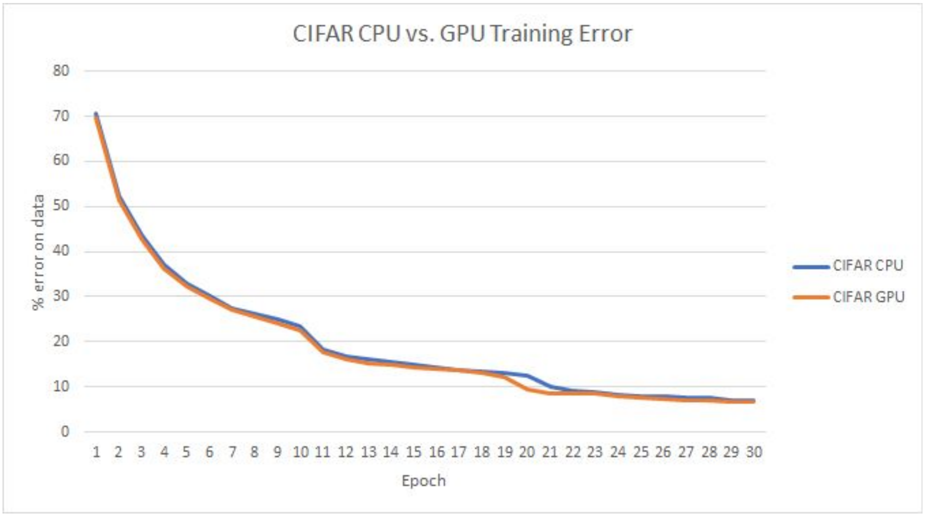
5194 HiDL Lab 1 Report - **CNTK**

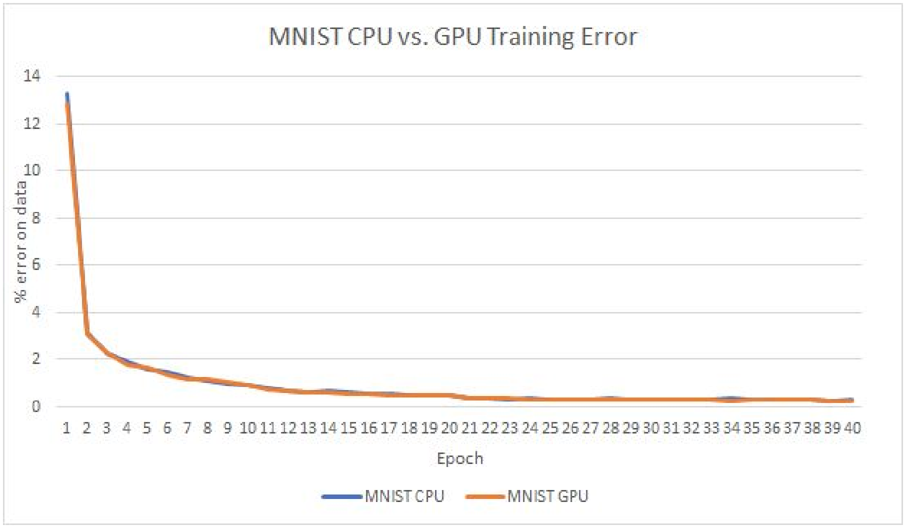
Christopher Britt; Shuangsheng Lou; Anthony J. Cavallero; Kawthar Shafie Khorassani

Based on your experiments, answer the following:

1. (5 points) What are the performance trends for MNIST and CIFAR10 for CPU and GPU?

   We noticed that the CPU node took longer to train than the GPU node. However, the other metrics such as training error were quite similar between them. Testing error for each net can be seen below the trends for CPU and GPU training.





Model:	MNIST CPU	MNIST GPU	CIFAR CPU	CIFAR GPU
Final % Error	.59%	.56%	18.06%	17.47%

We would also like to acknowledge that due to the absence of familiarity with the Cognitive Toolkit that the error rates for the CIFAR-10 dataset are exceptional high, this is likely due to having a model architecture that is insufficient in learning the features associated with classifying the images. However since the goal wasn't to achieve a high classification and low error rate, it was deemed passable to leave the Convolutional Networks as is.

2. (5 points) What DNN model works best on CPU vs. GPU and why?

We observed on Alexnet CPU 28 cores, to Alexnet GPU that the GPU performed almost twice as many samples/second; whereas for ResNet CPU 28 cores to ResNet GPU, the GPU performed almost 10 times as many samples/second. While AlexNet being a simpler model processed a significantly higher samples/second, due to the significantly higher percentile gain we believe that ResNet gains far more benefit from GPUs than Alexnet.

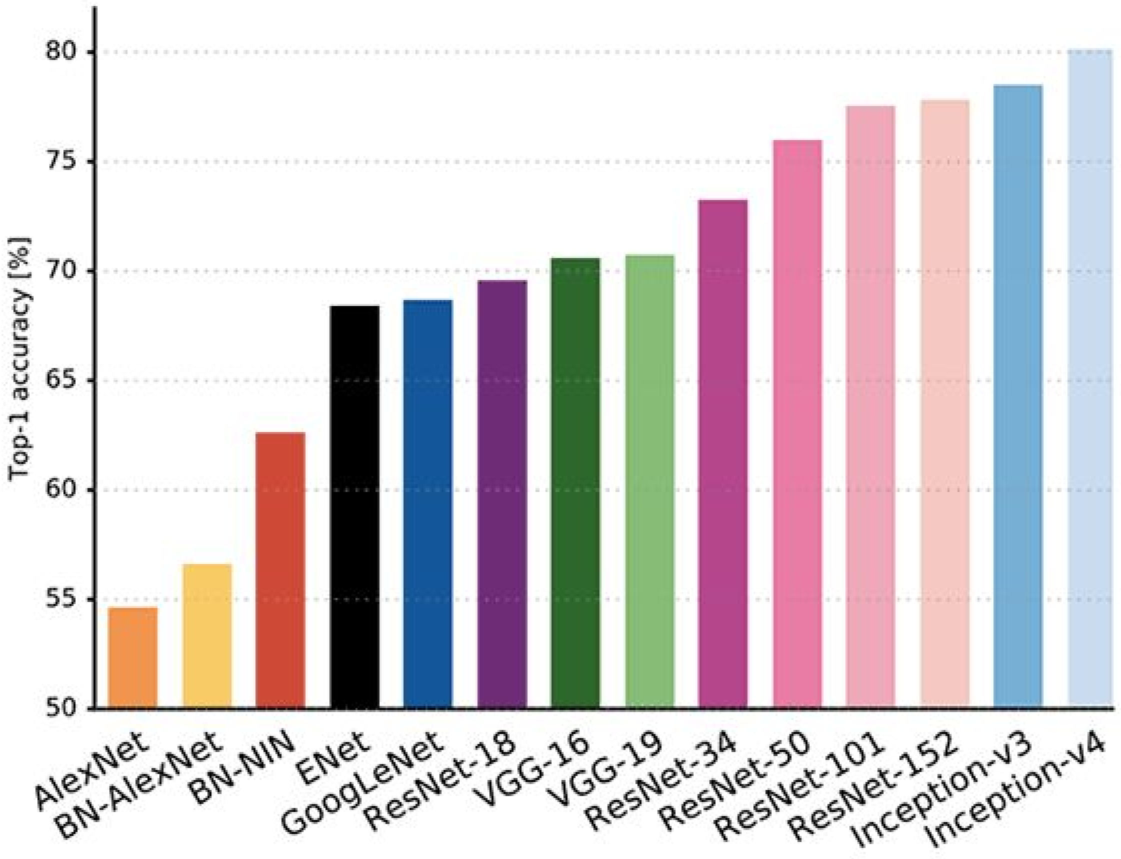
3. (5 points) Mention the scale and size of your dataset and explain why you chose it?

For our dataset we chose to use Tiny ImageNet [https://tiny-imagenet.herokuapp.com/] as the cntk framework appeared to heavily favor convolutional classification problems. We opted to use Tiny Imagenet (100k 64x64 images) rather than the full Imagenet due to difficulty in obtaining the full dataset.

4. (5 points) What is the best DNN architecture for your chosen dataset in terms of "accuracy" and in terms of "training time"? Describe if you find a suitable trade- off between these two metrics.

While, as mentioned in our conclusion, we were unable to acquire meaningful data on model accuracy, Alexnet processed almost 10 times as many samples/second. Thus we have concluded from a computational performance standpoint it is by far the superior model.

However, if we were able to get meaningful data on the accuracy and error of the networks, we would expect it to follow a pattern similar to the graph below on Top-1 Accuracy for different convolutional model including Alexnet and Resnet. This would mean that in the end, Alexnet would end up having a poorer accuracy when compared to Resnet on this dataset. What it really comes down to is given the accuracy of our model is the time to train and evaluate worth it? Even if Resnet performs better in the general case, in different applications we may not even need such a high classification rate. In this scenario Alexnet would be the best network to use if time was limited and you did not need a very accurate network. However if you did have freedom in the amount of time to spend training and testing your network with the goal of generating the best classification possible then Resnet would be your best option.



5. (5 points) Is the GPU always better than CPU for training with larger dataset? Explain your observation.

From our data that we were able to collect, the GPU performed significantly better than the CPU for both Alexnet and Resnet from a samples/second perspective. However we acknowledge the possibility that there may be certain datasets or problem domains that are better processed by CPUs rather than GPUs.

6. (5 points) For the single-node CPU run, do you see any trend when varying the number of cores and number of threads in the application? Explain your observation.

We noticed on Alexnet that by cutting the number of cores and threads in half from 28 to 14, that we moved from processing ~260 samples/second to processing ~200 samples/second. This demonstrates a rather significant overhead for the multithreading as a 100% increase in computational resources results in only a 30% increase in samples/second. We also see varying performance on Resnet as we move from 14 to 28 cores.Therefore we conclude varying the number of cores and threads may impact the performance of the model nonlinearly.

Conclusion

During the course of performing this lab we encountered and overcome a variety of issues related to the CNTK library. These stemmed both from a lack of high quality documentation provided by developers as well as CNTK being officially depreciated and abandoned by the developers, and thus requires outdated versions of certain libraries. Due to these difficulties it took us far longer than initially expected to get CNTK functioning, with our first successful model trained on the evening of October 24th. After having a successful implementation, we were able to quickly move through the lab's requirements. Unfortunately we did not have enough real time remaining to fully train the models with a sufficient number of epochs to achieve a meaningful accuracy. This has led us to focus our performance comparisons on the samples/second metric as we do not have meaningful data to compare the accuracy levels of the different models.

Appendix A - Running Alexnet on **CPU** 100k epoch size (Partial run due to time constraint, we estimate it would take ~10-12 hours to achieve a reasonable error rate)

While we unfortunately were unable to fully execute the model to its maximum accuracy due to time constraints, we were able to run it for a few hours and observe it beginning to lower the training error.

```
(local2) -bash-4.2$ python AlexNet_ImageNet_Distributed.py --epoch_size 100000
--num_epochs 10 --minibatch_size 100
-------------------------------------------------------------------
Build info:
            Built time: Apr 23 2019 21:28:38
            Last modified date: Tue Apr 23 21:05:53 2019
            Build type: release
            Build target: CPU-only
            With ASGD: yes
            Math lib: mkl
            Build Branch: HEAD
            Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
            MPI distribution: Open MPI
            MPI version: 1.10.7
-------------------------------------------------------------------
Selected CPU as the process wide default device.
Training 72357384 parameters in 16 parameter tensors.
 Minibatch[   1- 200]: loss = 5.243869 * 20000, metric = 99.02% * 20000;
 Minibatch[ 201- 400]: loss = 5.077543 * 20000, metric = 98.19% * 20000;
 Minibatch[ 401- 600]: loss = 4.985050 * 20000, metric = 97.61% * 20000;
 Minibatch[ 601- 800]: loss = 4.822816 * 20000, metric = 96.39% * 20000;
 Minibatch[ 801-1000]: loss = 4.666764 * 20000, metric = 94.52% * 20000;
Finished Epoch[1 of 10]: [Training] loss = 4.959208 * 100000, metric = 97.14% *
100000 2120.191s ( 47.2 samples/s);
 Minibatch[   1- 200]: loss = 4.484765 * 20000, metric = 92.78% * 20000;
 Minibatch[ 201- 400]: loss = 4.382358 * 20000, metric = 91.62% * 20000;
 Minibatch[ 401- 600]: loss = 4.240827 * 20000, metric = 90.17% * 20000;
 Minibatch[ 601- 800]: loss = 4.150714 * 20000, metric = 88.55% * 20000;
 Minibatch[ 801-1000]: loss = 4.053595 * 20000, metric = 87.72% * 20000;
Finished Epoch[2 of 10]: [Training] loss = 4.262452 * 100000, metric = 90.17% *
100000 2109.727s ( 47.4 samples/s);
 Minibatch[   1- 200]: loss = 3.932158 * 20000, metric = 85.23% * 20000;
 Minibatch[ 201- 400]: loss = 3.871258 * 20000, metric = 85.00% * 20000;
 Minibatch[ 401- 600]: loss = 3.841963 * 20000, metric = 83.86% * 20000;
 Minibatch[ 601- 800]: loss = 3.843797 * 20000, metric = 84.04% * 20000;
 Minibatch[ 801-1000]: loss = 3.764016 * 20000, metric = 82.56% * 20000;
```

```
Finished Epoch[3 of 10]: [Training] loss = 3.850638 * 100000, metric = 84.14% *
100000 2130.439s ( 46.9 samples/s);
```

Appendix B - Running Alexnet on **CPU** with 10k epoch size with 28 cores

```
(local2) -bash-4.2$ python AlexNet_ImageNet_Distributed.py --epoch_size
10000 --num_epochs 10 --minibatch_size 100
-------------------------------------------------------------------
Build info:
           Built time: Apr 23 2019 21:28:38
           Last modified date: Tue Apr 23 21:05:53 2019
           Build type: release
           Build target: CPU-only
           With ASGD: yes
           Math lib: mkl
           Build Branch: HEAD
           Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
           MPI distribution: Open MPI
           MPI version: 1.10.7
-------------------------------------------------------------------
Selected CPU as the process wide default device.
Training 25171464 parameters in 16 parameter tensors.

Finished Epoch[1 of 10]: [Training] loss = 5.270346 * 10000, metric =
99.17% * 10000 52.744s (189.6 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.162813 * 10000, metric =
99.06% * 10000 34.948s (286.1 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.132723 * 10000, metric =
98.61% * 10000 36.148s (276.6 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.119643 * 10000, metric =
98.89% * 10000 37.848s (264.2 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.114359 * 10000, metric =
98.82% * 10000 37.025s (270.1 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.101731 * 10000, metric =
98.74% * 10000 37.159s (269.1 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 5.106674 * 10000, metric =
98.58% * 10000 39.248s (254.8 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 5.106536 * 10000, metric =
98.68% * 10000 38.121s (262.3 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 5.101554 * 10000, metric =
98.93% * 10000 37.162s (269.1 samples/s);
Finished Evaluation [1]: Minibatch[1-100]: metric = 98.50% * 10000
```

Appendix C - Running Alexnet on **CPU** with 10k epoch size 14 core

```
(local2) -bash-4.2$ python AlexNet_ImageNet_Distributed.py --epoch_size
10000 --num_epochs 10 --minibatch_size 100
-------------------------------------------------------------------
Build info:

           Built time: Apr 23 2019 21:28:38
           Last modified date: Tue Apr 23 21:05:53 2019
           Build type: release
           Build target: CPU-only
           With ASGD: yes
           Math lib: mkl
           Build Branch: HEAD
           Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
           MPI distribution: Open MPI
           MPI version: 1.10.7
-------------------------------------------------------------------
Selected CPU as the process wide default device.
Training 25171464 parameters in 16 parameter tensors.
Finished Epoch[1 of 10]: [Training] loss = 5.305413 * 10000, metric =
99.49% * 10000 53.507s (186.9 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.268695 * 10000, metric =
99.17% * 10000 48.360s (206.8 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.159349 * 10000, metric =
98.80% * 10000 49.472s (202.1 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.116936 * 10000, metric =
98.69% * 10000 50.406s (198.4 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.087937 * 10000, metric =
98.53% * 10000 51.754s (193.2 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.039977 * 10000, metric =
98.06% * 10000 50.370s (198.5 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 4.984264 * 10000, metric =
97.90% * 10000 49.220s (203.2 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 4.950775 * 10000, metric =
97.29% * 10000 48.498s (206.2 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 4.889199 * 10000, metric =
96.95% * 10000 50.568s (197.8 samples/s);
Finished Epoch[10 of 10]: [Training] loss = 4.841125 * 10000, metric =
96.50% * 10000 49.909s (200.4 samples/s);
Finished Evaluation [1]: Minibatch[1-100]: metric = 96.82% * 10000;
```

Appendix D - Running Resnet-34 on **CPU** 14 cores epoch size of 1000

```
(local2) -bash-4.2$ python TrainResNet_ImageNet_Distributed.py --network
resnet34 --epochs 10 --epoch_size 1000
-------------------------------------------------------------------
Build info:

            Built time: Apr 23 2019 21:28:38
            Last modified date: Tue Apr 23 21:05:53 2019
            Build type: release
            Build target: CPU-only
            With ASGD: yes
            Math lib: mkl
            Build Branch: HEAD
            Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
            MPI distribution: Open MPI
            MPI version: 1.10.7
-------------------------------------------------------------------
Selected CPU as the process wide default device.
Finished Epoch[1 of 10]: [Training] loss = 5.566160 * 1024, metric = 98.63% *
1024 71.197s ( 14.4 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.488571 * 992, metric = 98.79% *
992 62.310s ( 15.9 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.375075 * 992, metric = 99.29% *
992 60.181s ( 16.5 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.305439 * 992, metric = 99.50% *
992 58.590s ( 16.9 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.264518 * 1024, metric = 99.41% *
1024 66.801s ( 15.3 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.258305 * 992, metric = 98.89% *
992 58.997s ( 16.8 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 5.230970 * 992, metric = 98.69% *
992 69.591s ( 14.3 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 5.193313 * 992, metric = 98.39% *
992 58.958s ( 16.8 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 5.220408 * 1024, metric = 98.83% *
1024 61.116s ( 16.8 samples/s);
Finished Epoch[10 of 10]: [Training] loss = 5.197086 * 976, metric = 98.57% *
976 58.203s ( 16.8 samples/s);
Finished Evaluation [1]: Minibatch[1-313]: metric = 98.76% * 10000;
```

Appendix E - Running Alexnet on **GPU** with 10k epoch size, 256 batch size

```
(local-cntk-gpu) [kshafie@o0719 HiDL]$ python AlexNet_ImageNet_Distributed.py
--epoch_size 10000 --num_epochs 10
-------------------------------------------------------------------
Build info:

            Built time: Apr 23 2019 21:29:24
            Last modified date: Tue Apr 23 21:05:52 2019
            Build type: release
            Build target: GPU
            With ASGD: yes
            Math lib: mkl
            CUDA version: 10.0.0
            CUDNN version: 7.6.4
            Build Branch: HEAD
            Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
            MPI distribution: Open MPI
            MPI version: 1.10.7
-------------------------------------------------------------------
Selected GPU[0] Tesla P100-PCIE-16GB as the process wide default device.
Training 25171464 parameters in 16 parameter tensors.

Finished Epoch[1 of 10]: [Training] loss = 5.301822 * 10240, metric = 99.35% *
10240 85.245s (120.1 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.298998 * 9984, metric = 99.40% *
9984 38.774s (257.5 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.292076 * 9984, metric = 99.13% *
9984 26.740s (373.4 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.220207 * 9984, metric = 98.91% *
9984 22.021s (453.4 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.168812 * 9984, metric = 98.97% *
9984 21.569s (462.9 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.134045 * 9984, metric = 98.99% *
9984 21.144s (472.2 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 5.102593 * 9984, metric = 98.64% *
9984 21.055s (474.2 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 5.072197 * 9984, metric = 98.47% *
9984 20.521s (486.5 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 5.047056 * 9984, metric = 98.16% *
9984 19.907s (501.5 samples/s);
Finished Epoch[10 of 10]: [Training] loss = 5.020217 * 9888, metric = 98.22% *
9888 20.158s (490.5 samples/s);
Finished Evaluation [1]: Minibatch[1-40]: metric = 97.92% * 10000;
```

Appendix F - Running Alexnet on **GPU** with 10k epoch size, 100 batch size

```
python AlexNet_ImageNet_Distributed.py --epoch_size 10000 --num_epochs 10
--minibatch_size 100
-------------------------------------------------------------------
Build info:

            Built time: Apr 23 2019 21:29:24
            Last modified date: Tue Apr 23 21:05:52 2019
            Build type: release
            Build target: GPU
            With ASGD: yes
            Math lib: mkl
            CUDA version: 10.0.0
            CUDNN version: 7.6.4
            Build Branch: HEAD
            Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
            MPI distribution: Open MPI
            MPI version: 1.10.7
-------------------------------------------------------------------
Selected GPU[0] Tesla P100-PCIE-16GB as the process wide default device.
Training 25171464 parameters in 16 parameter tensors.

Finished Epoch[1 of 10]: [Training] loss = 5.305516 * 10000, metric = 99.40% *
10000 20.124s (496.9 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.261480 * 10000, metric = 99.27% *
10000 17.155s (582.9 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.159215 * 10000, metric = 98.98% *
10000 17.162s (582.7 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.129863 * 10000, metric = 98.61% *
10000 17.562s (569.4 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.090939 * 10000, metric = 98.60% *
10000 18.316s (546.0 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.059207 * 10000, metric = 98.23% *
10000 17.738s (563.8 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 5.008572 * 10000, metric = 98.08% *
10000 17.426s (573.9 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 4.959070 * 10000, metric = 97.44% *
10000 17.503s (571.3 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 4.923456 * 10000, metric = 97.64% *
10000 18.215s (549.0 samples/s);
Finished Epoch[10 of 10]: [Training] loss = 4.862413 * 10000, metric = 96.66% *
10000 17.720s (564.3 samples/s);
Finished Evaluation [1]: Minibatch[1-100]: metric = 97.16% * 10000;
```

Appendix G - Running Resnet-34 on **GPU**

```
python TrainResNet_ImageNet_Distributed.py --network resnet34 --epochs 10
--epoch_size 1000
-------------------------------------------------------------------
Build info:

            Built time: Apr 23 2019 21:29:24
            Last modified date: Tue Apr 23 21:05:52 2019
            Build type: release
            Build target: GPU
            With ASGD: yes
            Math lib: mkl
            CUDA version: 10.0.0
            CUDNN version: 7.6.4
            Build Branch: HEAD
            Build SHA1: ae9c9c7c5f9e6072cc9c94c254f816dbdc1c5be6
            MPI distribution: Open MPI
            MPI version: 1.10.7
-------------------------------------------------------------------
Selected GPU[0] Tesla P100-PCIE-16GB as the process wide default device.
Finished Epoch[1 of 10]: [Training] loss = 6.217681 * 1024, metric = 98.93% *
1024 15.718s ( 65.1 samples/s);
Finished Epoch[2 of 10]: [Training] loss = 5.751491 * 992, metric = 98.79% *
992 5.123s (193.6 samples/s);
Finished Epoch[3 of 10]: [Training] loss = 5.632725 * 992, metric = 98.59% *
992 5.257s (188.7 samples/s);
Finished Epoch[4 of 10]: [Training] loss = 5.535466 * 992, metric = 99.40% *
992 5.391s (184.0 samples/s);
Finished Epoch[5 of 10]: [Training] loss = 5.450279 * 1024, metric = 99.12% *
1024 5.290s (193.6 samples/s);
Finished Epoch[6 of 10]: [Training] loss = 5.412857 * 992, metric = 99.50% *
992 5.068s (195.7 samples/s);
Finished Epoch[7 of 10]: [Training] loss = 5.367375 * 992, metric = 98.69% *
992 5.136s (193.1 samples/s);
Finished Epoch[8 of 10]: [Training] loss = 5.314719 * 992, metric = 98.39% *
992 5.091s (194.9 samples/s);
Finished Epoch[9 of 10]: [Training] loss = 5.348723 * 1024, metric = 98.63% *
1024 5.348s (191.5 samples/s);
Finished Epoch[10 of 10]: [Training] loss = 5.280454 * 976, metric = 98.98% *
976 5.065s (192.7 samples/s);
Finished Evaluation [1]: Minibatch[1-313]: metric = 98.55% * 10000;
```