

# Execution-Time Performance of Deep Learning Networks on CPU, GPU and TPU Runtime Environments

## Summary

A performance review of execution times on Google Colab, for five deep learning network examples, was conducted on CPU, GPU and TPU runtime environments using the MNIST dataset. The networks were 1) a multi-layer perceptron (MLP) network, 2) a convolutional neural network (CNN), 3) a recurrent neural network (RNN), 4) a long short-term memory network (LSTM), and 5) an autoencoder.

## General findings

Training times (Table 1) for all five network exemplars were significantly better on the GPU runtime environment than on Google Colab's CPU environment. Of the networks, the CNN had the greatest performance improvement on GPUs than CPUs only, with a speedup of over 33 times (3332%). This was followed by the LSTM, which had a speedup of over 22 times (2257%), while speed ups for the autoencoder, MLP and RNN were 1464%, 697% and 229% respectively.

Execution time performance for model testing was also significantly better on GPUs than CPUs, for the exemplars. Speedups for the LSTM, CNN, RNN, autoencoder and MLP were 1113%, 915%, 601%, 326%, and 177% respectively.

The TPU runtime environment performed worse than the CPU environment, on training times for the autoencoder, RNN and CNN. Performance time declines were most significant for the autoencoder (-10%). TPU training times were nevertheless significantly better for the LSTM (+9%), and marginally better for the MLP (+1%), than on CPU runtime. All model exemplars performed worse on model evaluation times, on TPUs than on CPUs.

## Discussion

To leverage advantages of using TPUs, optimizations could have been applied to the code used for the performance evaluations. Nevertheless, no customizations were made to the code used, for a head-to-head comparison in the environments. The network code examples were simply run under the three runtime environment options by changing the relevant Colab notebook settings.

## Appendix:

Table 1: Summary of CPU, GPU, TPU Performance

Deep Learning Network	Training Execution Time On:			Testing Execution Time On:			Training Time Speedup As Percentage of Performance on CPU			Testing Time Speedup As Percentage of Performance on CPU		
	CPU	GPU	TPU	CPU	GPU	TPU	CPU	GPU	TPU	CPU	GPU	TPU
MLP	71.622	8.983	70.666	0.929	0.335	0.954	0%	697%	1%	0%	177%	-3%
CNN	1259.83	36.711	1271.57	5.41	0.533	5.626	0%	3332%	-1%	0%	915%	-4%
RNN	125.111	38.01	126.034	6.573	0.938	6.952	0%	229%	-1%	0%	601%	-5%
LSTM	398.724	16.914	364.476	18.504	1.525	21.61	0%	2257%	9%	0%	1113%	-14%
Autoencoder	127.795	8.172	142.383	5.327	1.25	5.935	0%	1464%	-10%	0%	326%	-10%

## Multi-Layer Perceptron (MLP) Example using MNIST Dataset

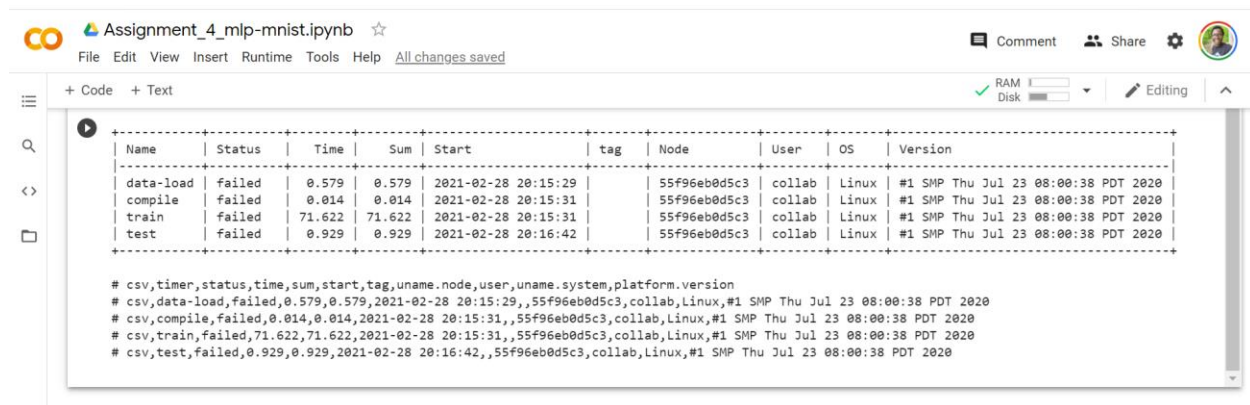


Figure 1: MLP using CPUs only

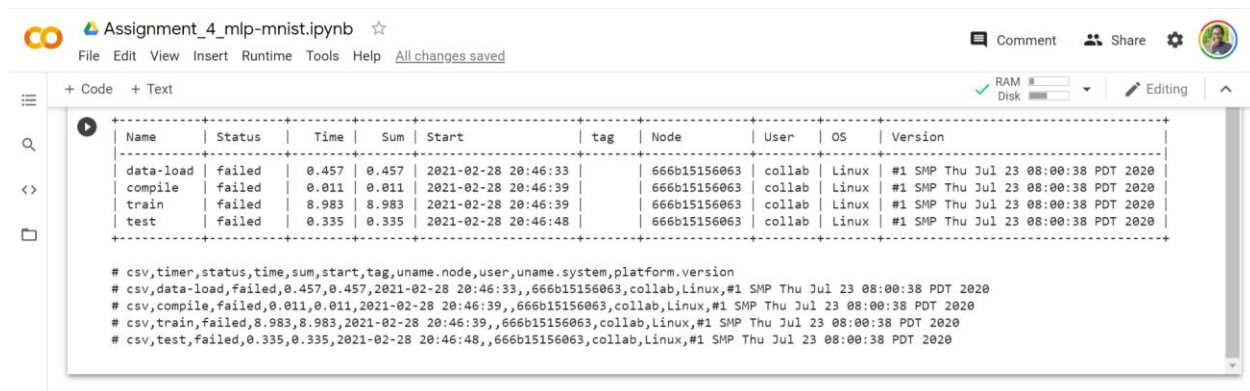


Figure 2: MLP using GPUs

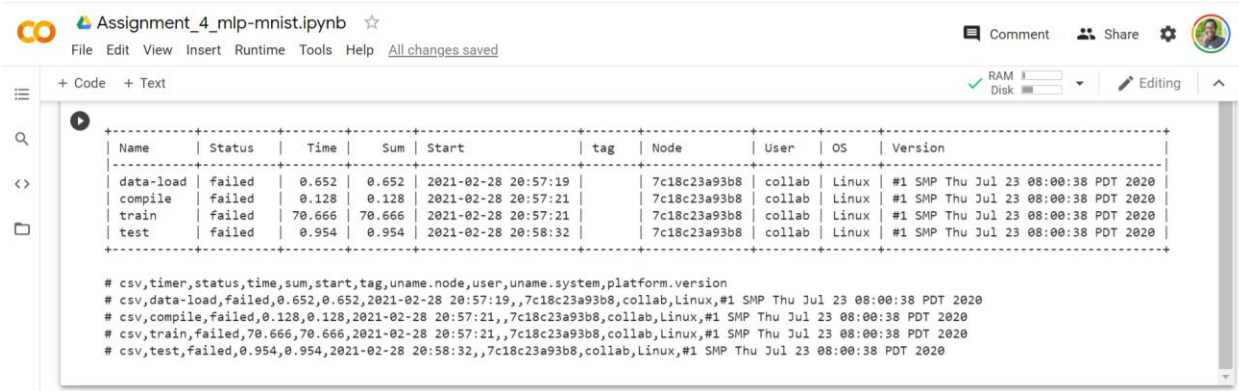


Figure 3: MLP using TPUs

## Convolutional Neural Networks (CNN) Example using MNIST Dataset

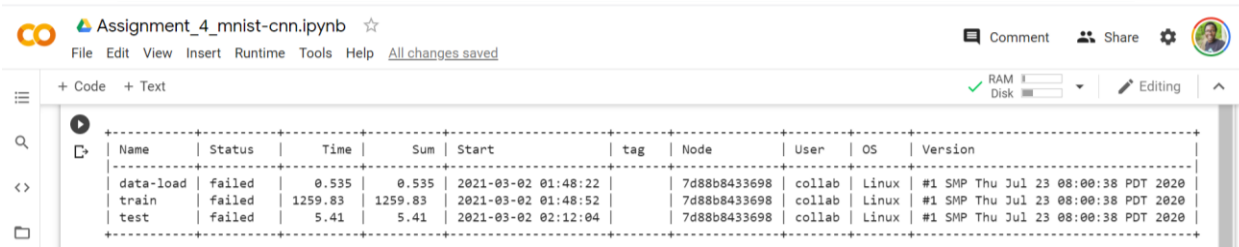


Figure 4: CNN using CPUs only

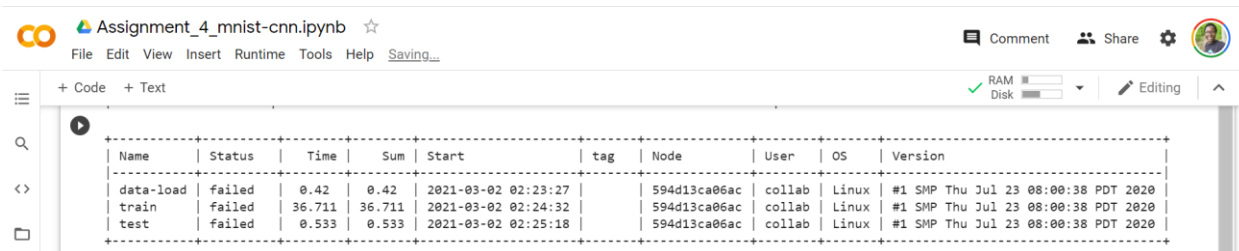


Figure 5: CNN using GPUs

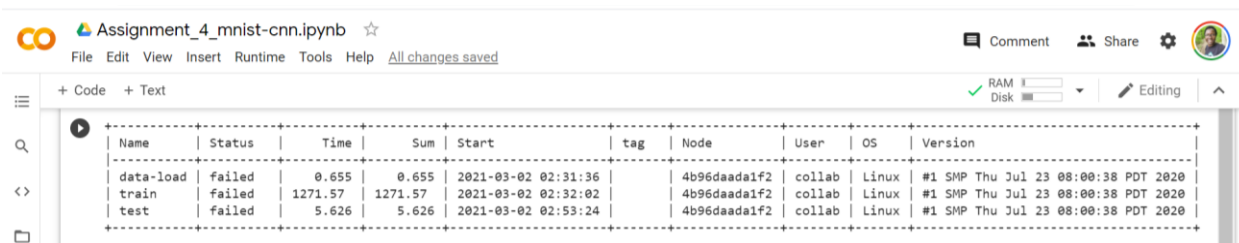
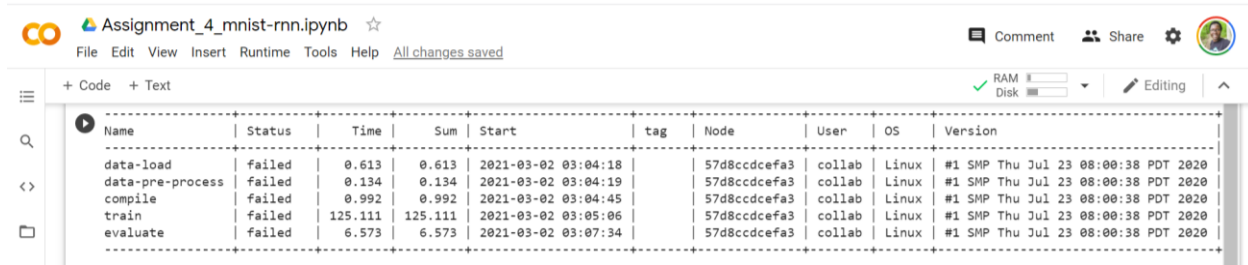


Figure 6: CNN using TPUs

## Recurrent Neural Networks (RNN) Example using MNIST Dataset



Assignment\_4\_mnist-rnn.ipynb

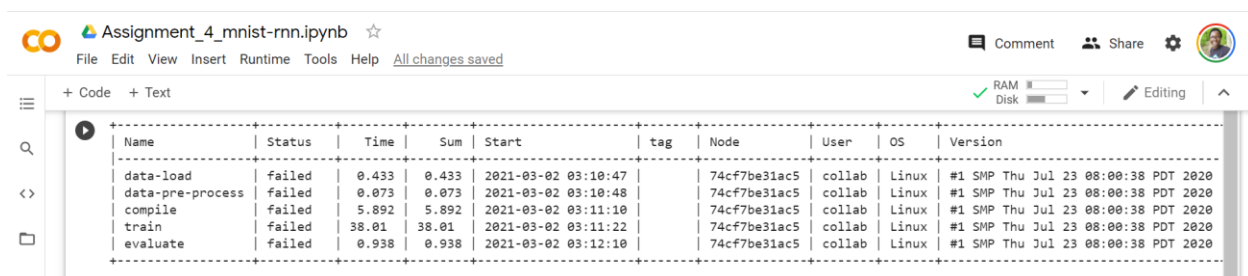
File Edit View Insert Runtime Tools Help All changes saved

RAM Disk

Editing

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.613	0.613	2021-03-02 03:04:18		57d8ccdcfa3	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.134	0.134	2021-03-02 03:04:19		57d8ccdcfa3	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	0.992	0.992	2021-03-02 03:04:45		57d8ccdcfa3	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	125.111	125.111	2021-03-02 03:05:06		57d8ccdcfa3	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	6.573	6.573	2021-03-02 03:07:34		57d8ccdcfa3	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 7: RNN using CPUs only



Assignment\_4\_mnist-rnn.ipynb

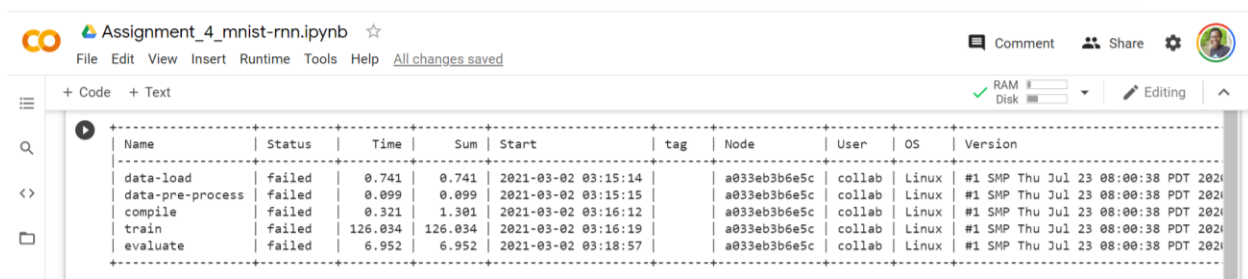
File Edit View Insert Runtime Tools Help All changes saved

RAM Disk

Editing

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.433	0.433	2021-03-02 03:10:47		74cf7be31ac5	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.073	0.073	2021-03-02 03:10:48		74cf7be31ac5	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	5.892	5.892	2021-03-02 03:11:10		74cf7be31ac5	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	38.01	38.01	2021-03-02 03:11:22		74cf7be31ac5	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	0.938	0.938	2021-03-02 03:12:10		74cf7be31ac5	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 8: RNN using GPUs



Assignment\_4\_mnist-rnn.ipynb

File Edit View Insert Runtime Tools Help All changes saved

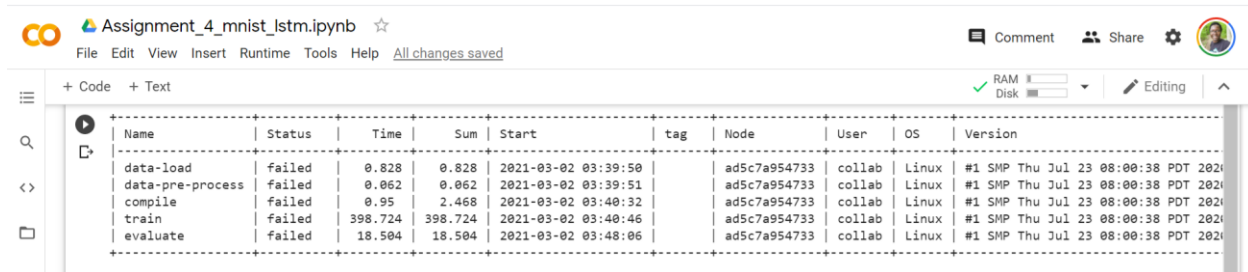
RAM Disk

Editing

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.741	0.741	2021-03-02 03:15:14		a033eb3b6e5c	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.099	0.099	2021-03-02 03:15:15		a033eb3b6e5c	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	0.321	1.301	2021-03-02 03:16:12		a033eb3b6e5c	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	126.034	126.034	2021-03-02 03:16:19		a033eb3b6e5c	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	6.952	6.952	2021-03-02 03:18:57		a033eb3b6e5c	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 9: RNN using TPUs

## Long Short-Term Memory (LSTM) Example using MNIST Dataset



Assignment\_4\_mnist\_lstm.ipynb

File Edit View Insert Runtime Tools Help All changes saved

RAM Disk

Editing

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.828	0.828	2021-03-02 03:39:50		ad5c7a954733	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.062	0.062	2021-03-02 03:39:51		ad5c7a954733	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	0.95	2.468	2021-03-02 03:40:32		ad5c7a954733	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	398.724	398.724	2021-03-02 03:40:46		ad5c7a954733	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	18.504	18.504	2021-03-02 03:48:06		ad5c7a954733	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 10: LSTM using CPUs only

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.561	0.561	2021-03-02 03:51:27		d9d9c12dca02	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.073	0.073	2021-03-02 03:51:28		d9d9c12dca02	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	6.452	6.452	2021-03-02 03:51:38		d9d9c12dca02	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	16.914	16.914	2021-03-02 03:51:52		d9d9c12dca02	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	1.525	1.525	2021-03-02 03:52:36		d9d9c12dca02	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 11: LSTM using GPUs

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
data-load	failed	0.78	0.78	2021-03-02 03:55:54		693d8c3f1239	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
data-pre-process	failed	0.097	0.097	2021-03-02 03:55:55		693d8c3f1239	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
compile	failed	1.517	1.517	2021-03-02 03:56:03		693d8c3f1239	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
train	failed	364.476	364.476	2021-03-02 03:56:11		693d8c3f1239	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	21.61	21.61	2021-03-02 04:04:09		693d8c3f1239	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

Figure 12: LSTM using TPUs

## Autoencoder Example using MNIST Dataset

Name	Status	Time	Sum	Start	tag	Node	User	OS	Version
train	failed	127.795	127.795	2021-03-02 19:56:12		d8c28af0eafb	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020
evaluate	failed	5.327	5.327	2021-03-02 20:05:46		d8c28af0eafb	collab	Linux	#1 SMP Thu Jul 23 08:00:38 PDT 2020

```
# csv,timer,status,time,sum,start,tag,uname.node,user,uname.system,platform.version
# csv,train,failed,127.795,127.795,2021-03-02 19:56:12,,d8c28af0eafb,collab,linux,#1 SMP Thu Jul 23 08:00:38 PDT 2020
# csv,evaluate,failed,5.327,5.327,2021-03-02 20:05:46,,d8c28af0eafb,collab,linux,#1 SMP Thu Jul 23 08:00:38 PDT 2020
```

Figure 13: Autoencoder using CPUs only



Figure 14: Autoencoder using GPUs

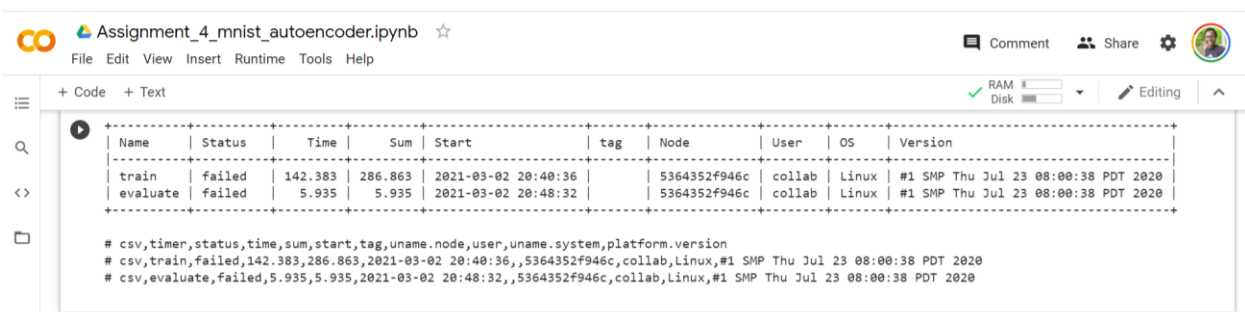


Figure 15: Autoencoder using TPUs