HAL Benchmarks 2020

ImageNet Distributed Mixed-precision Training Benchmark

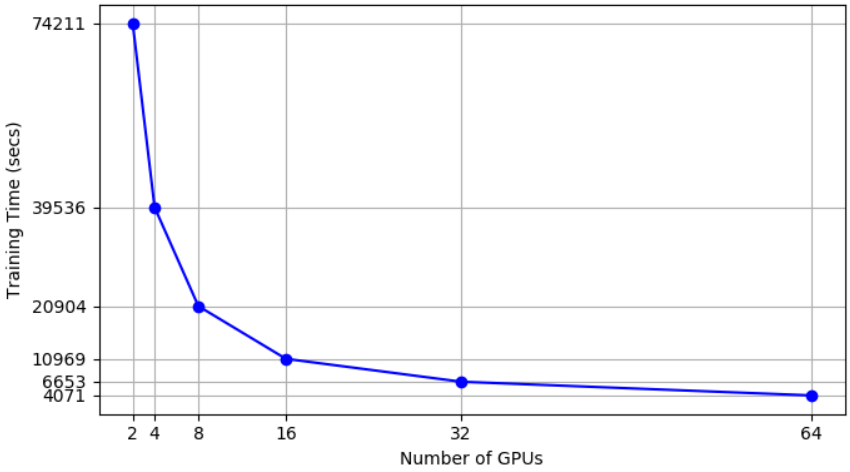
Github repo for all source code and details: https://github.com/richardkxu/distributed-pytorch

Jupyter notebook tutorial for the key points: https://github.com/richardkxu/distributed-pytorch/blob/master/ddp_apex_tutorial.ipynb

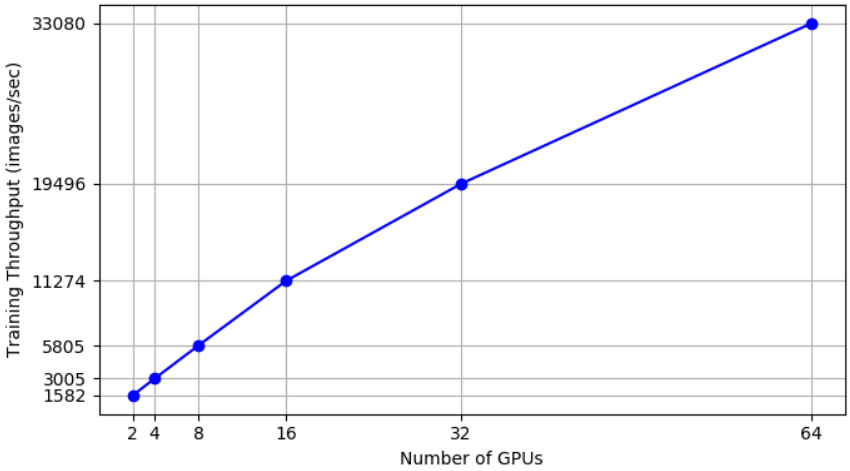
HAL paper: https://dl.acm.org/doi/10.1145/3311790.3396649

Benchmark Results

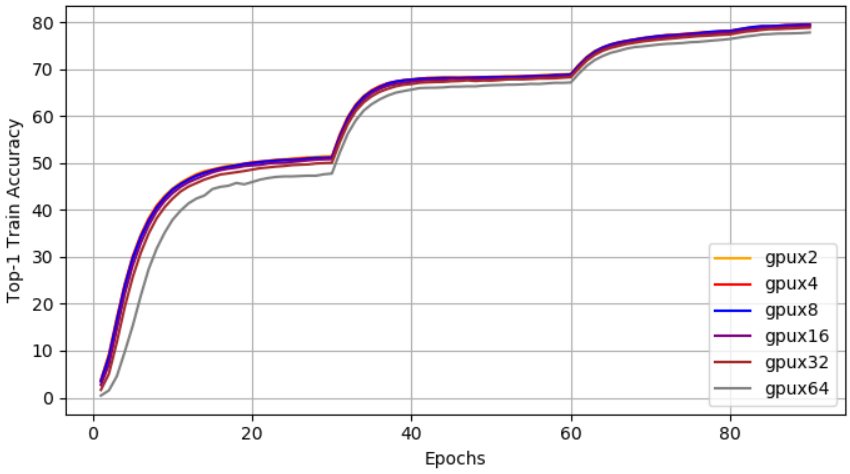
Training Time: Time to solution during training. The number of GPUs ranges from 2 GPUs to 64 GPUs. ImageNet training with ResNet-50 using 2 GPUs takes 20.00 hrs, 36.00 mins, 51.11 secs. With 64 GPUs across 16 compute nodes, we can train ResNet-50 in 1.00 hr, 7.00 mins, 51.31 secs, while maintaining the same level of top1 and top5 accuracy.

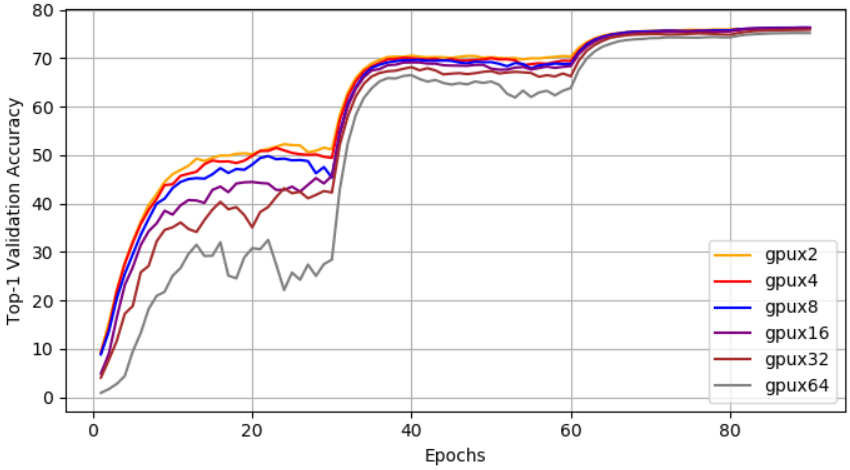


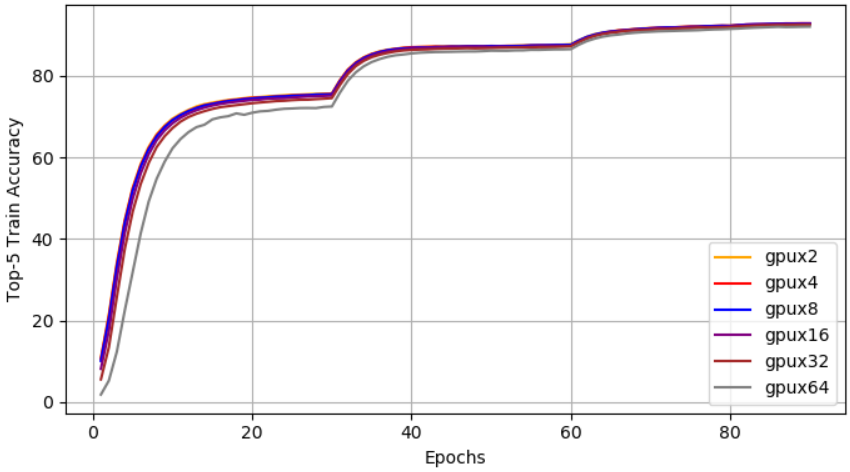
Training Throughput: Scalability of training global througput (images/sec) with respect to the number of GPUs. We are able to achieve near linear scaling in training throughput. Note that training from 2 GPUs to 4 GPUs is within node scaling, while the training from 4 GPUs to 64 GPUs is across node scaling.

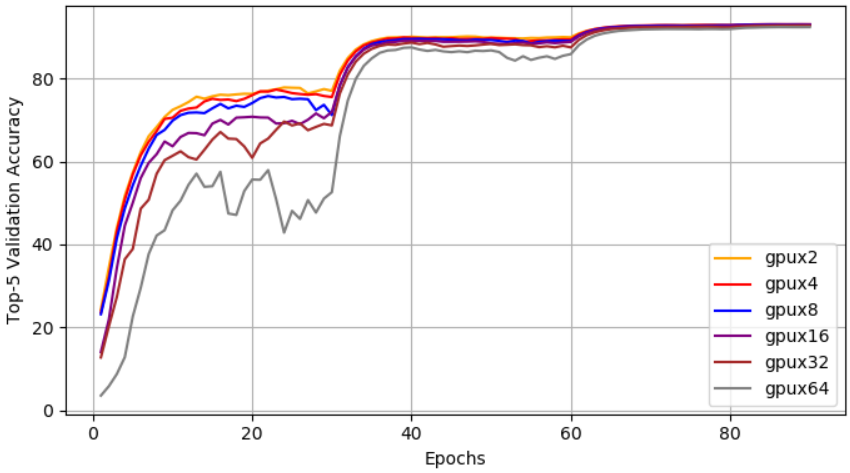


Top1, Top5 Accuracy: We achieve distributed training speed-up without the loss of accuracy. All experiments reached a top1 accuracy of 76% and a top5 accuracy of 93%. All experiments have very similar learning curve. We do notice that with larger global batch size, the training is more unstable at early stages (epoch 1-30). However, the small and large batch size training curves match closely after 30 epochs and they all reach the same accuracy at the end.

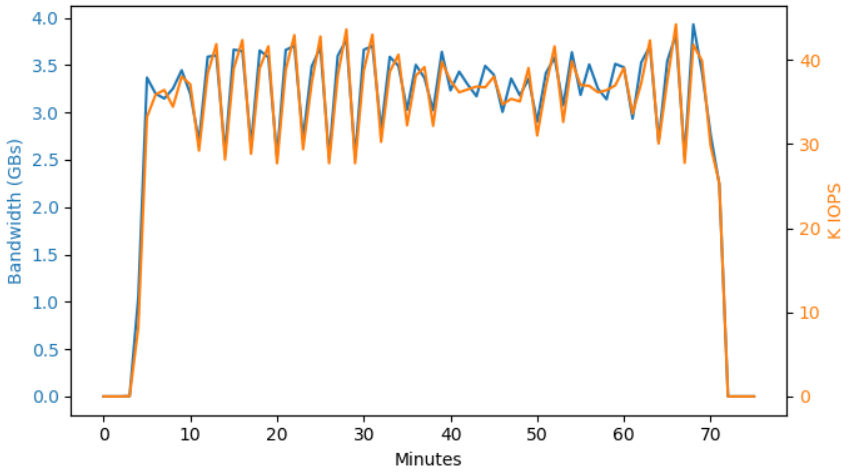

Top-1 Train Accuracy
Epochs
gpux2
gpux4
gpux8
gpux16
gpux32
gpux64


Top-1 Validation Accuracy
Epochs
gpux2
gpux4
gpux8
gpux16
gpux32
gpux64


Top-5 Train Accuracy
Epochs
gpux2
gpux4
gpux8
gpux16
gpux32
gpux64



I/O Bandwidth: I/O Bandwidth (GB/s) and IOPS of our file system throughout our full system ImageNet training using 64 GPUs. Between 10th and 60th epoch, the average bandwidth is 3.30 GB/s and the average IOPS is 36.5K.



Software Stack

- IBM WMLCE 1.6.2
- Python 3.7
- PyTorch 1.2.0
- NVIDIA Apex 0.1.0
- CUDA 10.1

Benchmark Details

To demonstrate the performance and scalability of our system, we conduct ImageNet32 training by scaling ResNet-50 across multiple GPUs and multiple computer nodes. We use the official implementation of ResNet-50 by PyTorch. We use a standard momentum with ?? of 0.9 and a weight decay ?? of 0.0001. All models are trained for 90 epochs regardless of batch sizes. We perform the learning rate scaling and gradual warmup to tackle training instability at early stages for large batch size.

Each of our compute nodes has 4 NVIDIA Volta V100 GPUs. We scale ResNet-50 from 2 GPUs on the same computer node to 64 GPUs across 16 compute node, doubling the number of GPUs for each intermediate runs. We use a per-GPU batch size of 208 images, which is the largest batch size we can fit with distributed mixedprecision training. Therefore, our global batch size ranges from 416 images to 13312 images. We use Automatic Mixed Precision (Amp) and Distributed Data Parallel (DDP) from NVIDIA Apex for mixed-precision and distributed training. An optimization level of "O2" is used for mixed-precision training to benefit from FP16 training while keeping a few parameters to be FP32. We use NVIDIA Collective Communication Library (NCCL) as our distributed backend.

References

- https://arxiv.org/abs/1706.02677
- https://arxiv.org/abs/1709.05011

- https://pytorch.org/
- https://github.com/NVIDIA/apex
- https://developer.nvidia.com/nccl