- Assignment 2 is due Wednesday AM!
- Course project groups due Oct 11, Monday!
- Project proposal aka Introduction (10/25)
LIMITATIONS OF DATA PARALLEL

“fraction of training time spent in communication stalls”
MODEL PARALLEL TRAINING
Advantages?
CHALLENGE 1: WORK PARTITIONING

Goal: Balanced stages in the pipeline. Why?
Steady state throughput is the throughput of the slowest stage

Stages can be replicated!

![Diagram showing work partitioning with stages and workers.](image-url)
WORK PARITIONING

Profiler: computation time for forward, backward
  size of output activations, gradients (network transfer)
  size of parameters (memory)

Dynamic programming algorithm
Intuition: Find optimal partitions within a server,
  Then find best split across servers using that
CHALLENGE 2: WORK SCHEDULING

Traditional data parallel
forward iter(i)
backward iter(i)
forward iter(i+1)
...

Pipeline parallel: Worker can
Forward pass to push to downstream
Backward pass to push to upstream
CHALLENGE 2: WORK SCHEDULING

Num active batches $\approx$ num_workers / num_replicas_input

Schedule one-forward-one-backward (1F1B)
Round-robin for replicated stages $\rightarrow$
same worker for fwd, backward
CHALLENGE 3: EFFECTIVE LEARNING

Naïve pipelining
Different model versions forward and backward
CHALLENGE 3: EFFECTIVE LEARNING

Weight stashing
   Maintain multiple versions of the weights
   One per active mini-batch

Use latest version for forward pass.
Retrieve for backward
No guarantees across stages!
STALENESS, MEMORY OVERHEAD

How to avoid staleness:
   Vertical sync

Memory overhead
   Similar to data parallel?
SUMMARY

Pipeline parallelism: Combine inter-batch and intra-batch
Partitioning: Replication, dynamic programming
Scheduling: 1F1B
Weight management: Stashing, vertical sync
DISCUSSION

https://forms.gle/j2GCDyqCejBH8DaCA
List two takeaways from the following table

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Model Size</th>
<th>GPUs (#Servers x #GPUs/Server)</th>
<th>PipeDream Config</th>
<th>Speedup over DataParallel (Epoch Time)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resnet-50</td>
<td>97MB</td>
<td>4x4 2x8</td>
<td>16 16</td>
<td>1x</td>
</tr>
<tr>
<td>VGG-16</td>
<td>528MB</td>
<td>4x4 2x8</td>
<td>15-1 15-1</td>
<td>5.28x 2.98x</td>
</tr>
<tr>
<td>GNMT-8</td>
<td>1.1GB</td>
<td>3x4 2x8</td>
<td>Straight 16</td>
<td>2.95x 1x</td>
</tr>
</tbody>
</table>
What are some other workload scenarios (e.g. things we discussed for MapReduce or Spark) that could use similar ideas of pipelined parallelism? Develop such one example and its execution
Next class: Ray

Assignment 2 is due soon!
Course project: Oct 11 (Monday) Submit titles, groups