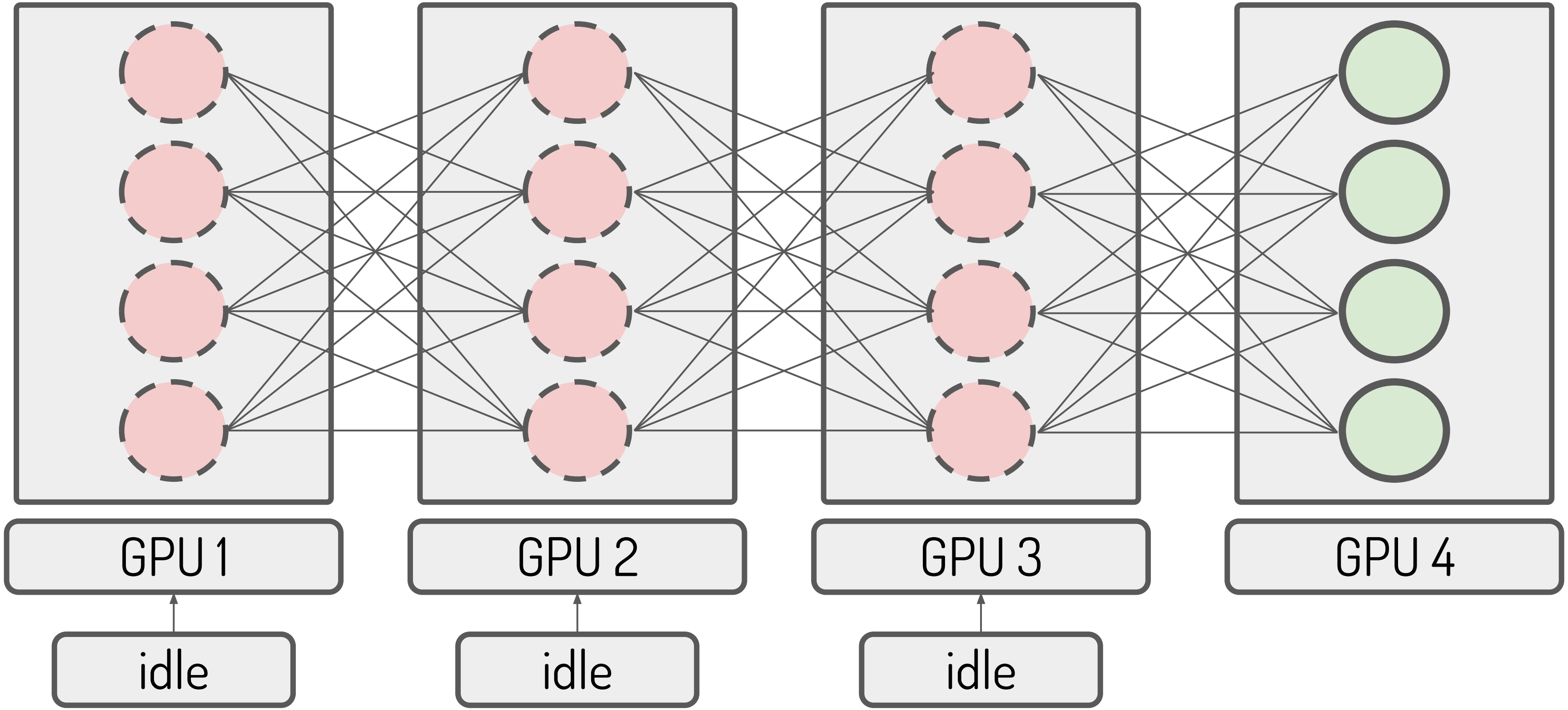
Model Parallelism: Naive



Model Parallelism: Naive

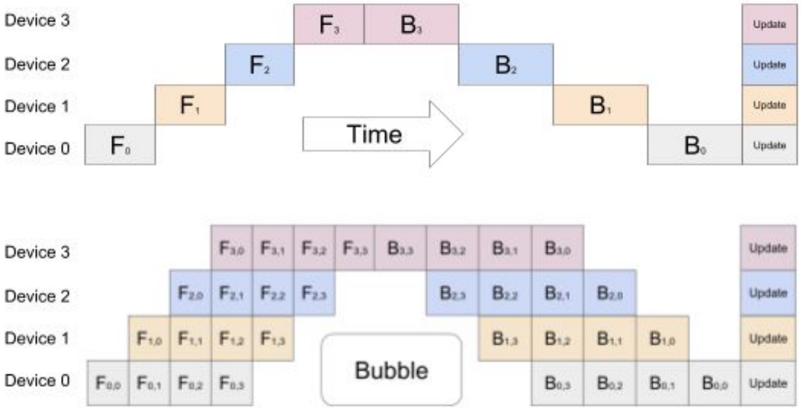


Model Parallelism: Naive



Pipeline Parallelism

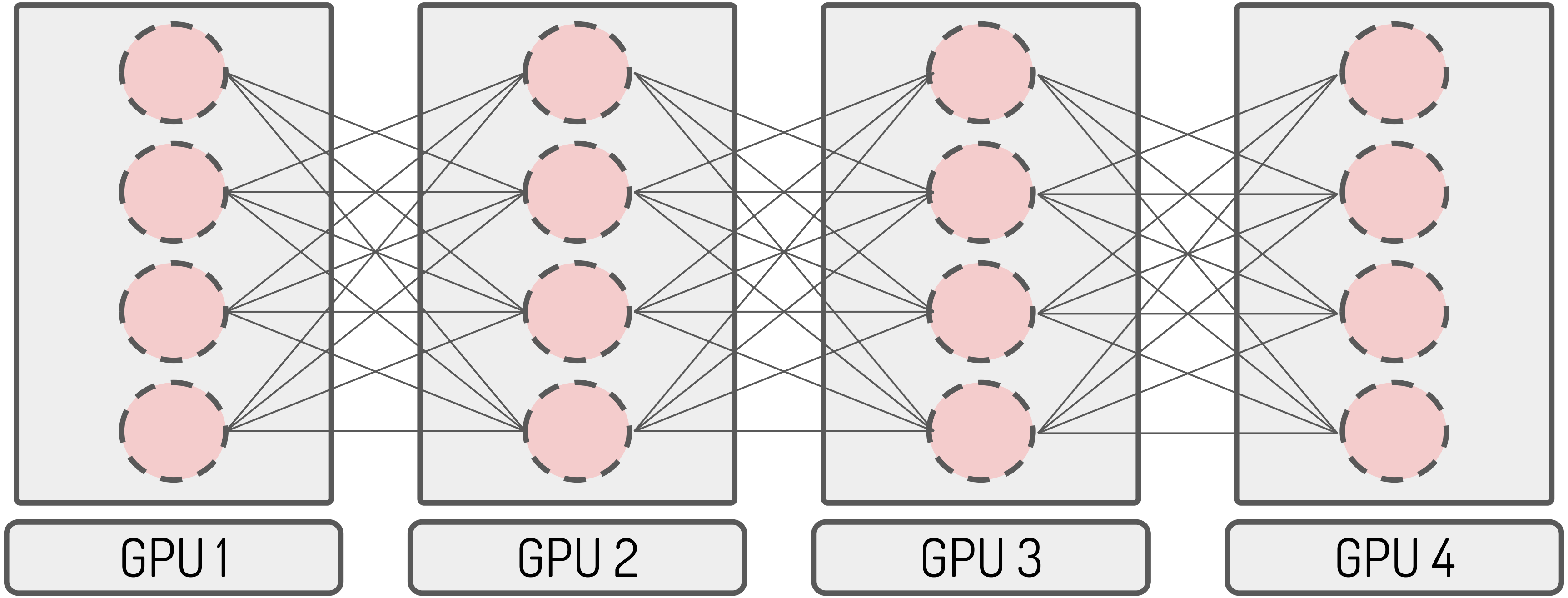
Pipeline Parallelism



Top: *The naive model parallelism strategy leads to severe underutilization due to the sequential nature of the network. Only one accelerator is active at a time.* ***Bottom:*** *GPipe divides the input mini-batch into smaller micro-batches, enabling different accelerators to work on separate micro-batches at the same time.*

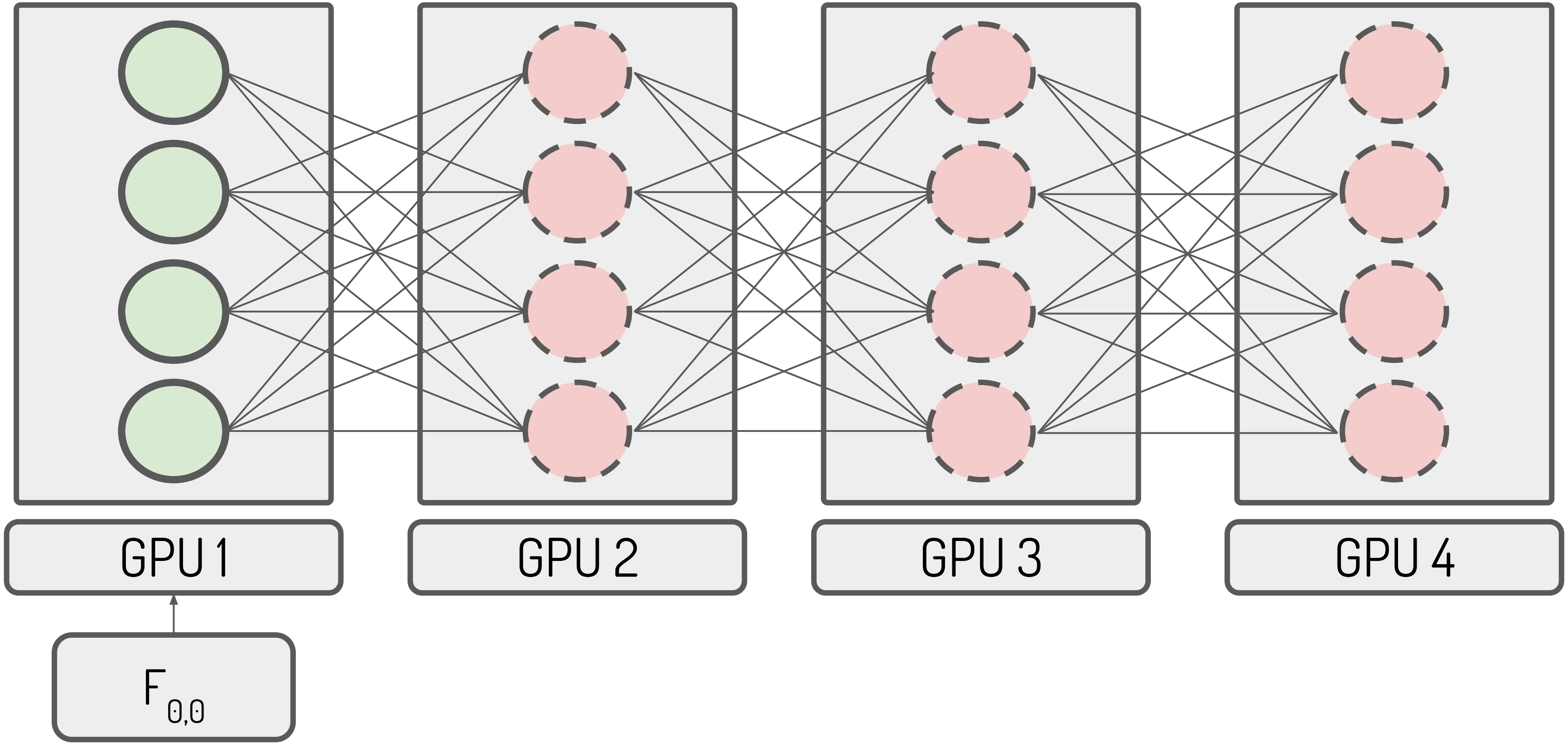
Illustration by [Google AI Blog (GPipe)]

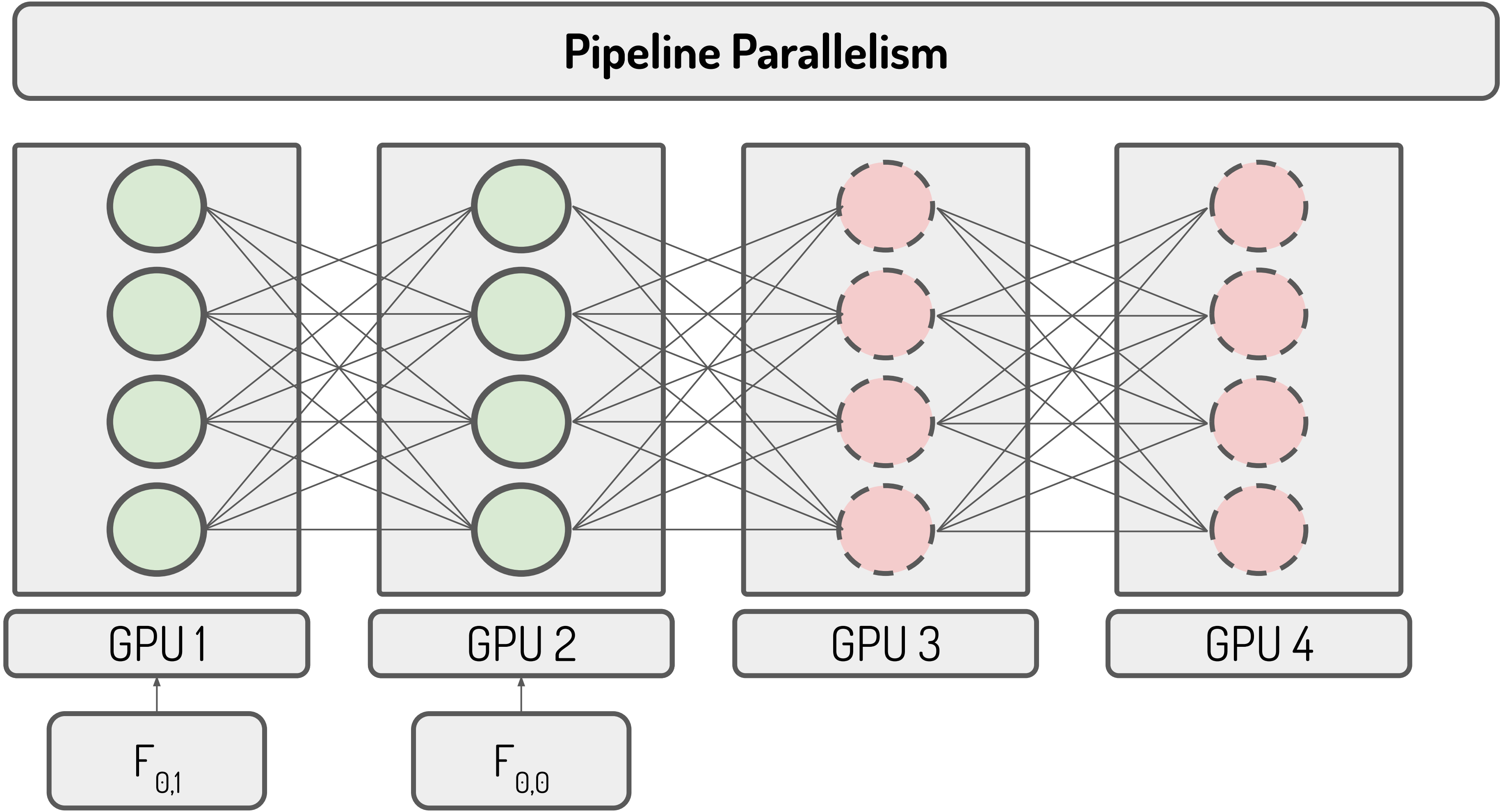
Pipeline Parallelism



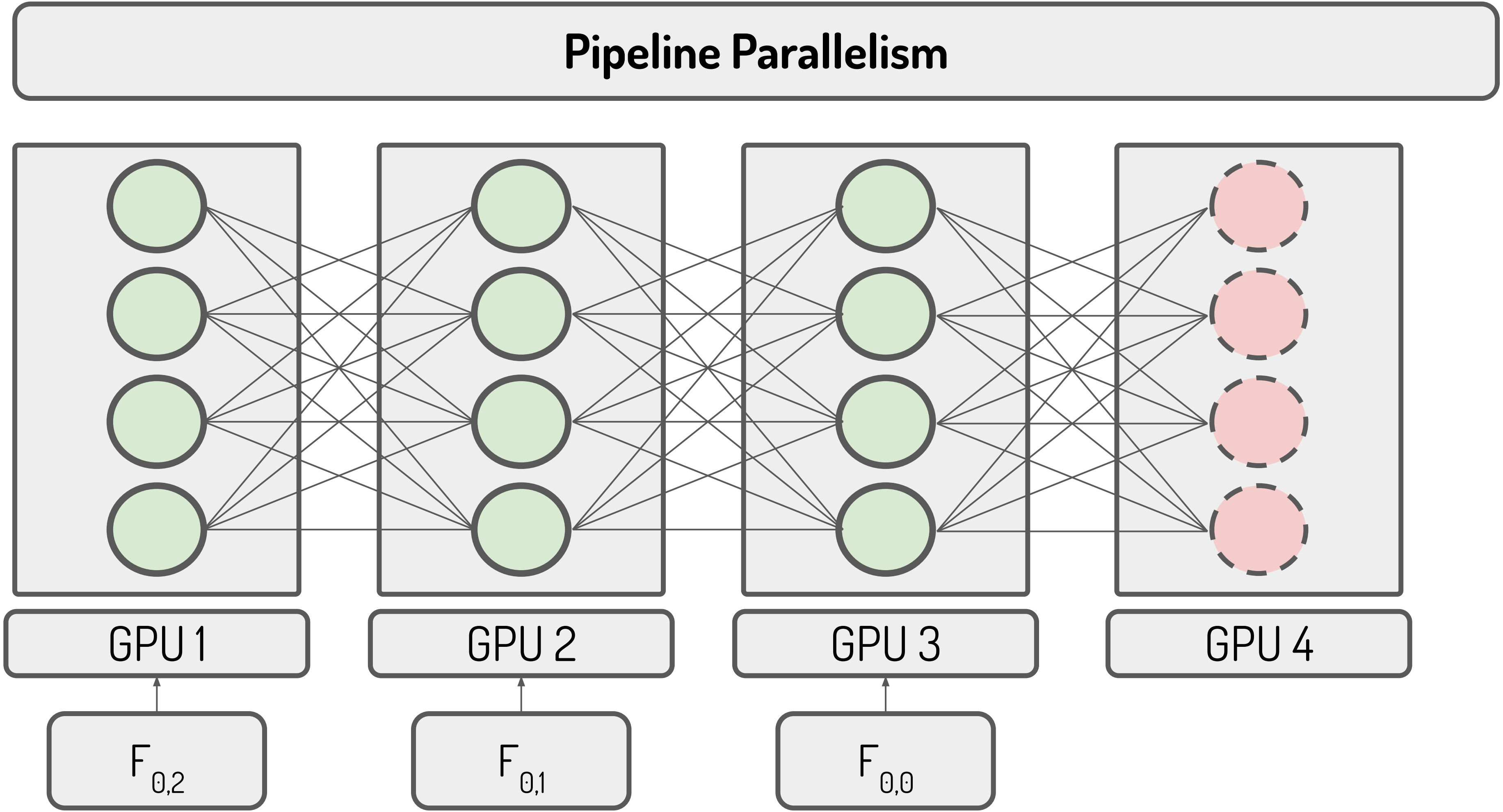
Split mini-batch into sequential micro-batches

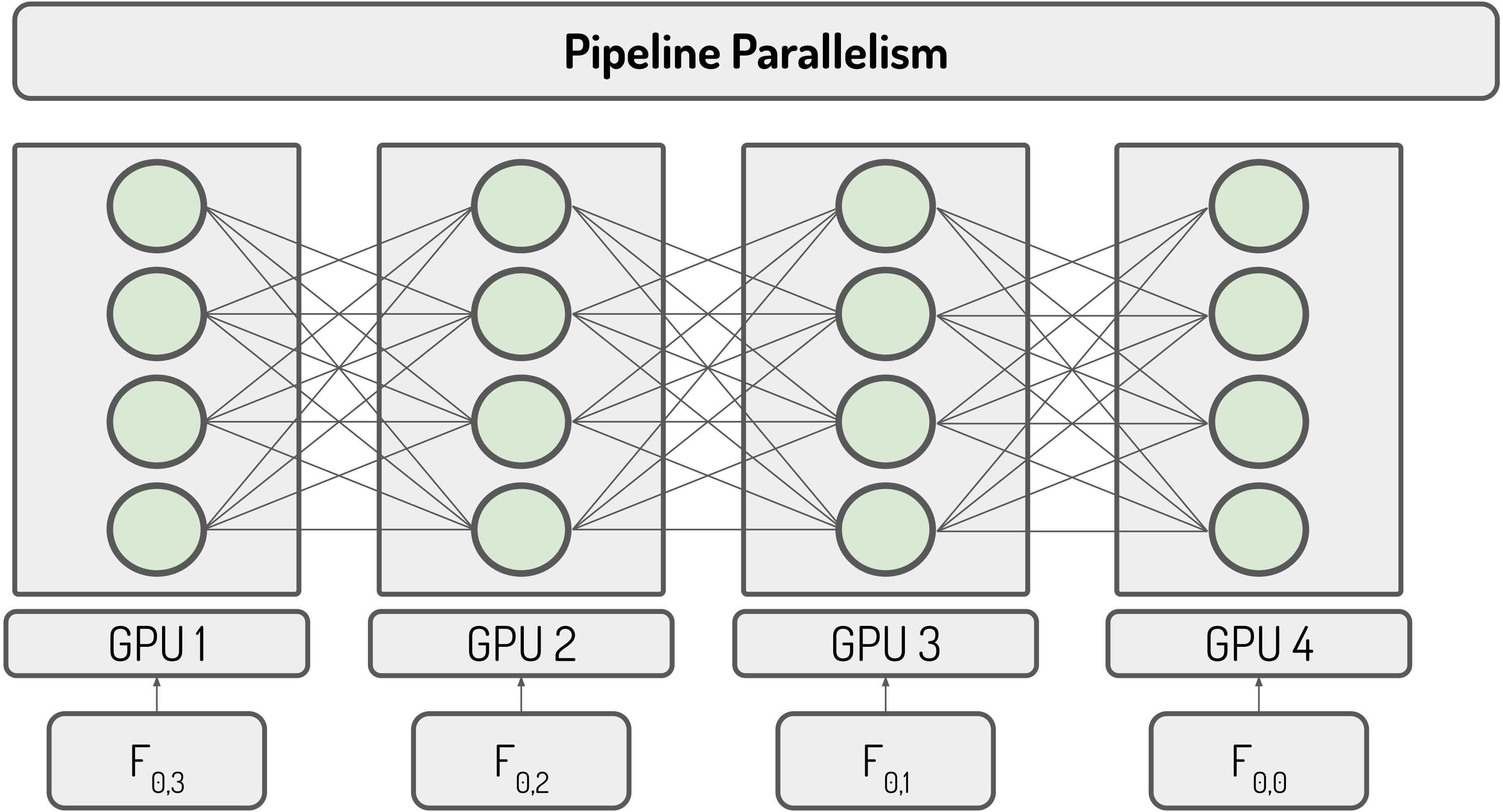
Pipeline Parallelism

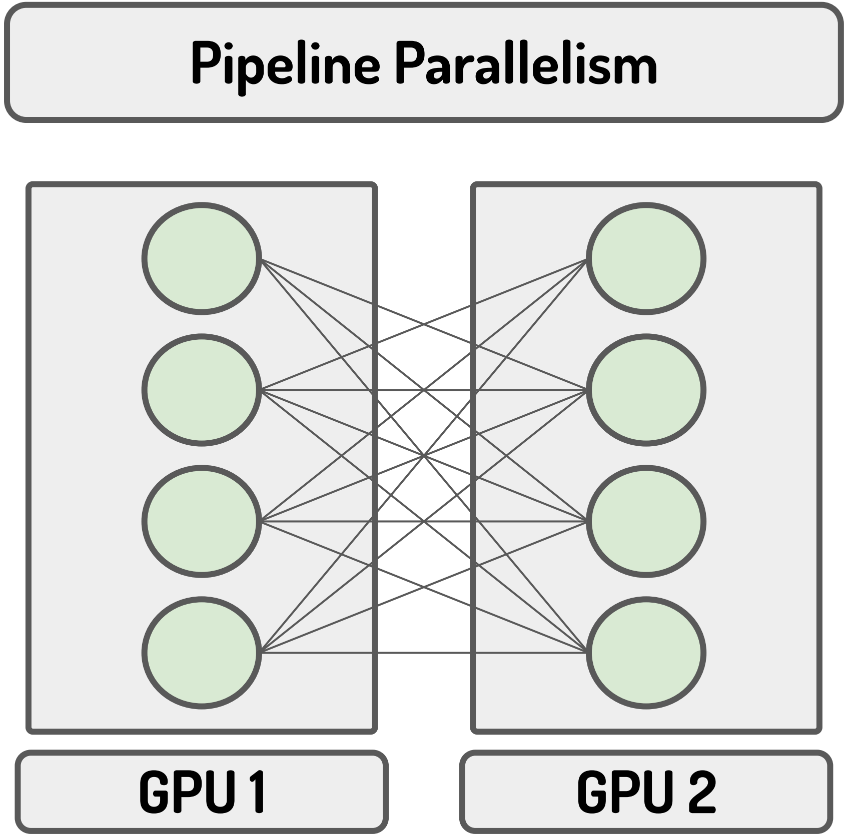


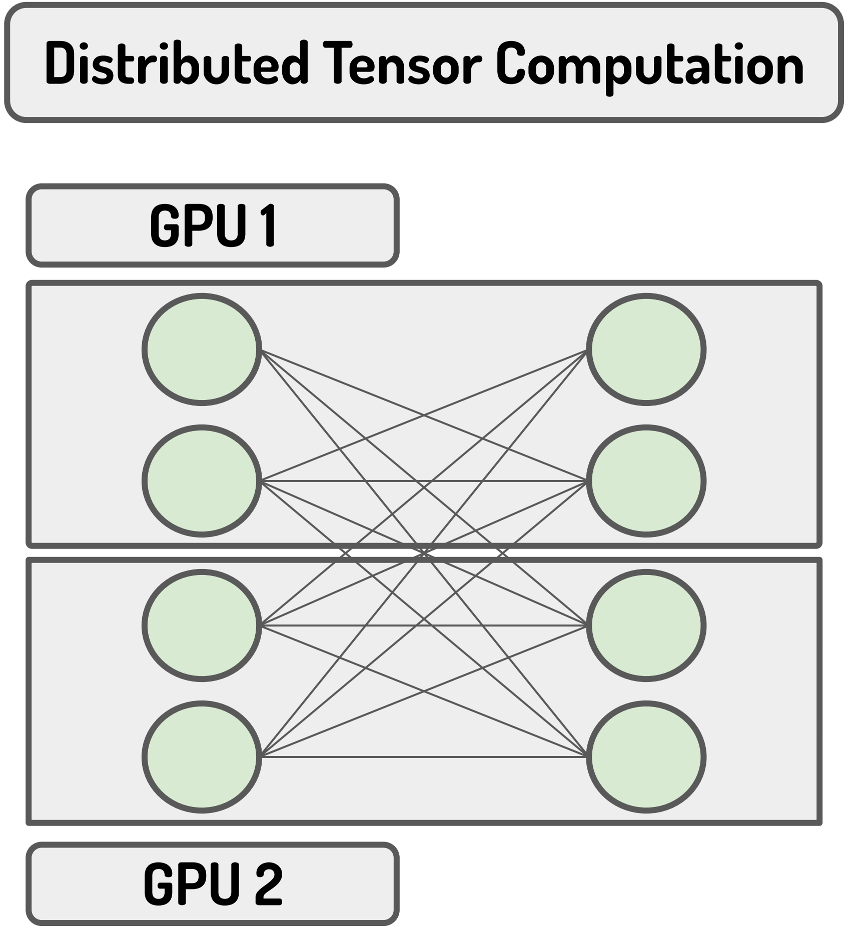


Pipeline Parallelism
GPU 1
GPU 2
GPU 3
GPU 4
$F_{0,1}$
$F_{0,0}$


Pipeline Parallelism
GPU 1
GPU 2
GPU 3
GPU 4
$F_{0,2}$
$F_{0,1}$
$F_{0,0}$


Pipeline Parallelism
GPU 1
GPU 2
GPU 3
GPU 4
$F_{0,3}$
$F_{0,2}$
$F_{0,1}$
$F_{0,0}$


Pipeline Parallelism
GPU 1
GPU 2


Distributed Tensor Computation
GPU 1
GPU 2

Combining Ideas!

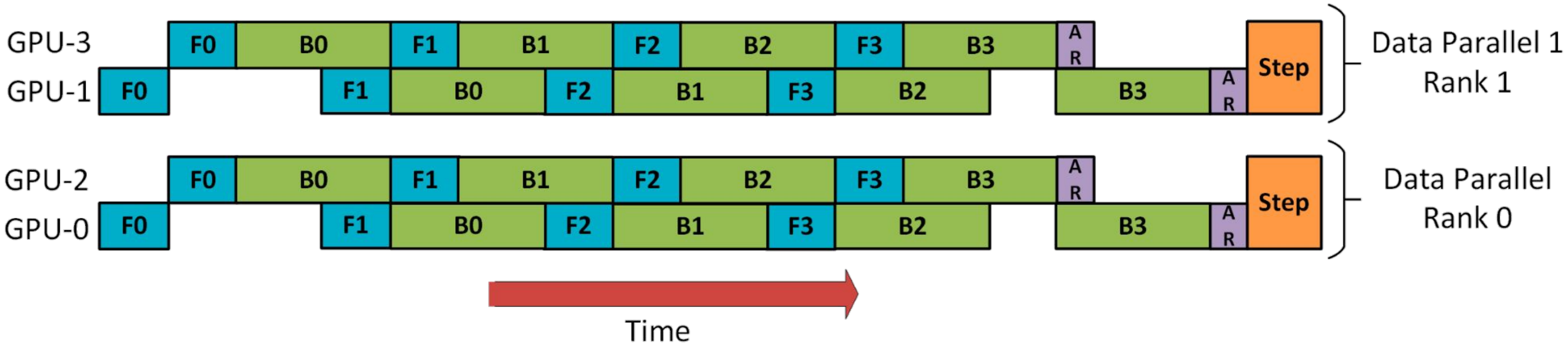
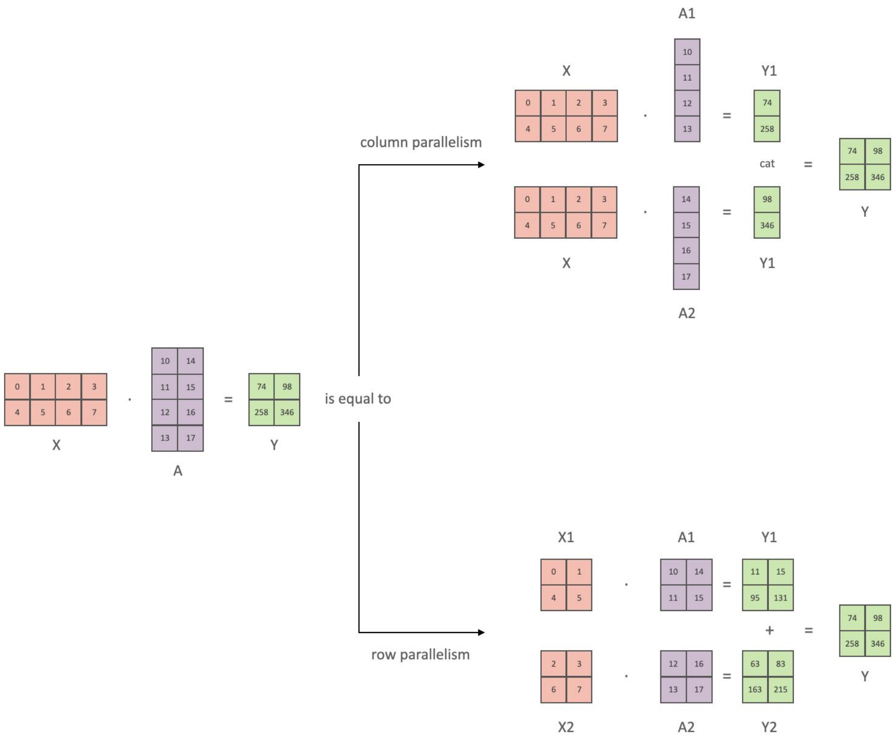
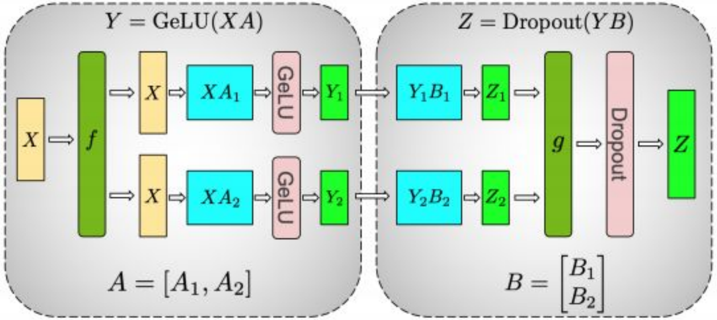


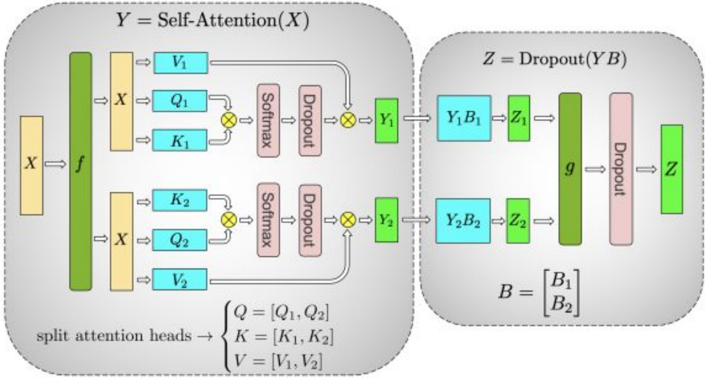
Illustration by DeepSpeed

Tensor Parallelism



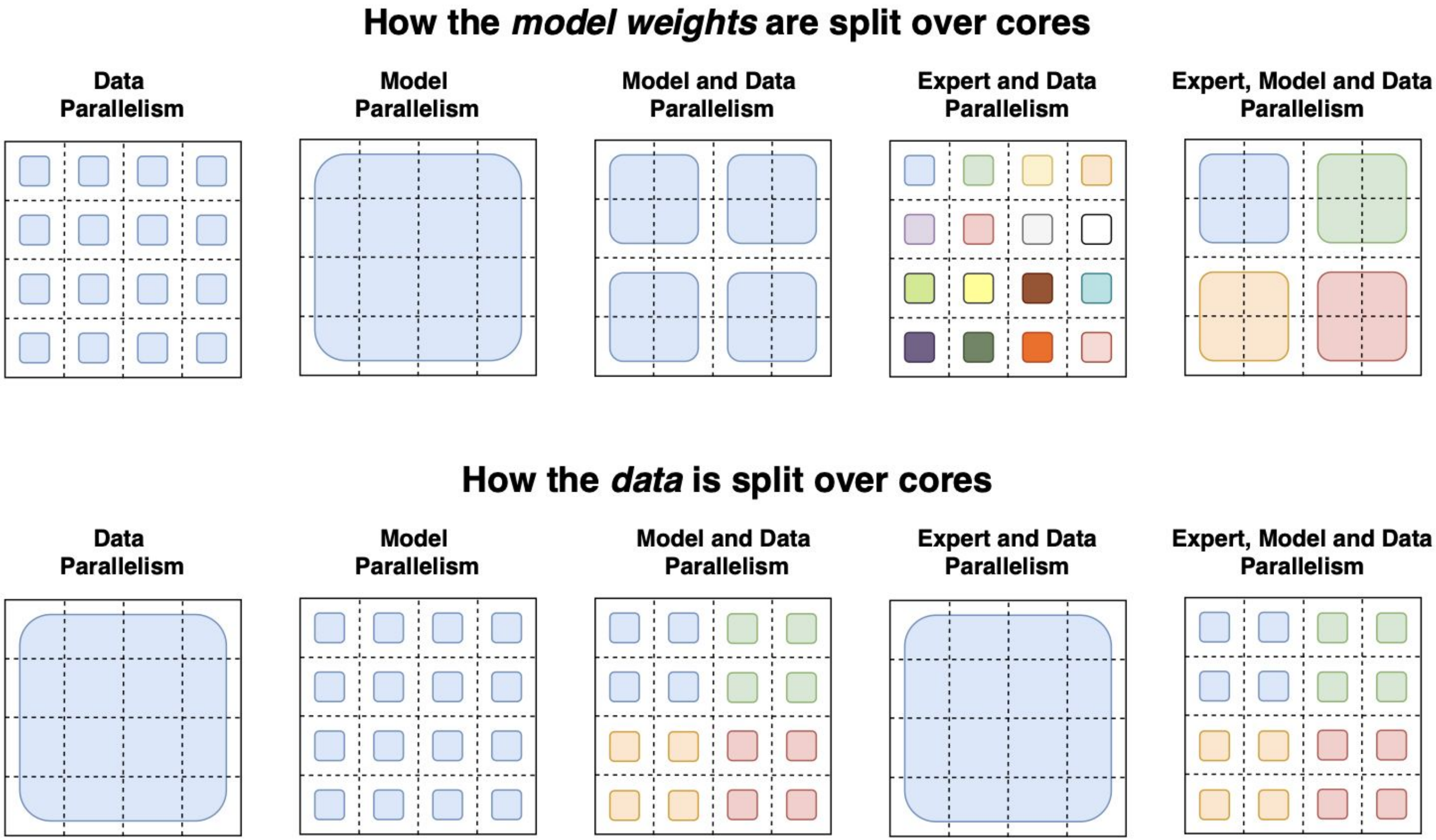


(a) MLP



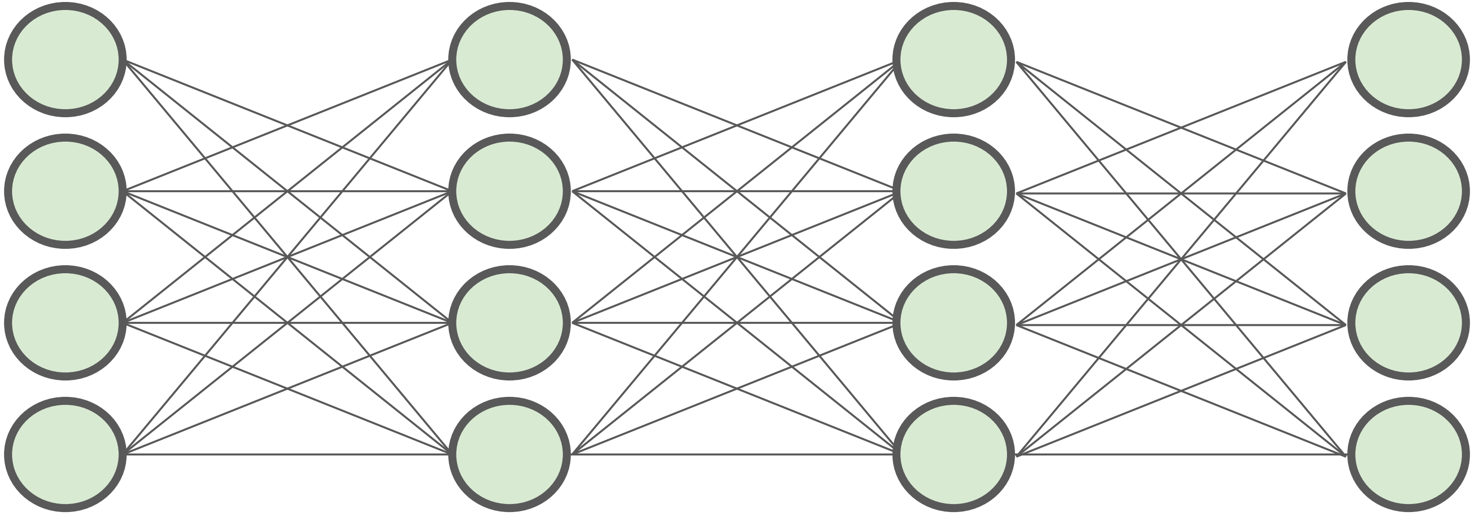
(b) Self-Attention

Illustration by [NVIDIA (Megatron-LM)]



Credit: Fedus et al. (Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity)

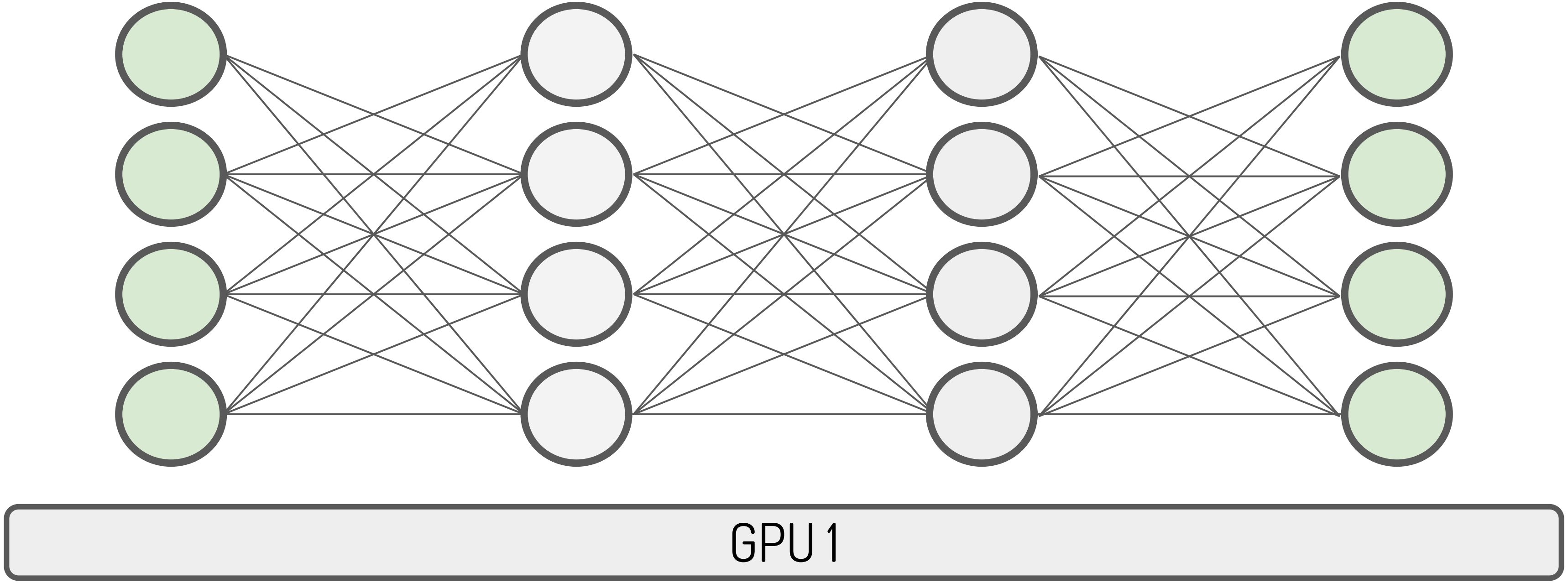
Gradient Checkpointing



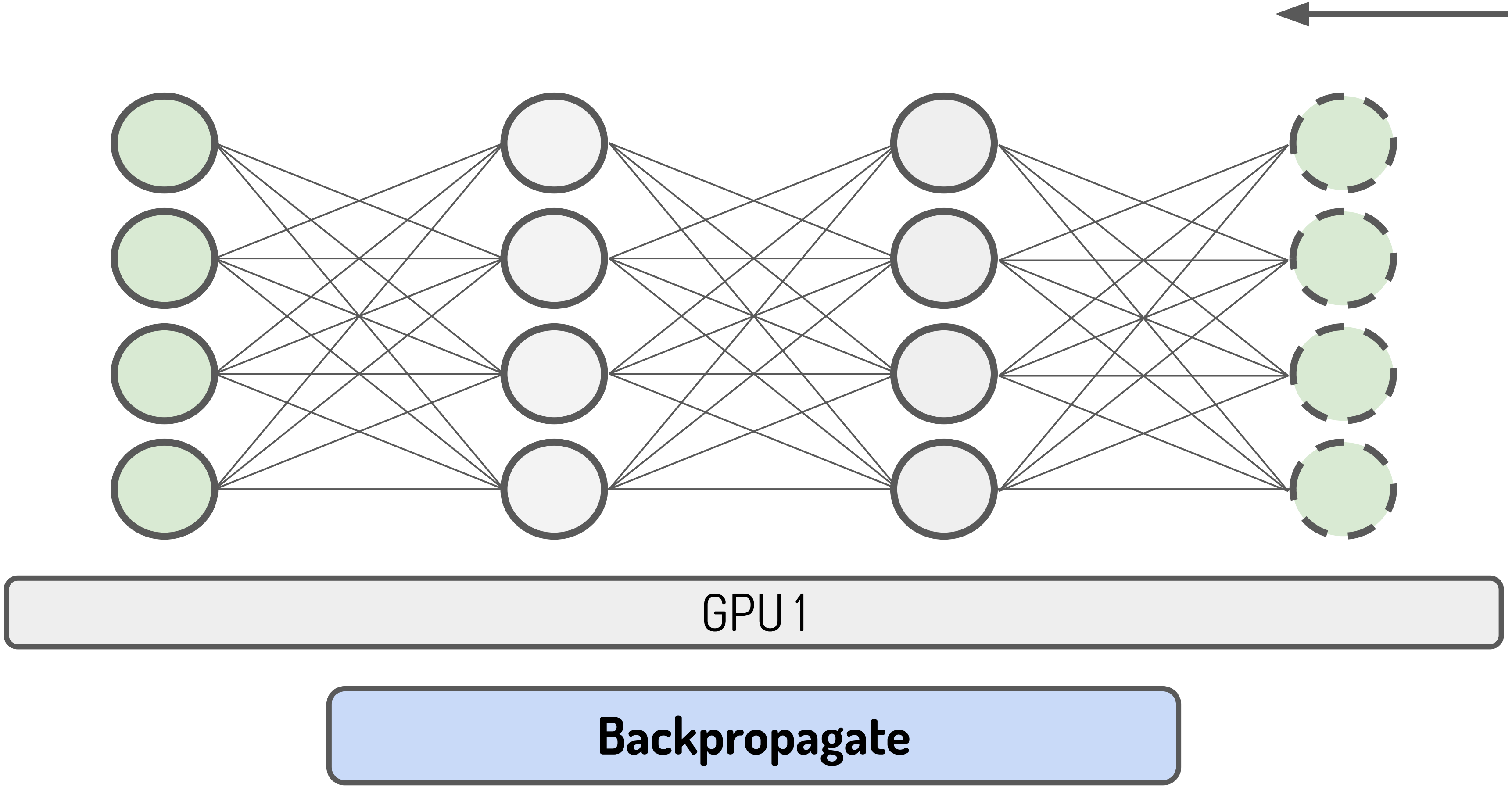
GPU 1

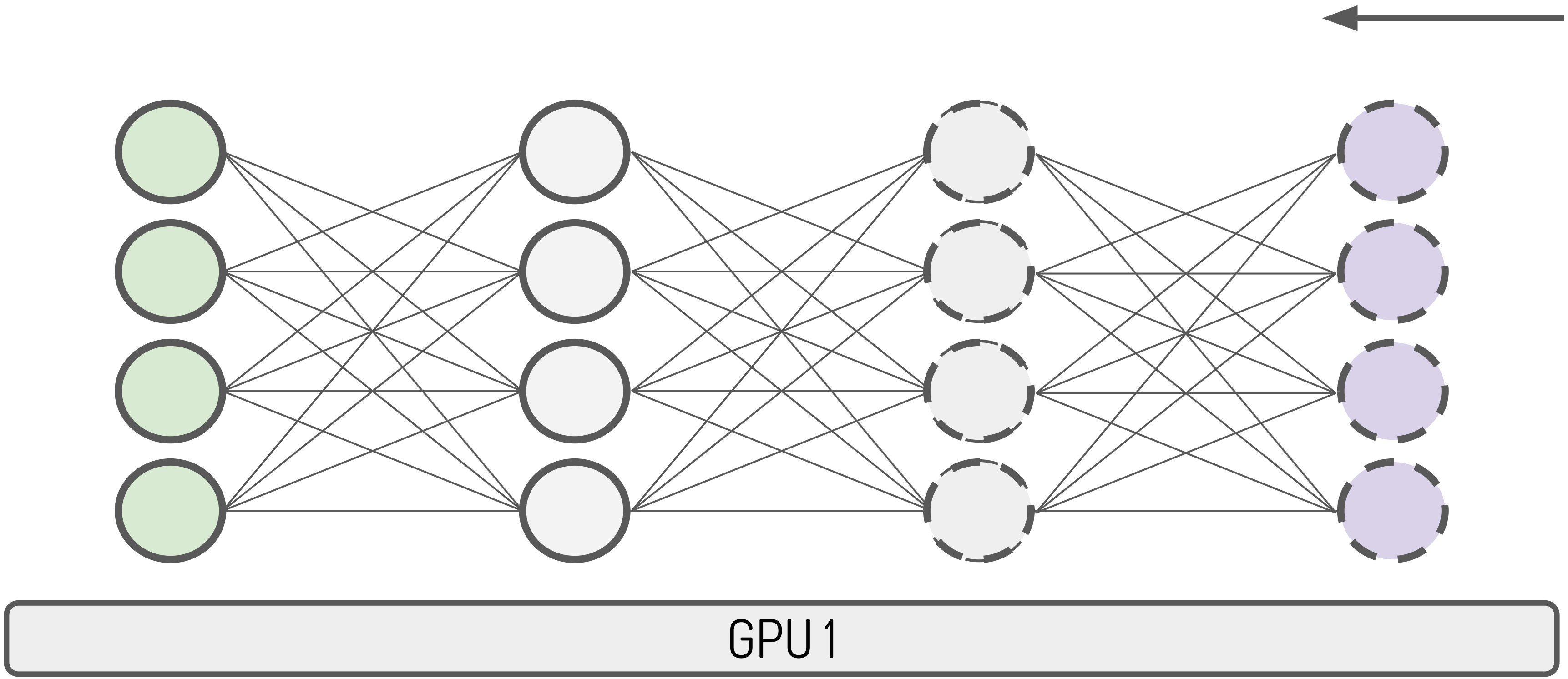
Trade off memory for compute

Gradient Checkpointing

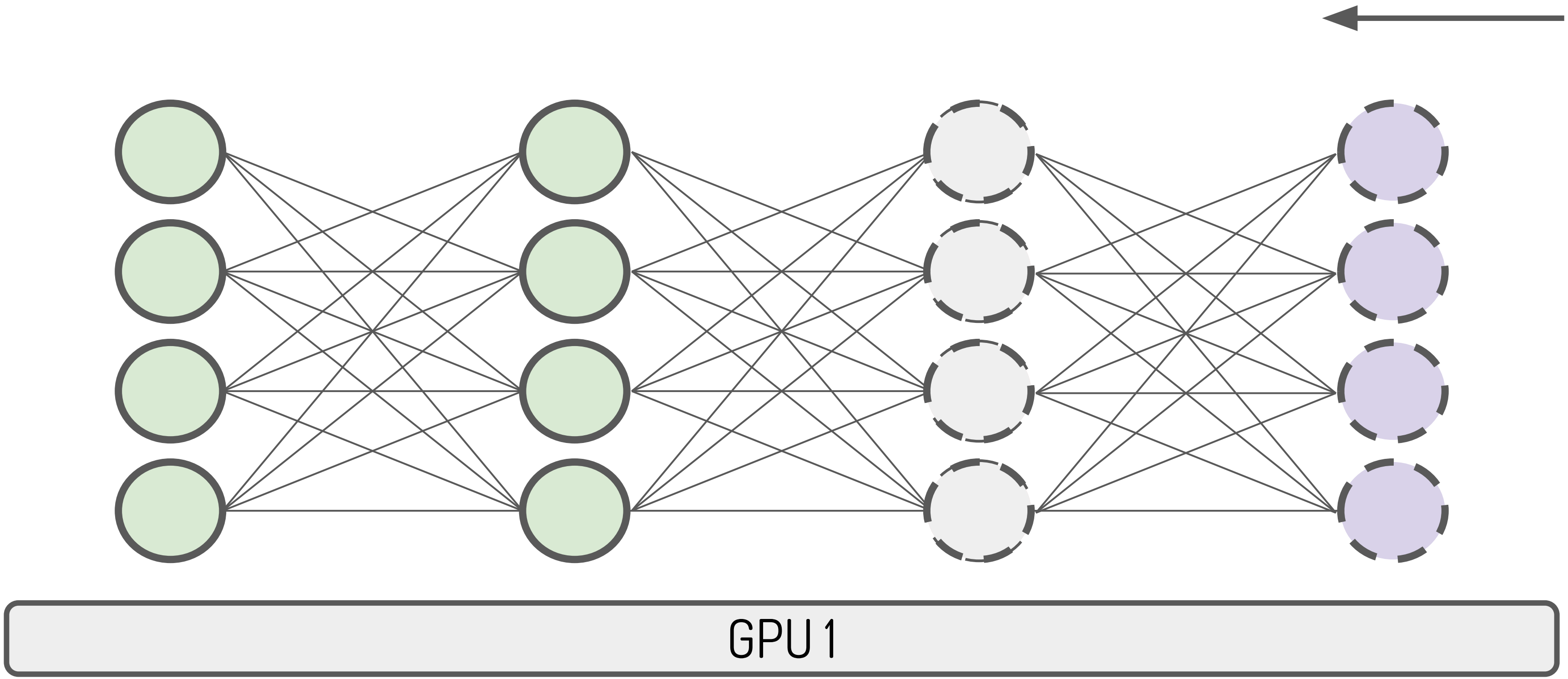


Don’t store some activations in forward pass

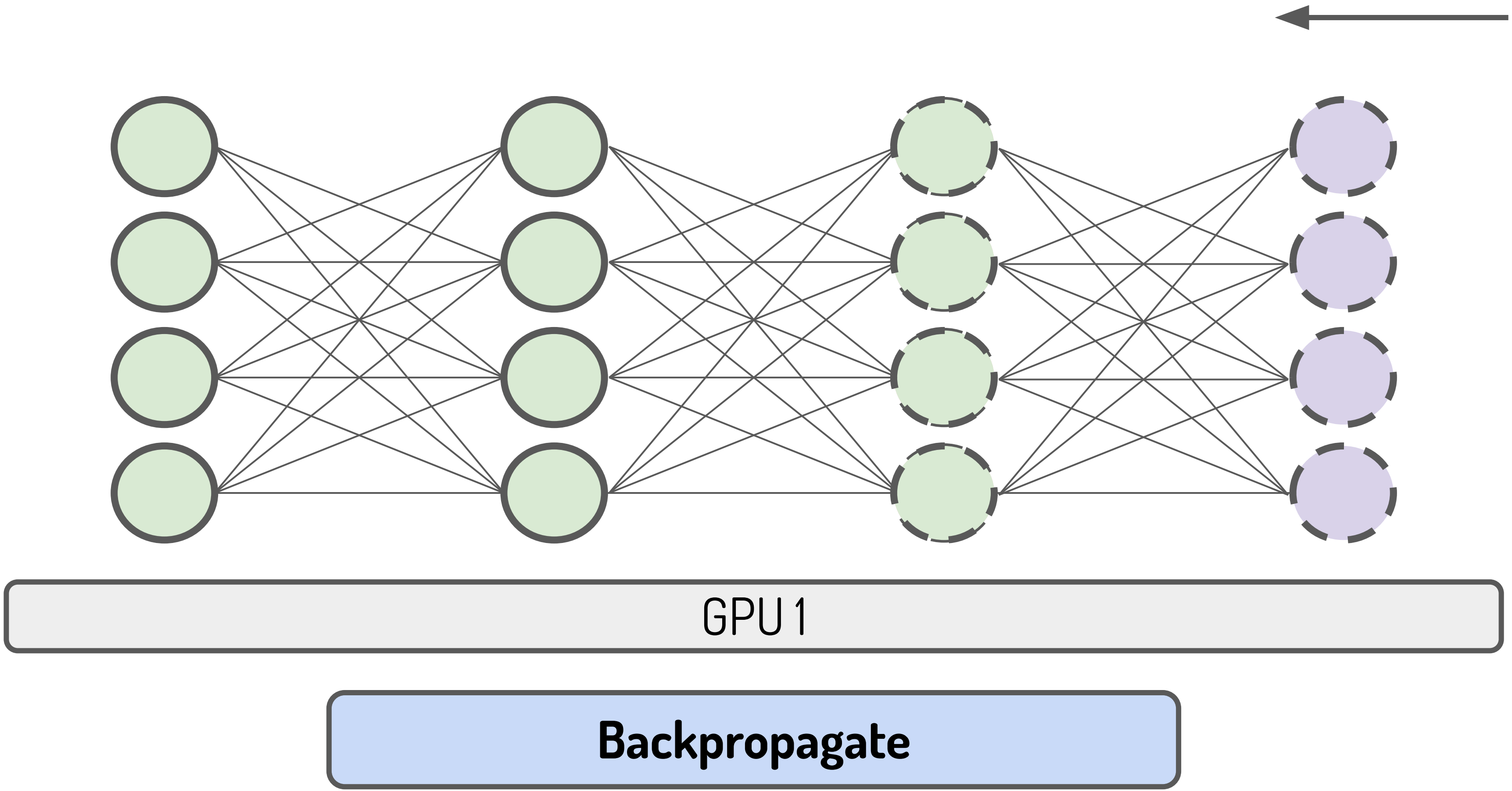

GPU 1
Backpropagate

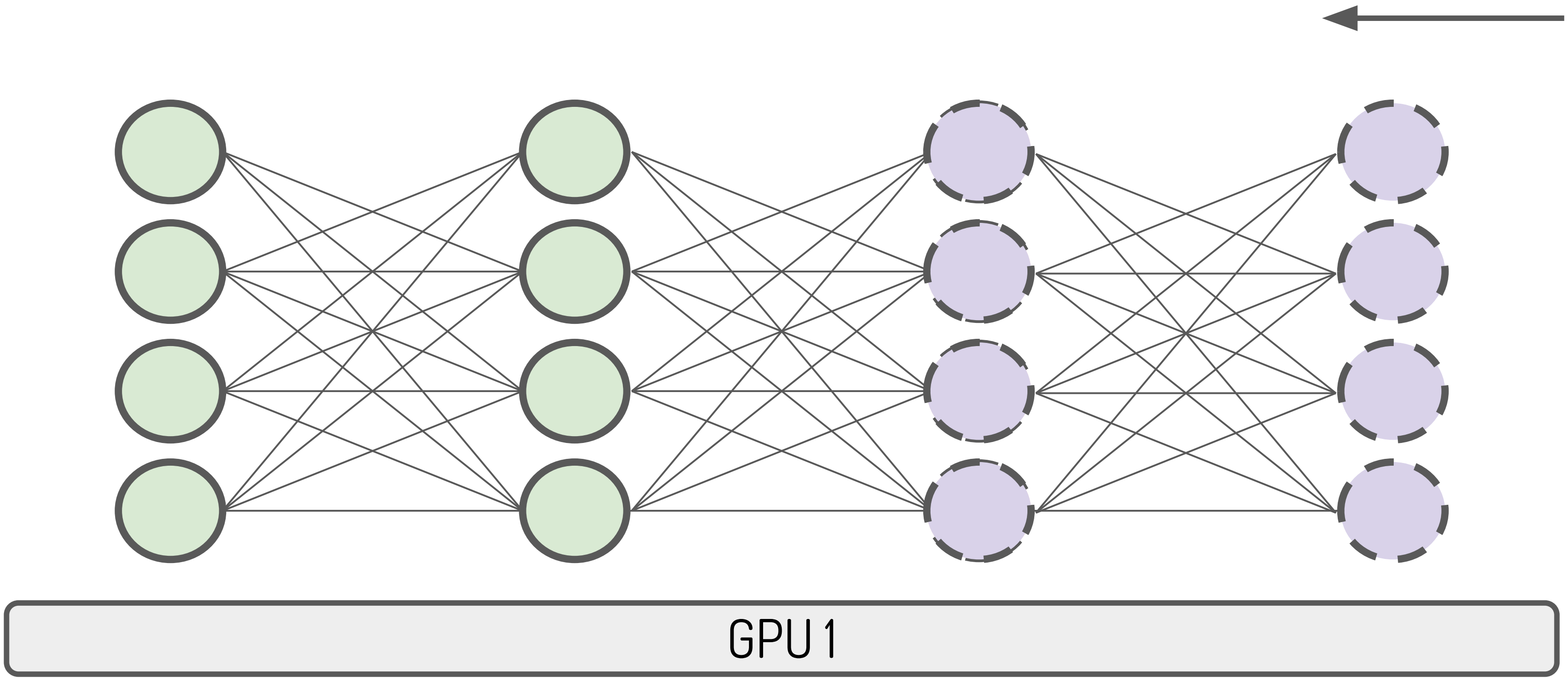


Don’t have activations!

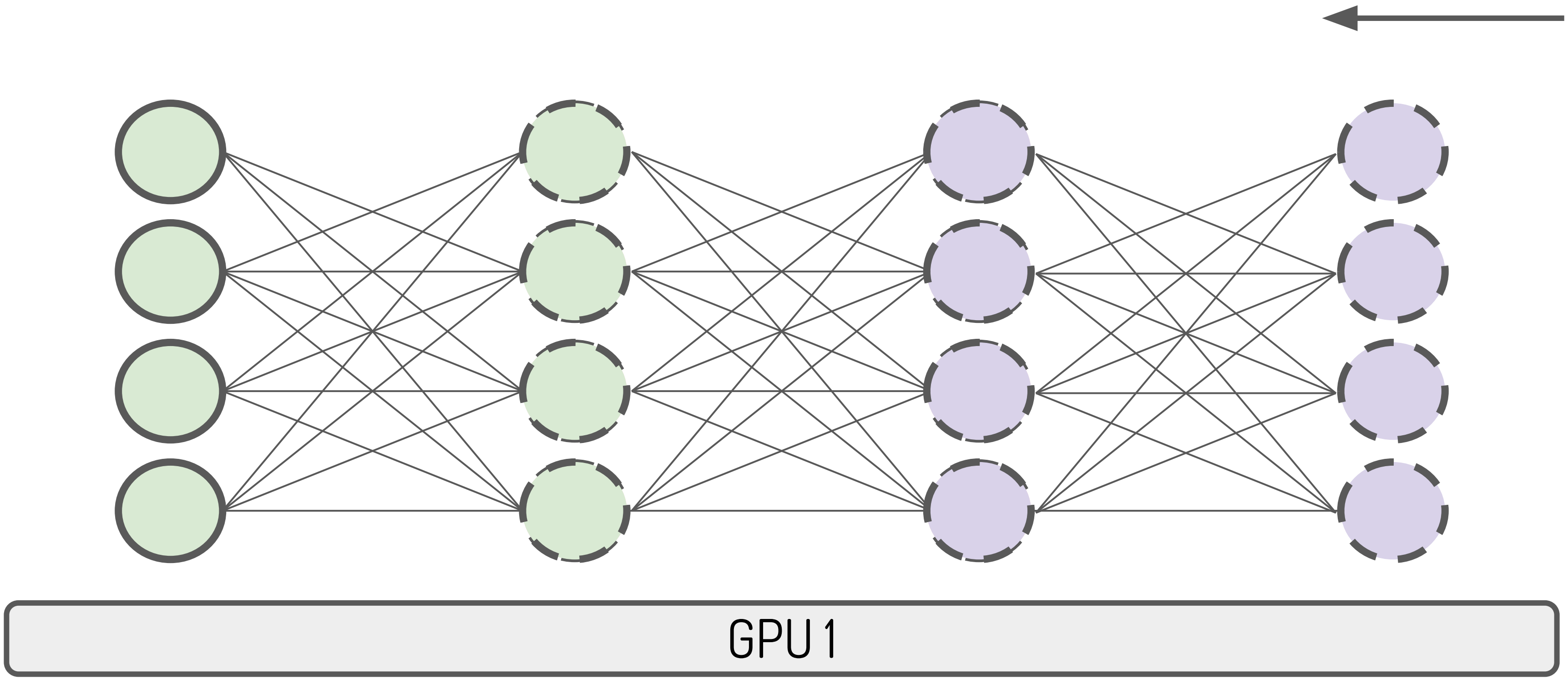


Recompute activations from checkpoint

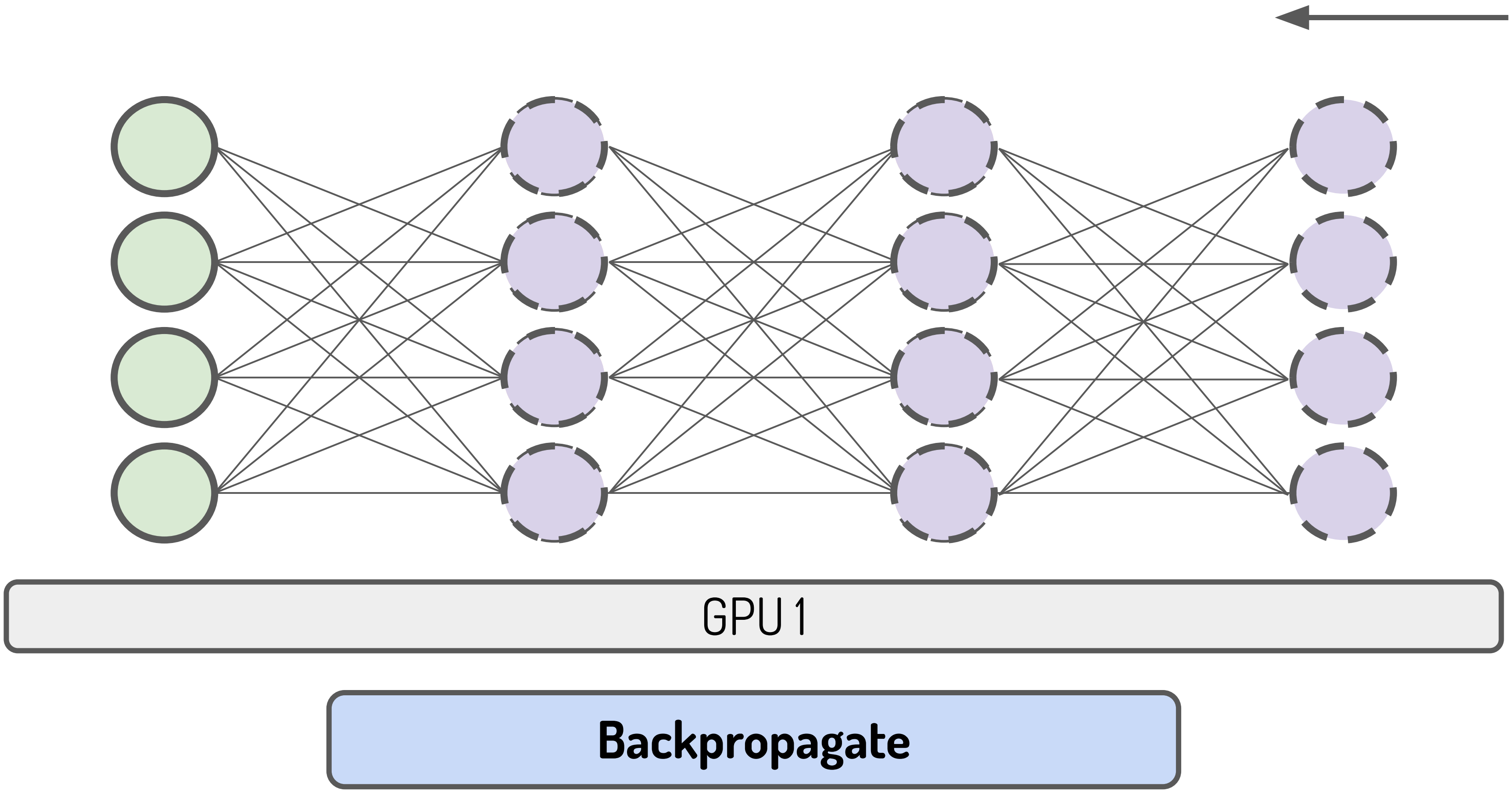

GPU 1
Backpropagate

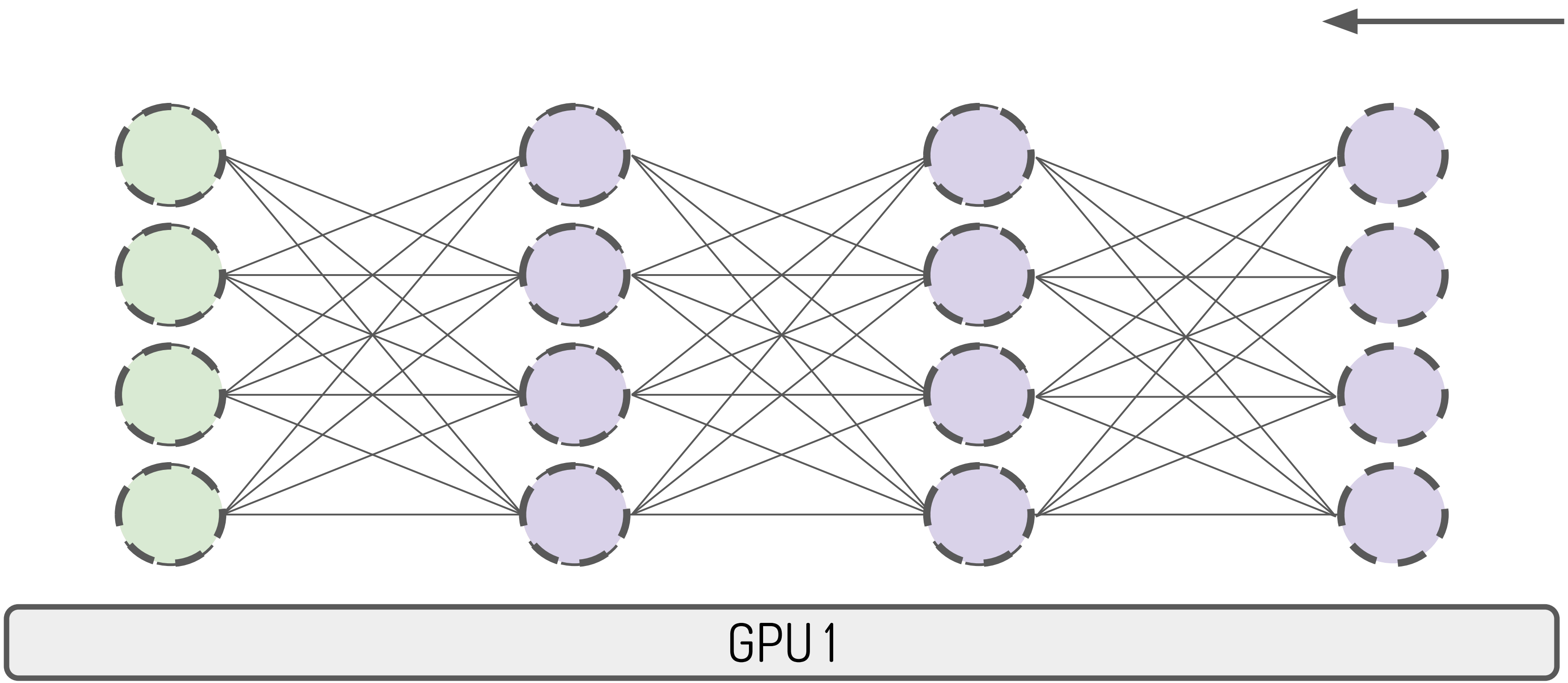


Backpropagate

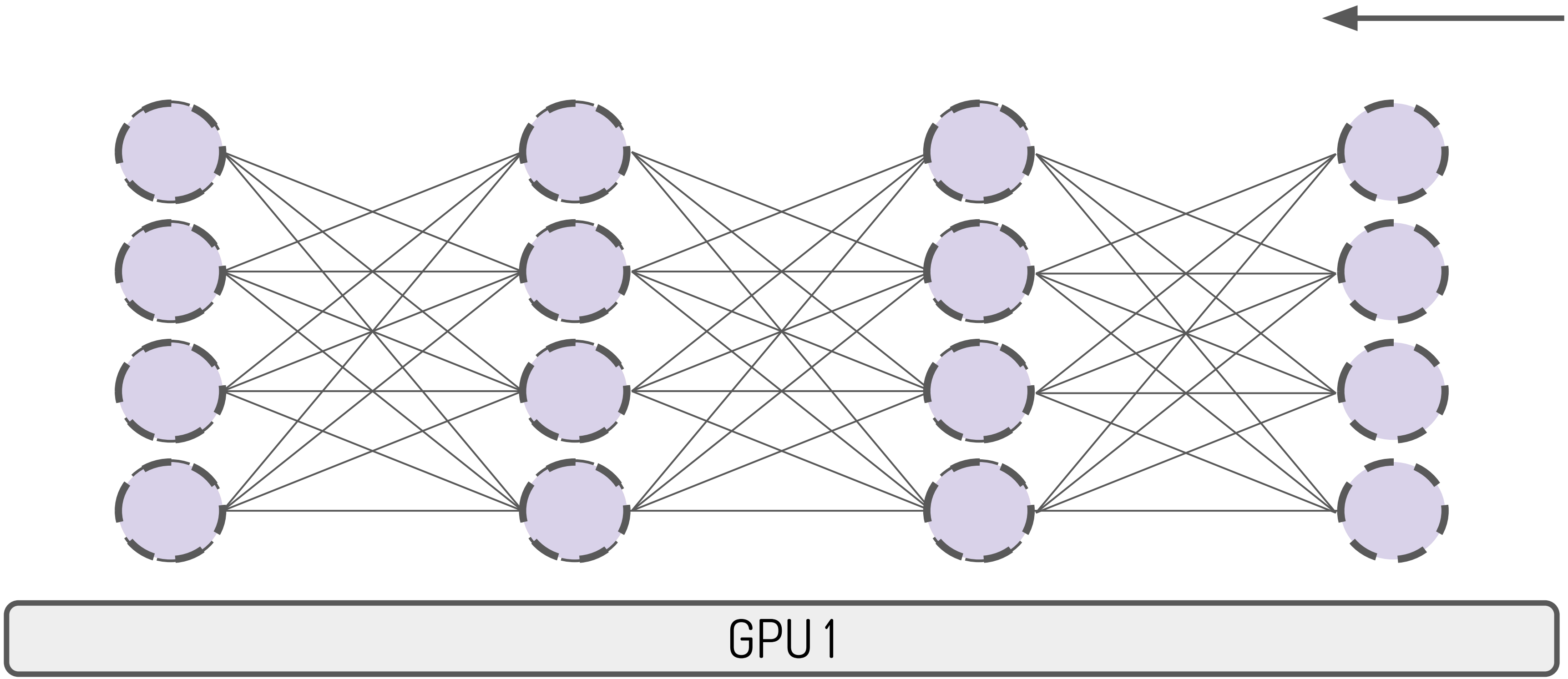


Backpropagate


GPU 1
Backpropagate



Backpropagate



Backpropagate

Machine Learning Systems Design

Modeling Pipeline

Next Lecture: High-Performance Modeling

CE 40959 Spring 2023
Ali Zarezade
SharifMLSD.github.io