CS 744: PYTORCH

Shivaram Venkataraman

Fall 2020

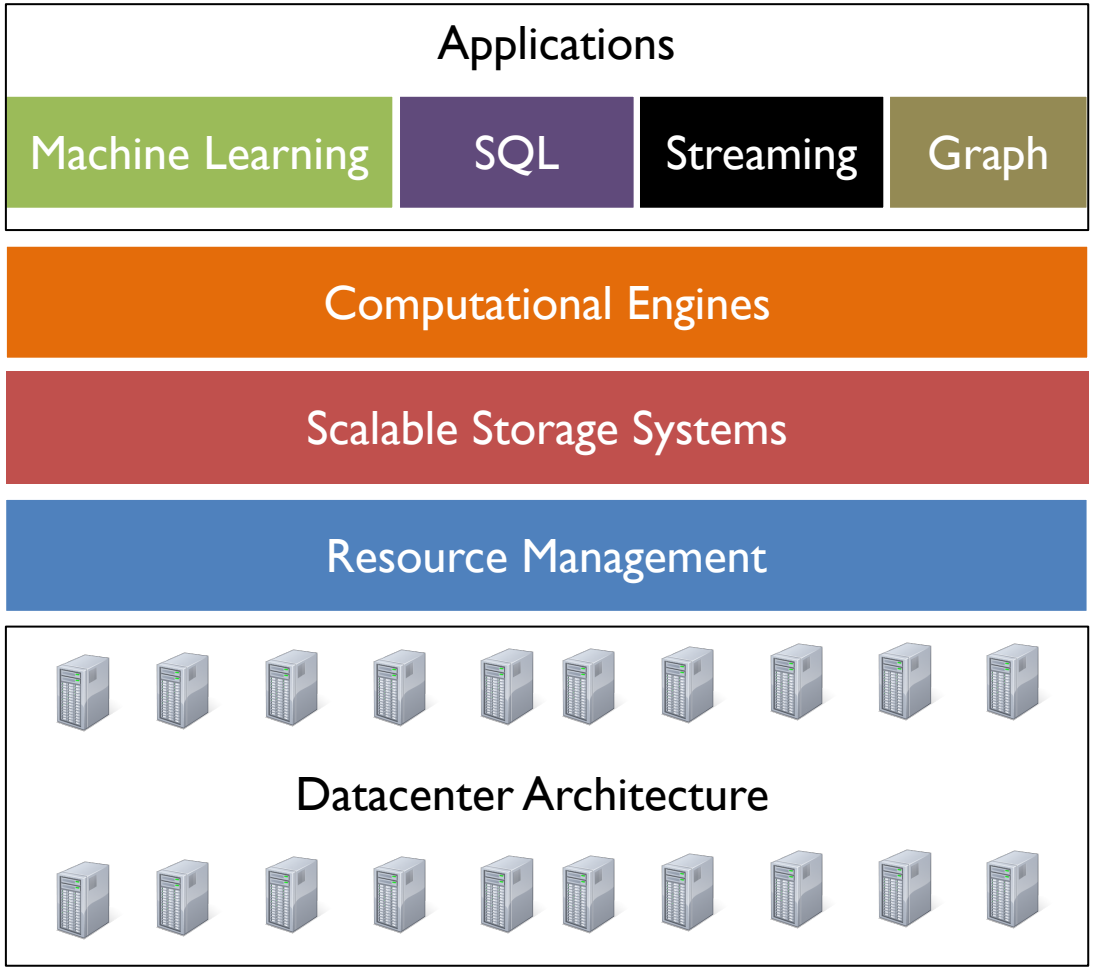
ADMINISTRIVIA

Assignment 2 out!

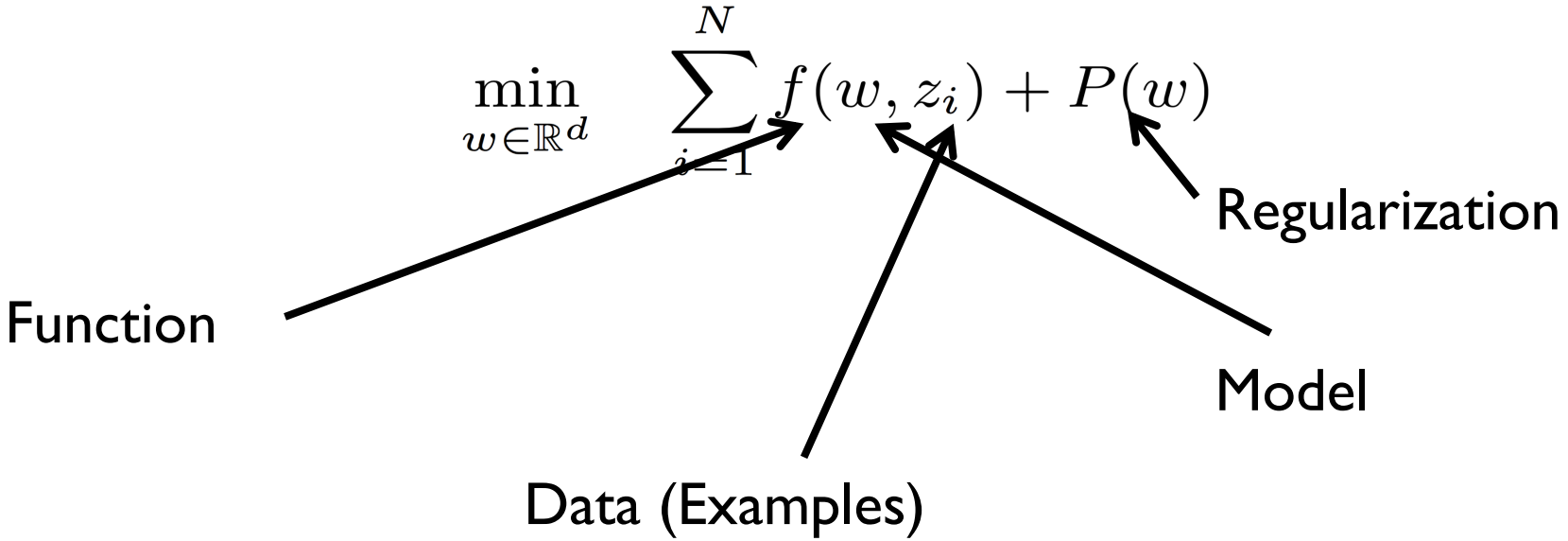
Bid on topics, submit group (1 sentences) – Oct 5

Project Proposal (2 pages) – Oct 16

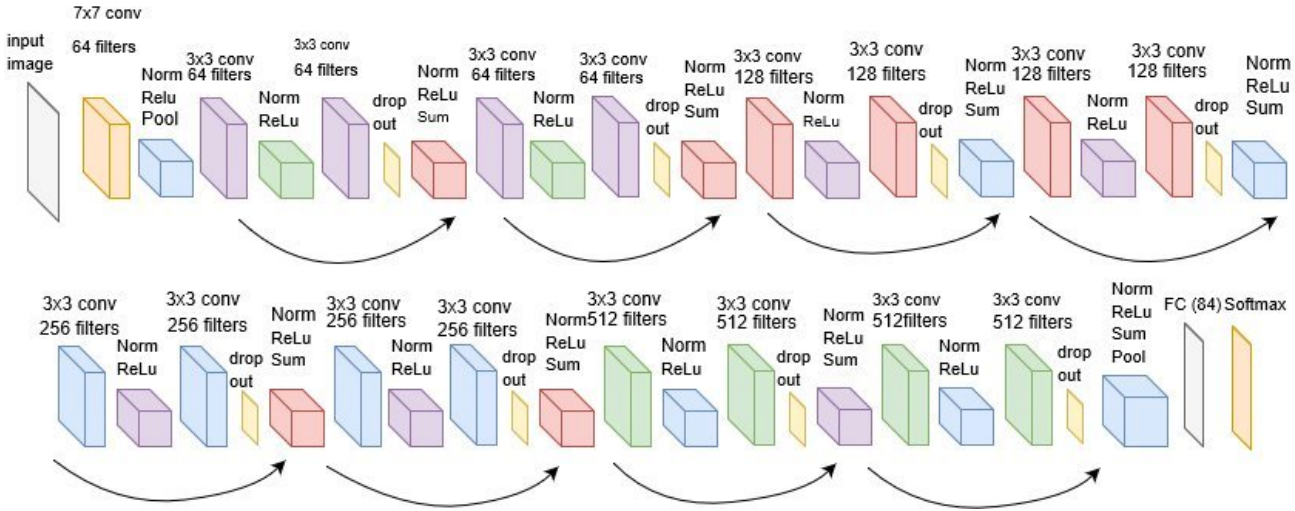
- Introduction
- Related Work
- Timeline (with eval plan)


Applications
Machine Learning
SQL
Streaming
Graph
Computational Engines
Scalable Storage Systems
Resource Management
Datacenter Architecture

EMPIRICAL RISK MINIMIZATION



DEEP LEARNING



ResNet18

Convolution
ReLU
MaxPool
Fully Connected
SoftMax

STOCHASTIC GRADIENT DESCENT

$$w^{(k+1)} = w^{(k)} - \alpha_k \nabla f(w^{(k)})$$

Initialize w

For many iterations:

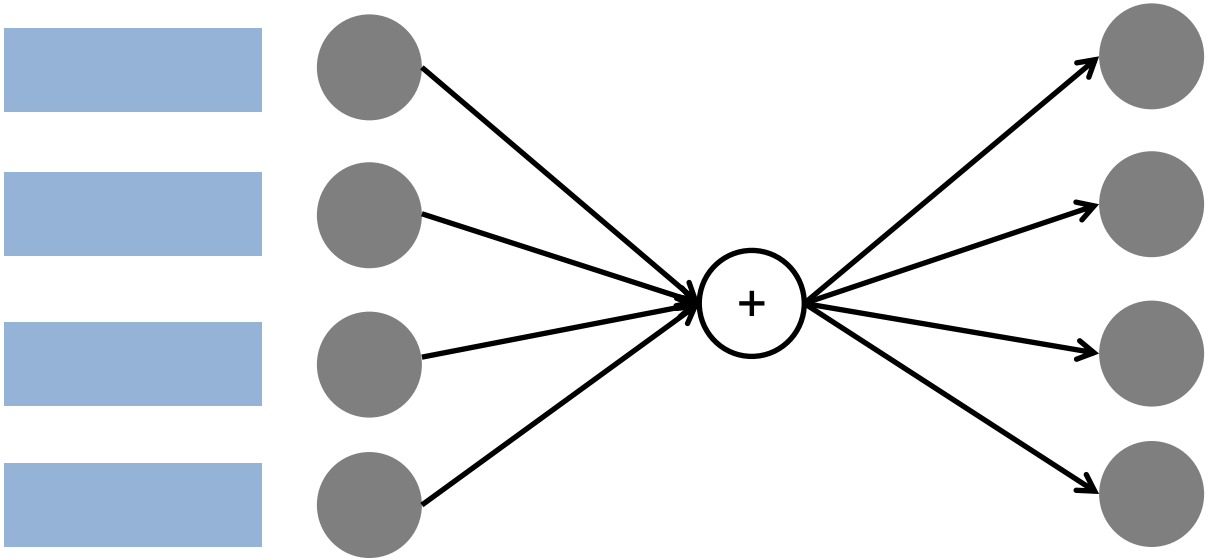
    Loss = Forward pass

    Gradient = backward

    Update model

End

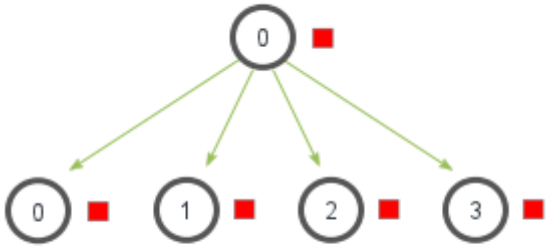
DATA PARALLEL MODEL TRAINING



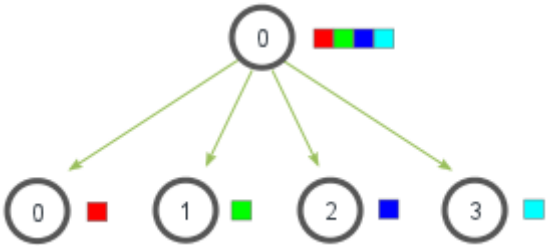
COLLECTIVE COMMUNICATION

Broadcast, Scatter

MPI_Bcast

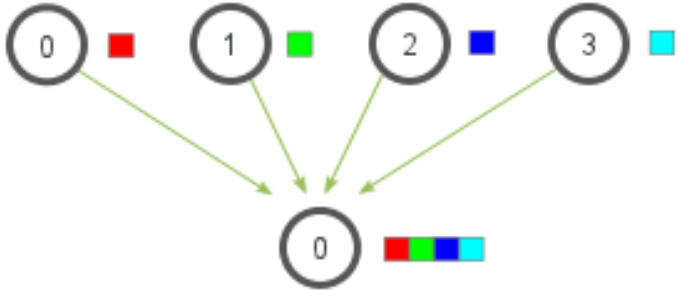


MPI_Scatter

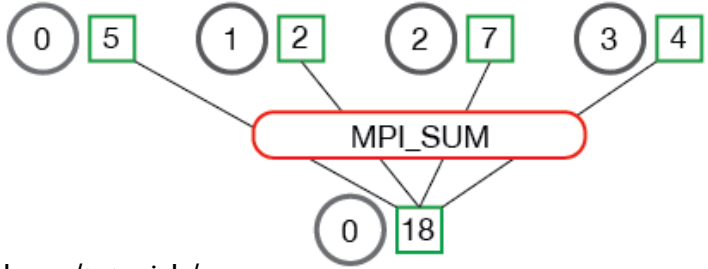


Gather, Reduce

MPI_Gather

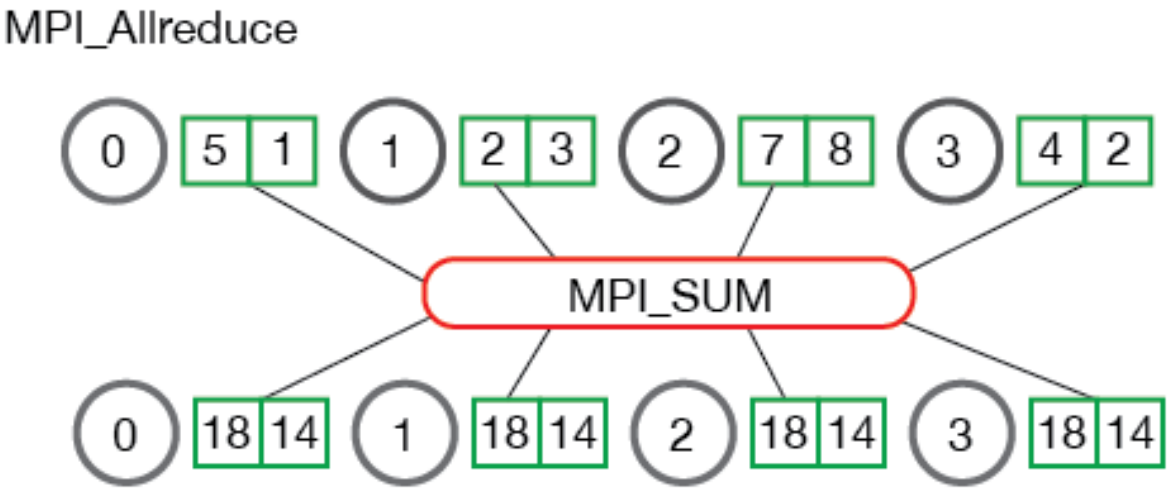


MPI_Reduce



From https://mpitutorial.com/tutorials/

ALL REDUCE



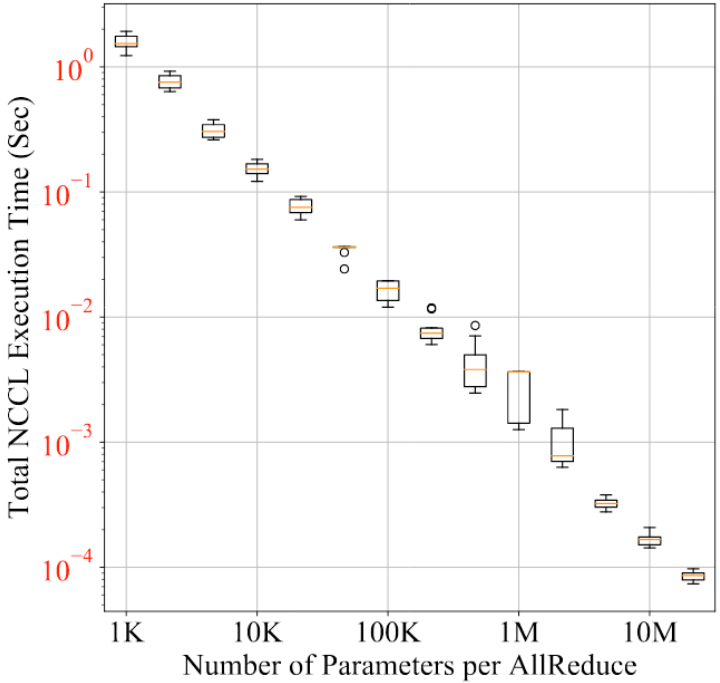
From https://mpitutorial.com/tutorials/

DISTRIBUTED DATA PARALLEL API

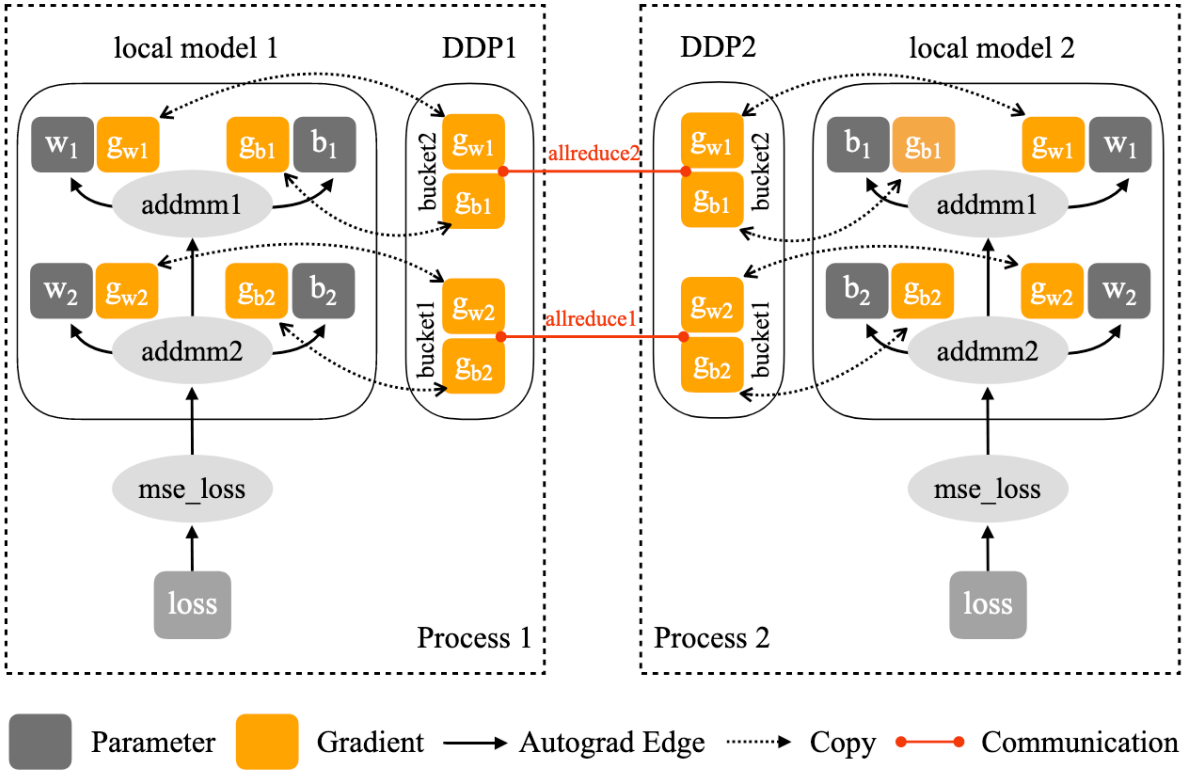
```
9   # setup model and optimizer
10  net = nn.Linear(10, 10)
11  net = par.DistributedDataParallel(net)
12  opt = optim.SGD(net.parameters(), lr=0.01)
13
14  # run forward pass
15  inp = torch.randn(20, 10)
16  exp = torch.randn(20, 10)
17  out = net(inp)
18
19  # run backward pass
20  nn.MSELoss()(out, exp).backward()
21
22  # update parameters
23  opt.step()
```

GRADIENT BUCKETING

Why do we need gradient bucketing?



GRADIENT BUCKETING + ALL REDUCE



GRADIENT ACCUMULATION

```
ddp = DistributedDataParallel(net)
with ddp.no_sync():
  for inp, exp in zip(inputs, expected_outputs):
    # no synchronization, accumulate grads
    loss_fn(ddp(inp), exp).backward()
# synchronize grads
loss_fn(ddp(another_inp), another_exp).backward()
opt.step()
```

IMPLEMENTATION

Bucket_cap_mb

Parameter-to-bucket mapping

Round-robin ProcessGroups

BREAKDOWN

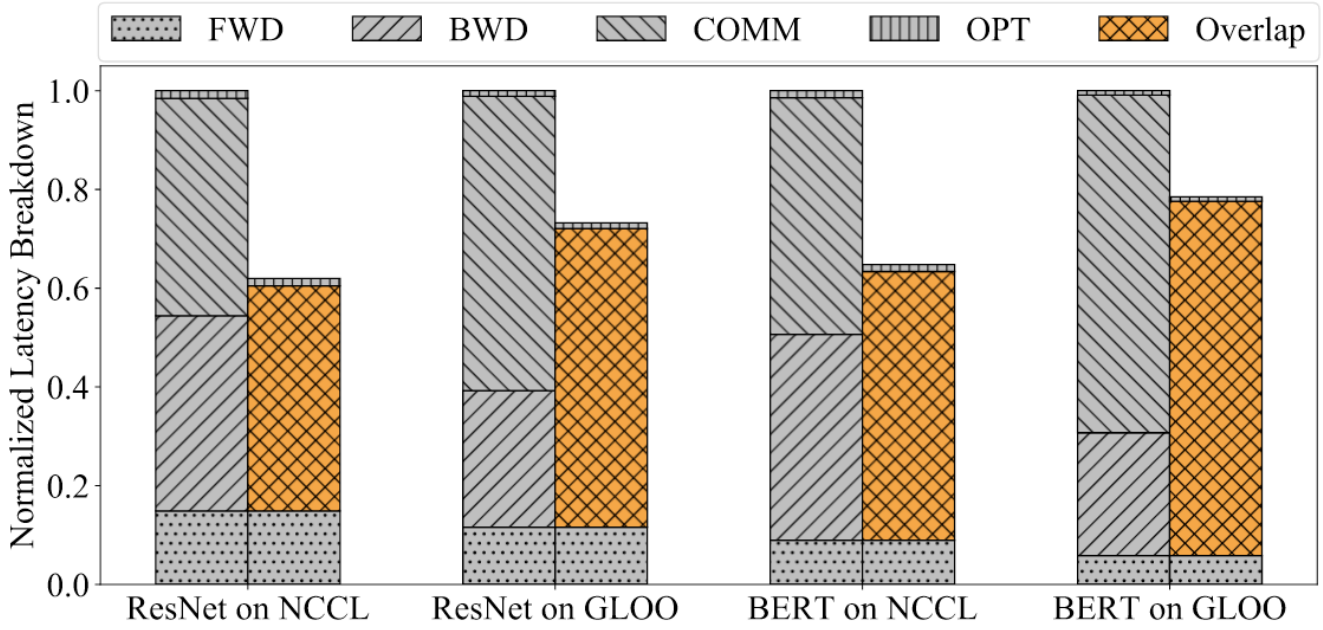


Figure 6: Per Iteration Latency Breakdown

SUMMARY

Pytorch: Framework for deep learning

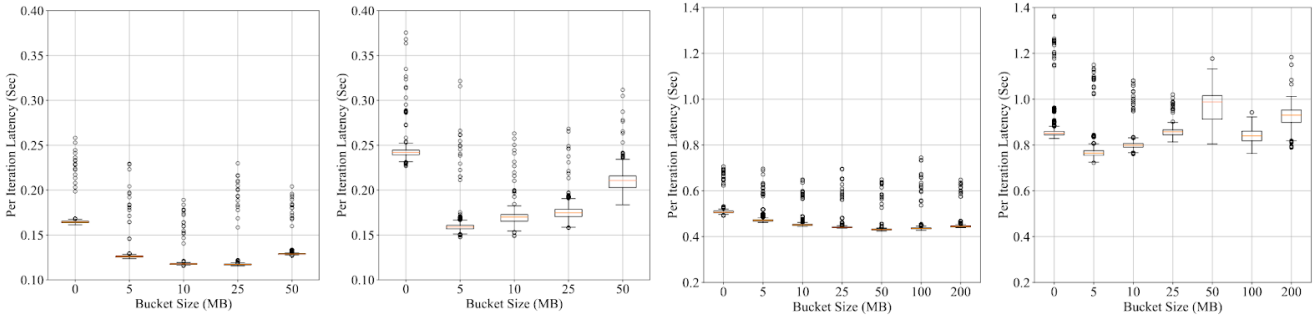
DistributedDataParallel API

Gradient bucketing, AllReduce

Overlap computation and communication

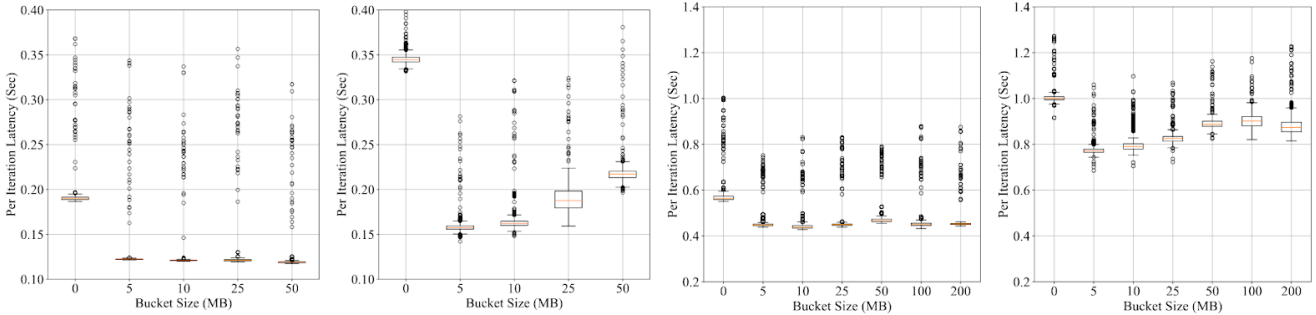
DISCUSSION

https://forms.gle/6xhVBNBhdzsJ6gBE6



(a) ResNet50 on NCCL (b) ResNet50 on Gloo (c) BERT on NCCL (d) BERT on Gloo

Figure 7: Per Iteration Latency vs Bucket Size on 16 GPUs



(a) ResNet50 on NCCL (b) ResNet50 on Gloo (c) BERT on NCCL (d) BERT on Gloo

Figure 8: Per Iteration Latency vs Bucket Size on 32 GPUs

What could be some challenges in implementing similar optimizations for AllReduce in Apache Spark?

NEXT STEPS

Next class: PipeDream

Assignment 2 is due soon!

Project Proposal

- Groups by Oct 5
- 2 pager by Oct 16