Micron's 9300 NVMe™ SSD Brings Performance to Immense Machine Learning Training Datasets

Real-world datasets are huge, vastly overwhelming system memory. NVMe SSDs like our 9300 can help you tame training times.

Overview

Most available machine learning (ML) datasets are small — so small that they will fit completely into a relatively small amount of system memory.

While this is an ideal training scenario (one in which there is sufficient system memory to store the entire training dataset), it may not be what one experiences when training with real-world datasets. They tend to be much, much larger.

Fitting immense, real-world training sets into system memory can be very expensive and, in some cases, physically impossible.

This illustrates:

- System resources taxed by AI/ML applications, including standard benchmark data movement
- How using container resource limits helps accurately model real-world AI/ML workloads with standard benchmarks — focused on dataset-to-system memory ratios and how storage IO changes with changing ratios
- How fast storage ameliorates some of the effects of large training data sets

Finally, we'll show how our 9300 NVMe SSDs enable simultaneous training and ingest.

Fast Facts

Real-world model training relies on parallelism — an optimal system ingests as it trains.

Micron's 9300 NVMe SSDs support 11X to 19X higher ingest rates while model training is running.

Containers help accurately model immense training set behavior using standard benchmarks.

System Resource Exercised by AI/ML Applications

To best architect an AI/ML training platform, one should understand which platform resources are exercised heaviest when running the workloads (benchmarks). This will help with sizing, selection and configuration. We expect some resources to be heavily used (GPUs for instance), while other resource stressing may be unexpected.

CPUs and GPUs: Figure 1 shows GPU and GPU memory average utilization for each of the four MLPerf benchmarks used.

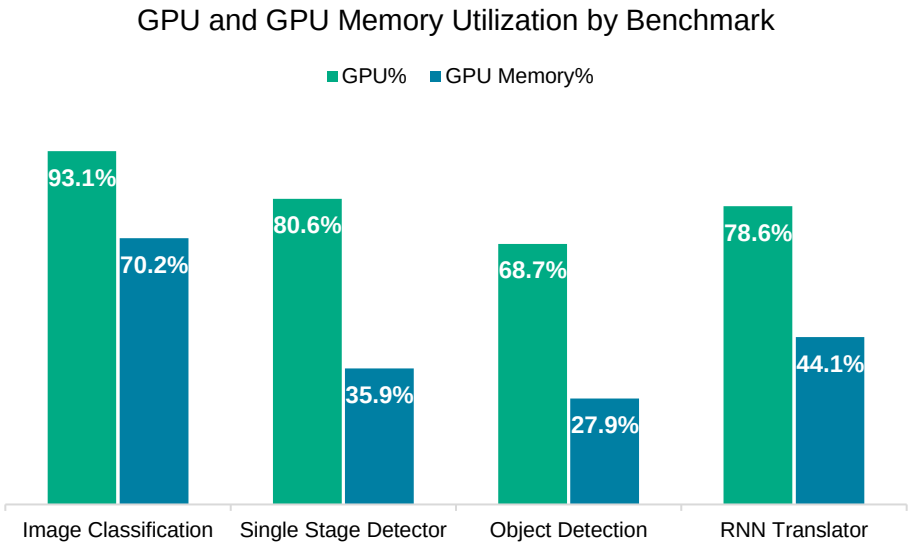


Figure 1: MLPerf Resource Use

The high GPU utilization indicates that the training benchmarks are well optimized, taking advantage of the multiple GPUs in the training platform. The predominantly low and highly variable GPU memory utilization is an artifact of the small data sets used in the MLPerf benchmark. (They do not indicate poor GPU memory optimization or suboptimal batch sizing.) With larger training sets, one would expect to see higher GPU memory utilization (see following sections).

Storage: Figure 2 shows average storage throughput during testing. The average values are low, with a maximum throughput of just over 18 MB/s for image classification. The reason for low storage throughput is directly related to the benchmark's training set size: when training sets are small (ImageNet is 150GB and all the datasets for these benchmarks together occupy less than 400GB), the entire set can be loaded into system memory on the first epoch.

For all subsequent epochs, the training set is accessed from system memory, not storage. This means that we would observe substantial storage throughput for the first epoch and essentially none for all subsequent epochs, lowering the throughput average to what is shown in Figure 2.

Storage utilization is low when the training set fits into system memory. Training data is read from storage only for the first epoch. For all subsequent epochs, it is read from system memory.

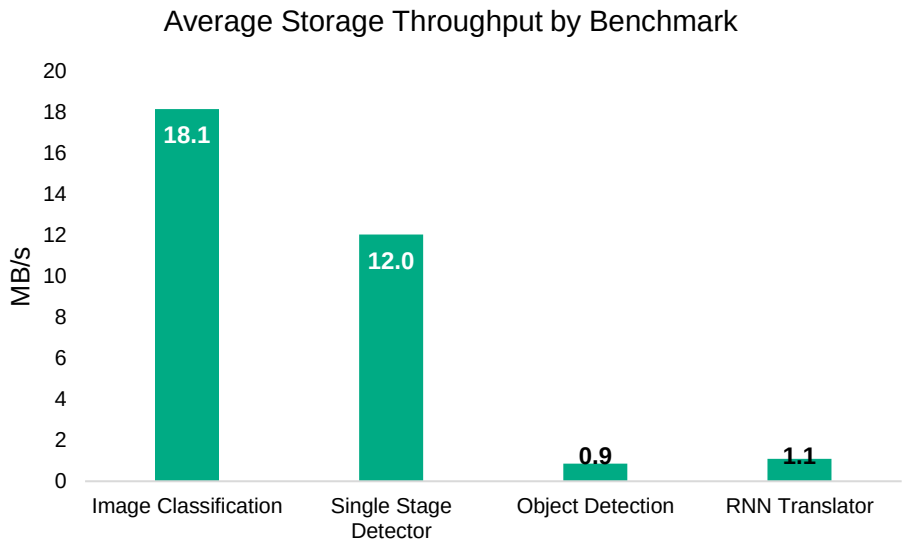


Figure 2: Storage Throughput

Outside of benchmarking, real-world training sets are often very, very large — so large they rarely fit into economical system memory. This is a major difference between benchmarks and real-world dataset training.

CPU (details): Figure 3 is non-normalized, indicating that 32 cores would be loaded 100% and suggests how one might design their training platform.

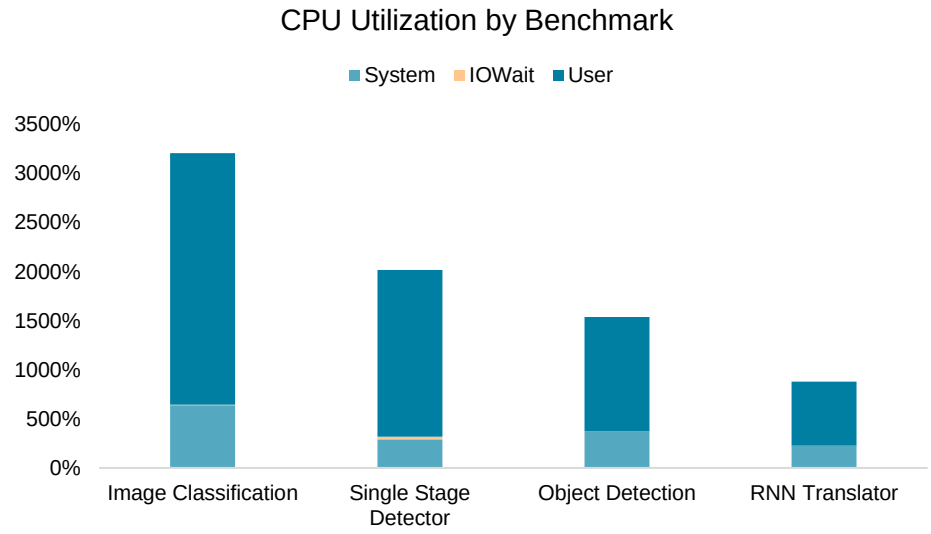


Figure 3: CPU Utilization Details

Accurately Modeling Large Dataset Machine Learning with MLPerf: Containers

Real-world ML datasets rarely fit into available system memory. When they do not, training rates can become painfully slow due to heavy reliance on storage. Since benchmarks use small training datasets, we use container resource limits with MLPerf to model large, real-world datasets.

Benchmark training datasets are very small — so small that they are easily stored in system memory on the first epoch.

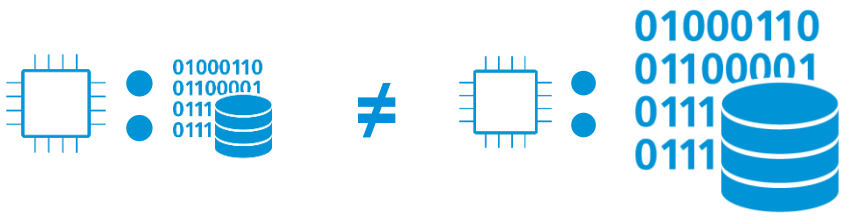
Real-world training data sets are much larger.

System-Memory-to-Dataset-Size: The Ratio Matters

Since most available ML benchmarking datasets are small, they easily fit into a relatively small amount of system memory and are far smaller than common real-world training datasets. (For example, Micron uses some datasets that are several terabytes.)

Containers Align Benchmark Testing to Real-World Use

There is a discontinuity between system-memory-to-training-dataset-size ratio compared to system-memory-to-actual-dataset-size ratio (possibly several orders of magnitude), as seen in Figure 4.



System-memory-to-dataset-size:
Training Benchmarks

System-memory-to-dataset-size:
Actual Deployment

Figure 4: Training- and Real-World-Memory-to-Dataset-Size Discontinuity

To reconcile this difference between benchmark-dataset-size-to-system-memory and real-world-dataset-size-to-system-memory, we used a containers approach.

We limited the amount of memory to which the container had access during training with the MLPerf benchmarks. This enabled us to control the available memory-to-dataset-size ratio and approximate real-world training dataset results using economical memory.

Container resource limits help reconcile benchmark and real-world training set size differences.

Without limiting the memory, the application would read the entire dataset into memory during the first training epoch, then fetch the data from the file system cache instead of doing any reads from disk for all subsequent epochs.

Figure 5 illustrates the container and training set size to system memory ratio concept.

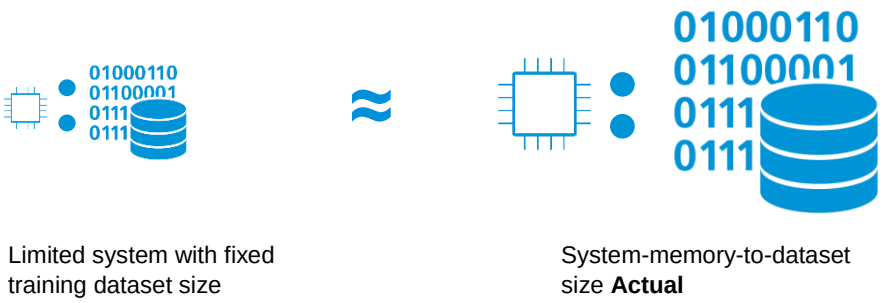


Figure 5: Container-Limited Memory Emulates Real-World-System-Memory-to-Dataset Ratios

Limiting Container Memory Causes More Storage IO

To verify that limiting memory via containers produced expected storage IO, we compared MLPerf results between two configurations differing only in the amount of system memory to which their container had access. Figure 6 shows the results: containers with access to all system memory are shown as "No Memory Limit" while containers with limited memory access are shown as "Memory Limited".

Increasing available memory inversely affects storage utilization.

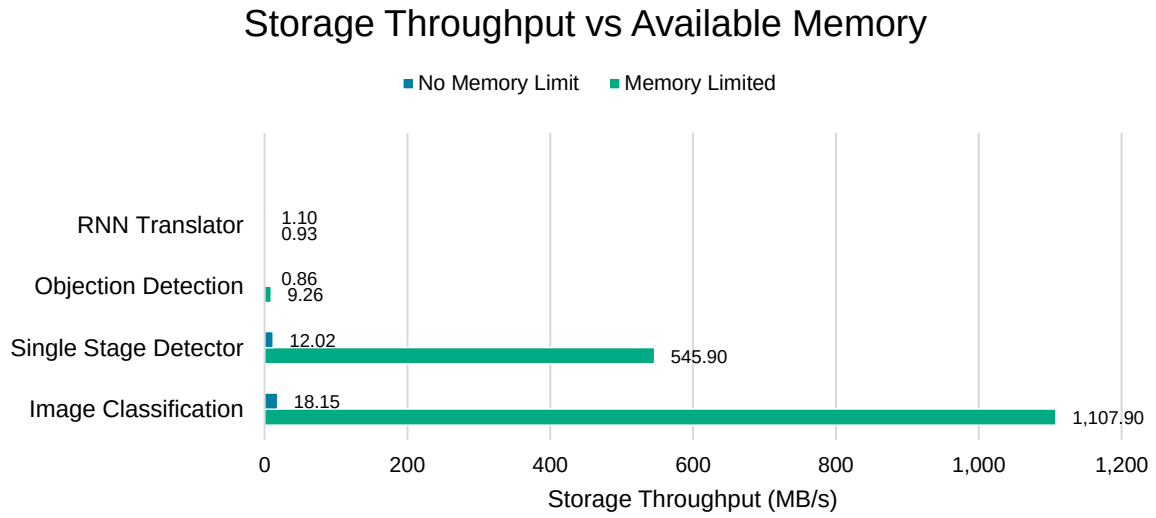


Figure 6: Memory Limiting Results

For example, the Image Classification and Single State Detector benchmarks show significant differences of 61X and 45X higher storage utilization, respectively, when memory is limited. Object Detection showed a less dramatic difference of 11X higher storage utilization with memory limiting. (RNN Translator showed negligible storage utilization in both configurations.)

This aligns to expectation: while the training process is 62 epochs, the unlimited memory configuration read the training set for the first epoch and stored it in (unlimited) system memory. All subsequent epochs read the training set from system memory. With larger datasets, this would not be feasible.

Effects of Adding Fast Storage

We've seen that when the training set fits into system memory, the training set is read from storage once (only for the first epoch), leading to very low storage IO. We've also seen that when we run MLPerf in a memory-limited container to more closely mirror real-world deployments, we see additional storage IO.

How fast of a storage system is needed when the training dataset does not fit into memory?

Isolated Training (no other tasks running): SATA SSDs Are Fast Enough

When running an isolated training benchmark SATA and NVMe SSDs perform very similarly (Figure 7). This is expected as it is a result of the storage throughput in Figure 7. When run in isolation, SATA SSDs can provide sufficient throughput to support an IO-intensive training workload (for example, Image Classification requires 1.2 GB/s).

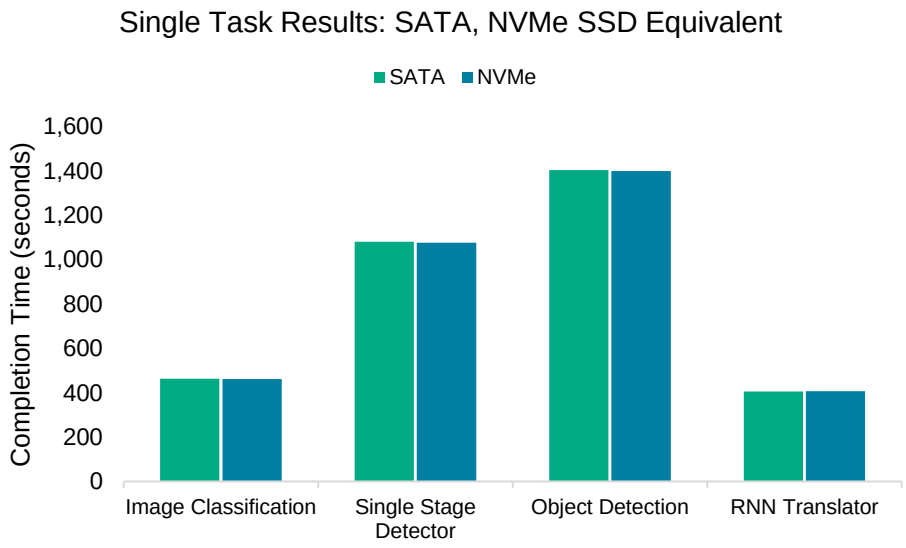


Figure 7: Isolated Training Times

Real-World Training: Parallel Processing is Ideal

The results in Figure 7 may not reflect how training is actually implemented. Real-world training systems are not isolated, and training isn't the only task running (that is, storage is not idle except when reading in the training dataset). Actual deployments are quite complex.

- Training is not a one-pass, then complete process
- Multiple models may need to be trained
- Datasets are very large
- Datasets need to be cached in (copied to) local storage, used, then evicted
- As the dataset is evicted, the next dataset needs to be cached into local storage

Figures 8a and 8b illustrate an important difference between isolated and real-world training.

In Figure 8a (isolated training), the ingest and training processes are serial: Model A training data is ingested, then ingest stops as Model A is trained. Once Model A training completes, Model A training data is discarded, and Model B training data is ingested. Once complete, Model B is trained. This serial process repeats for all models.

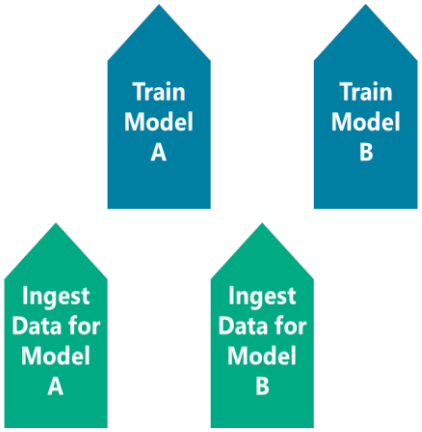


Figure 8a: Isolated Training: Serial

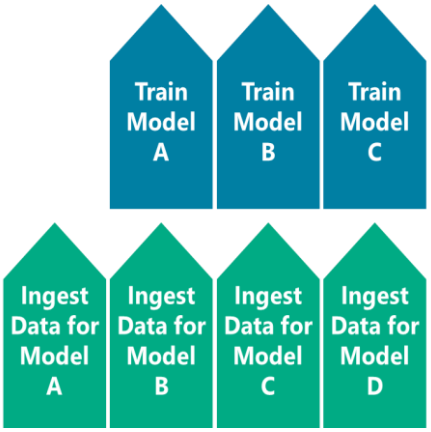


Figure 8b: Real World Training: Parallel

Figure 8b shows actual training processes which are more parallel. In real-world, parallel training we ingest data for training Model A. As Model A is trained, we ***simultaneously*** ingest the training data set for Model B. Train and ingest run at the same time, a parallel process that occurs for all models.

Supporting simultaneous training and ingest requires storage devices that can sustain a significant ingest rate while the training occurs.

Micron 9300 NVMe SSDs: 11X to 19X Higher Ingest During Training

To measure data ingest rates during training for each SSD type, we used Flexible IO (FIO) to generate large block, sequential write IO (128KB transfer size at 32 threads) to the platform's storage while the training workloads ran. We compared the resulting data ingest rate for each SSD type. Figure 9 shows ingest rates by SSD type and MLPerf benchmark. Clearly NVMe SSDs will support the parallel Data Ingest + Model Training process described in Figure 8b.

Figure 9: NVMe Data Ingest During Training Eclipses SATA

The 9300

Figure 10 shows two test configurations: one with limited system memory and Micron's 9300 NVMe SSD (blue) and a second with unlimited access to system memory but without the Micron 9300 (grey).

Similar height vertical bars in Figure 1 indicate that when storage devices are able to deliver sufficient throughput to maintain training performance (blue), one can fully utilize GPUs without investing in enough system memory to house the entire training dataset (costly and potentially impossible). Naturally, results will depend on the software stack in use.

This training time similarity is reinforced by the correlation line which ranges from a low of 0.94 to a high of 1.00. The correlation line data is calculated by dividing 9300 + Memory Limited benchmark results by the No Memory Limit results for each benchmark. (A correlation value of 1.00 indicates no detectable difference in training time between the two configurations, meaning that adding the 9300 ameliorates the effect of reducing system memory.)

Micron's 9300 NVMe SSD mitigates the performance effects of large training data sets on platforms with limited memory.

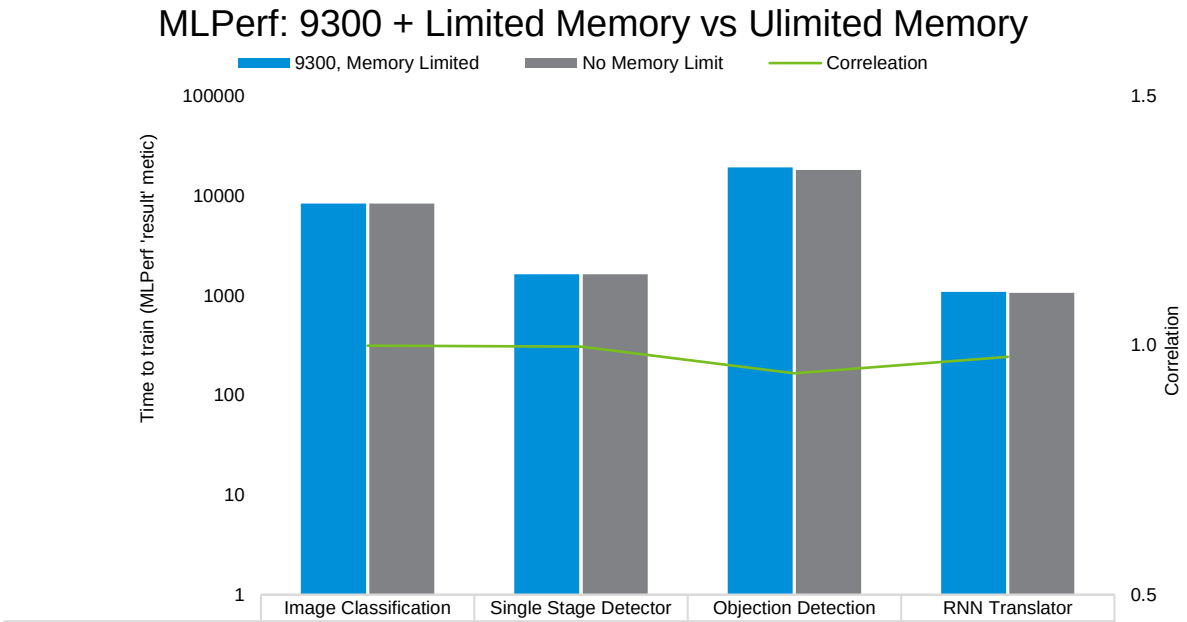


	Image Classification	Single Stage Detector	Objection Detection	RNN Translator
9300, Memory Limited	8366	1630	19217	1083
No Memory Limit	8360	1626	18138	1058
Correleation	1.00	1.00	0.94	0.98

Figure 10: Effect of Micron 9300 on MLPerf results (system memory limited)

Conclusion

ML training benchmarks like MLPerf help ML teams and platform designers glean an understanding of how their configurations will perform. But the disparity between the size of benchmark datasets and those in common practical use may cause unexpected results.

For example, the largest dataset used by the MLPerf benchmarks is the ImageNet dataset (used for the image classification benchmark) — is just 136GB. That is far smaller than training datasets we use at Micron, which are several terabytes in size. Other real-world training datasets may also be quite a bit larger than the datasets used in standard benchmarks.

We can use a containers approach to manage datasets to available memory size rations, effectively modeling large, real-world ML training datasets. We showed that fast NVMe SSD storage like Micron's 9300 can help keep training times quick when dataset size exceeds system memory.

Learn More

For more details on Micron's 9300 NVMe SSDs, visit www.micron.com/9300, and for our latest technology insights, visit www.micron.com/insight. Follow us on twitter @MicronStorage and connect with us on Linkedin

How We Tested

Choosing Platform Components

We used a compute performance very similar to the Nvidia DGX-1. Its main specifications are shown in Table 1:

Hardware and Software Used	
Server	Supermicro SYS-4029GP-TVRT
Storage	8X 7.68TB Micron 9300 PRO NVMe SSDs (RAID-10)
GPU Type	Nvidia V100 GPUs SXM form factor 32GB HBM2 memory per GPU NVLink Cube Mesh interconnect between GPUs
GPU Count	8
CPUs	2X Intel Xeon Platinum 8180M CPUs (28 cores @ 2.5 GHz)
Memory	3TB Micron DRAM (24X 128GB DDR4 2666 MHz DIMMs)
Operating System	Ubuntu 18.04.2 (all available updates installed)
Note	All tests executed with the docker containers defined in the Nvidia submitted code for their v0.5.0 submission.

Correlating Results

The MLPerf benchmark (https://mlperf.org/) is undergoing rapid development. As of this document's publication, MLPerf Training v0.6 posted on 07/10/19 was the most recent available (results from the prior v0.5 benchmark revision were published at the end of 2018). Because of this fluidity, it is imperative to demonstrate that MLPerf results are reproduceable and show strong correlation with results published by others.

In December of 2018, the first benchmark results for MLPerf were submitted by Intel, Google & Nvidia. The results measured the performance of different machine learning algorithms on the submitters' various hardware, using time to train to an accuracy threshold as their metrics.

Micron has been using similar benchmarks in our Austin, Texas, performance engineering lab to help us understand how machine learning training stresses storage resources. To ensure our results were close to existing, published results (and hence validating our test process), we compared our MLPerf results to commercial results noted above.

Figure 11 below demonstrates that our test results (Micron in blue) show strong correlation with results submitted by these commercial entities (Commercial in grey). Correlation values = absolute value (Micron results/Commercial results); a correlation value of 1 indicates complete correlation.

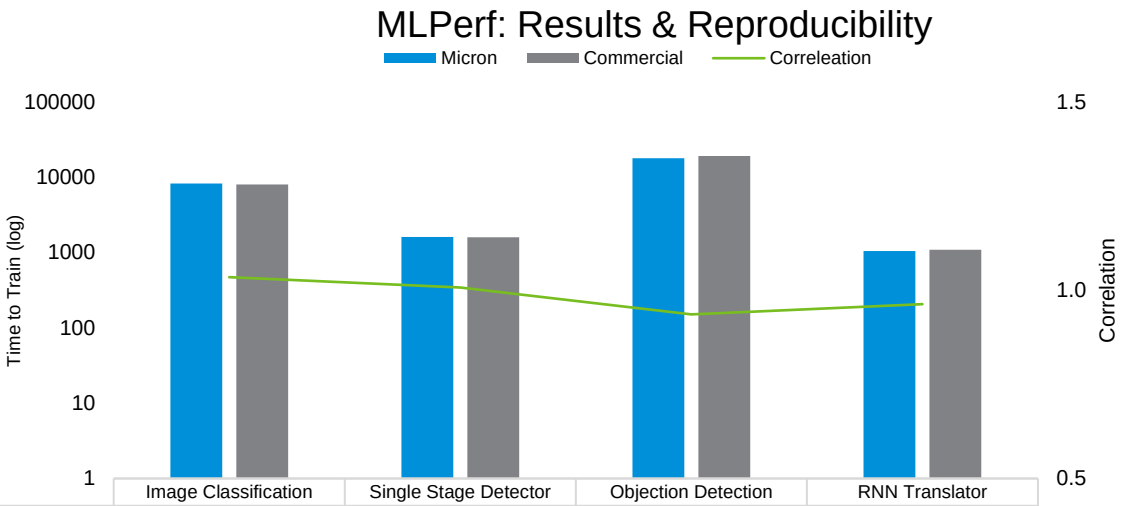


	Image Classification	Single Stage Detector	Objection Detection	RNN Translator
Micron	8360	1626	18138	1058
Commercial	8076	1614	19374	1098
Correleation	1.04	1.01	0.94	0.96

Figure 11: Verifying Reproduceable Results

Correlation values in Figure 11 range from a low of 0.94 to a high of 1.04. This narrow distribution indicates very good correlation results.

micron.com