Squeeze more juice out of a single GPU in deep learning

Weiguang Guan, guanw@sharcnet.ca
SHARCNet/Digital Research Alliance of Canada

FAQ

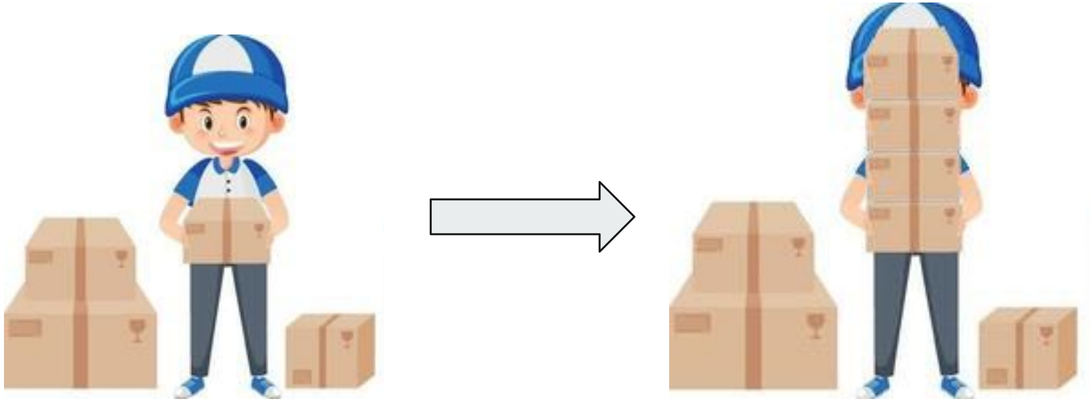
- Is a single GPU sufficient for my training task?
- Do I need to use multiple GPUs
- Is it true that the more GPUs you use, the better?

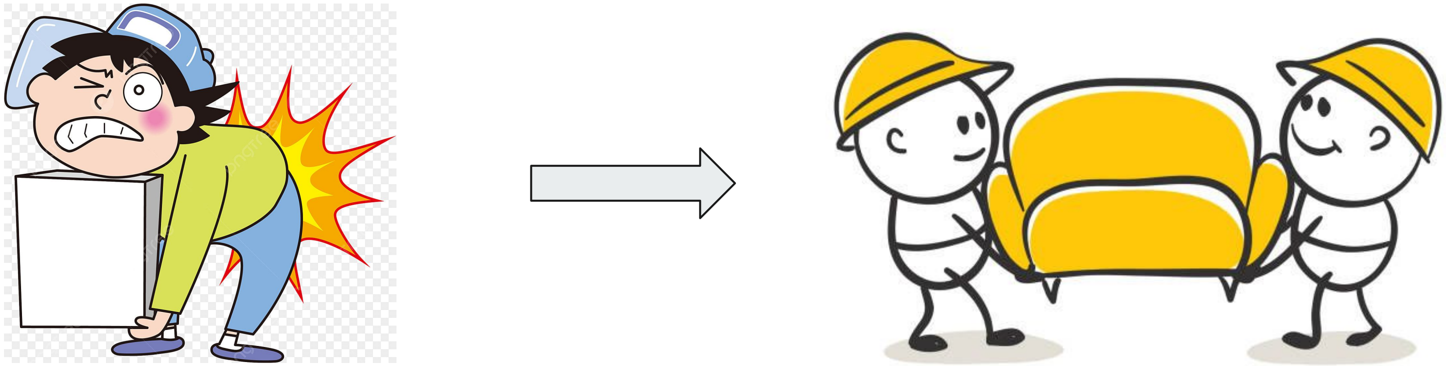
In most cases, single GPU is more than enough!

Choice of using multiple GPUs or a single GPU

Depending on workload

- Size of neural network
- Size of training data
- Capability of GPU





How could I know …

- Comparative method
  - How many GPUs and what GPUs are used in training similar NNs
- Timing tests using
  - Single GPU (T4, V100, V100, A100, …)
  - Multiple GPUs

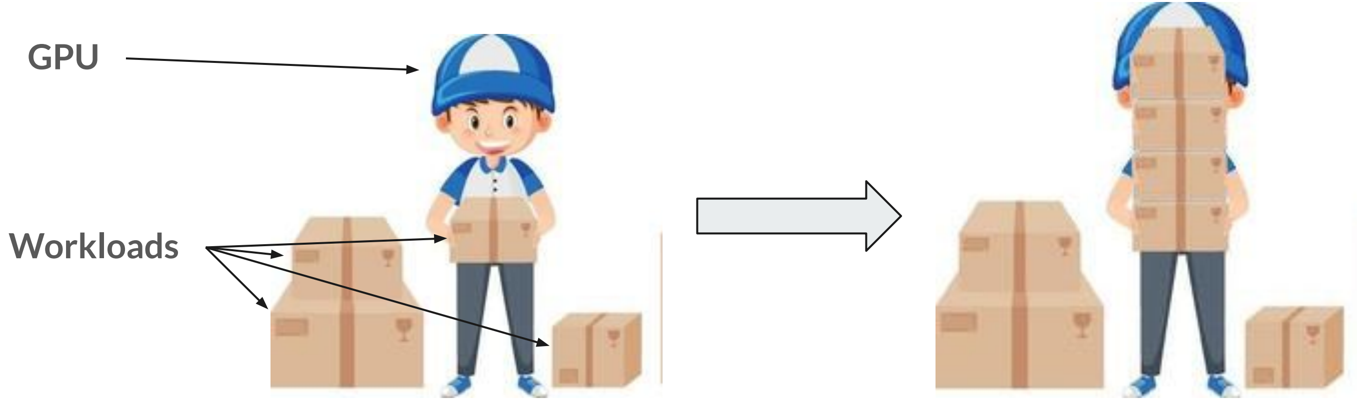
Tip: Use *watch -n1 nvidia-smi* to monitor GPU usage

Reference GPU units

https://docs.alliancecan.ca/wiki/Allocations_and_compute_scheduling

	FP32 score	FP16 score	Memory score	Weighted Score
Weight:	1.6	1.6	0.8	(RGU)
Model				
P100-12gb	0.48	0.00	0.3	1.0
P100-16gb	0.48	0.00	0.4	1.1
T4-16gb	0.42	0.21	0.4	1.3
V100-16gb	0.81	0.40	0.4	2.2
V100-32gb	0.81	0.40	0.8	2.6
A100-40gb	1.00	1.00	1.0	4.0
A100-80gb*	1.00	1.00	2.0	4.8

What can we do if we find a single GPU is under-utilized



Simultaneously run multiple training processes on a single GPU.

NOTE: Usually one needs to run NN training multiple times in order to find optimal hyper-parameters (learning rate, batch size, ...).

Two methods to simultaneously run multiple trainings

- Simply run multiple training processes on a single GPU
- Split a GPU into multiple logical ones and run a training process on each logical GPU.

Physical/logical GPUs

Tensorflow deals with logical GPUs rather physical ones. For example,

with tf.device(logical_gpu) :

- By default, a physical GPU corresponds to a logical GPU
- A single GPU can be split to multiple logical GPUs

Some useful TF functions

- *tf.config.list_physical_devices('GPU')*, which returns a list of physical GPUs
- *tf.config.list_logical_devices('GPU')*, which returns a list of logical GPUs
- *tf.config.set_logical_device_configuration(device, configs_of_logical_devices)*, which splits *device* into multiple logical ones based on *configs_of_logical_devices*.

An example to show the whole process

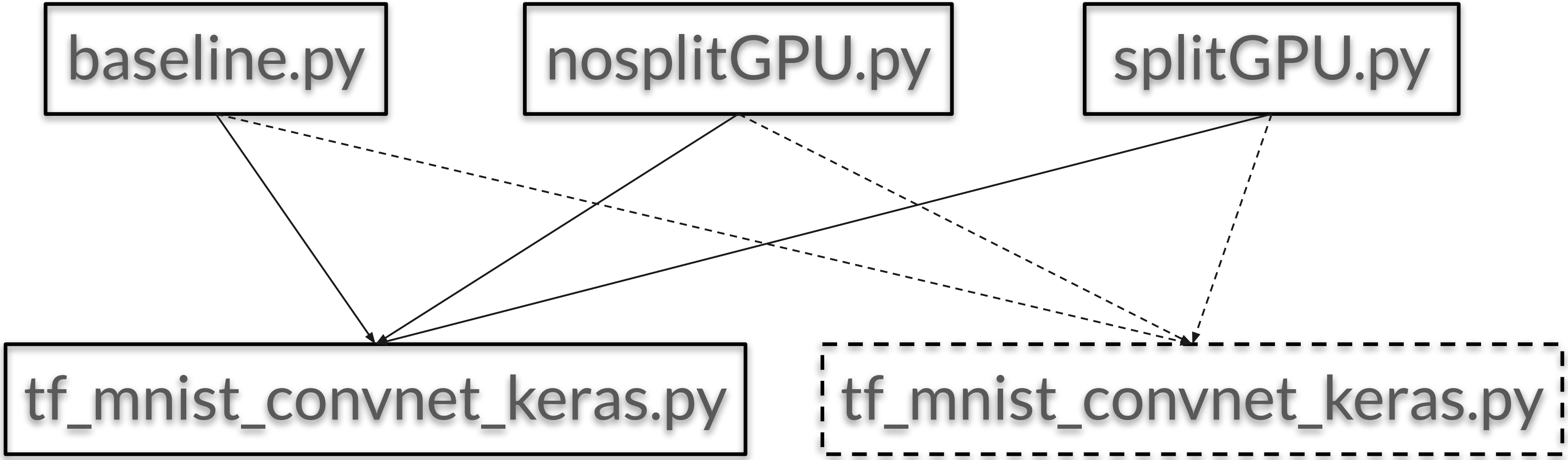
Two NNs:

- A small NN for recognizing handwritten digits
- A medium sized NN: Resnet-50

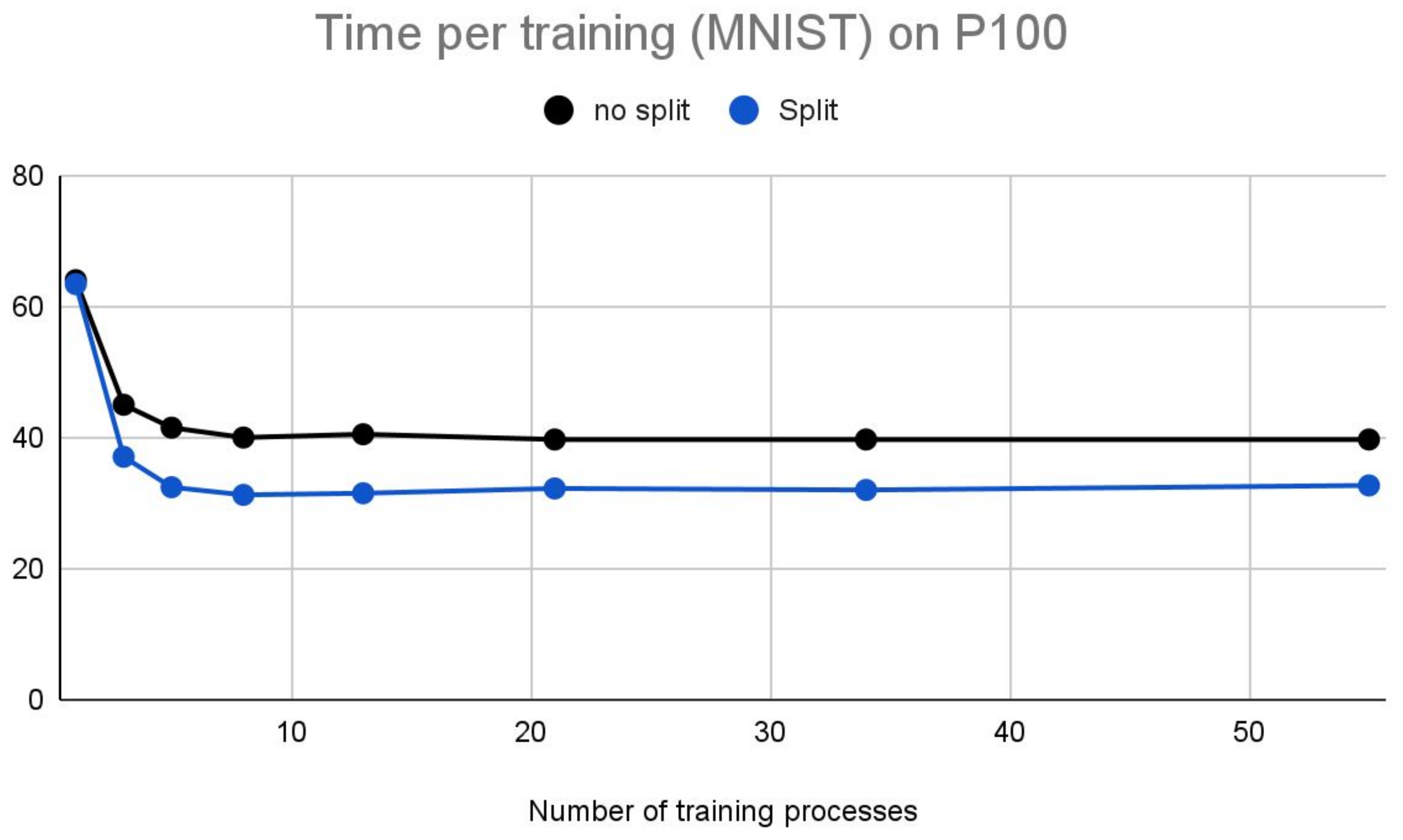
Experiments:

- Run a regular NN on a single GPU as baseline
  - Check the GPU utilization
- Run N training processes in parallel on a single GPU, where N=3, 5, 8, 13, 21, 34, ... with/without splitting it into multiple logical GPUs

Let’s take a look at the code!

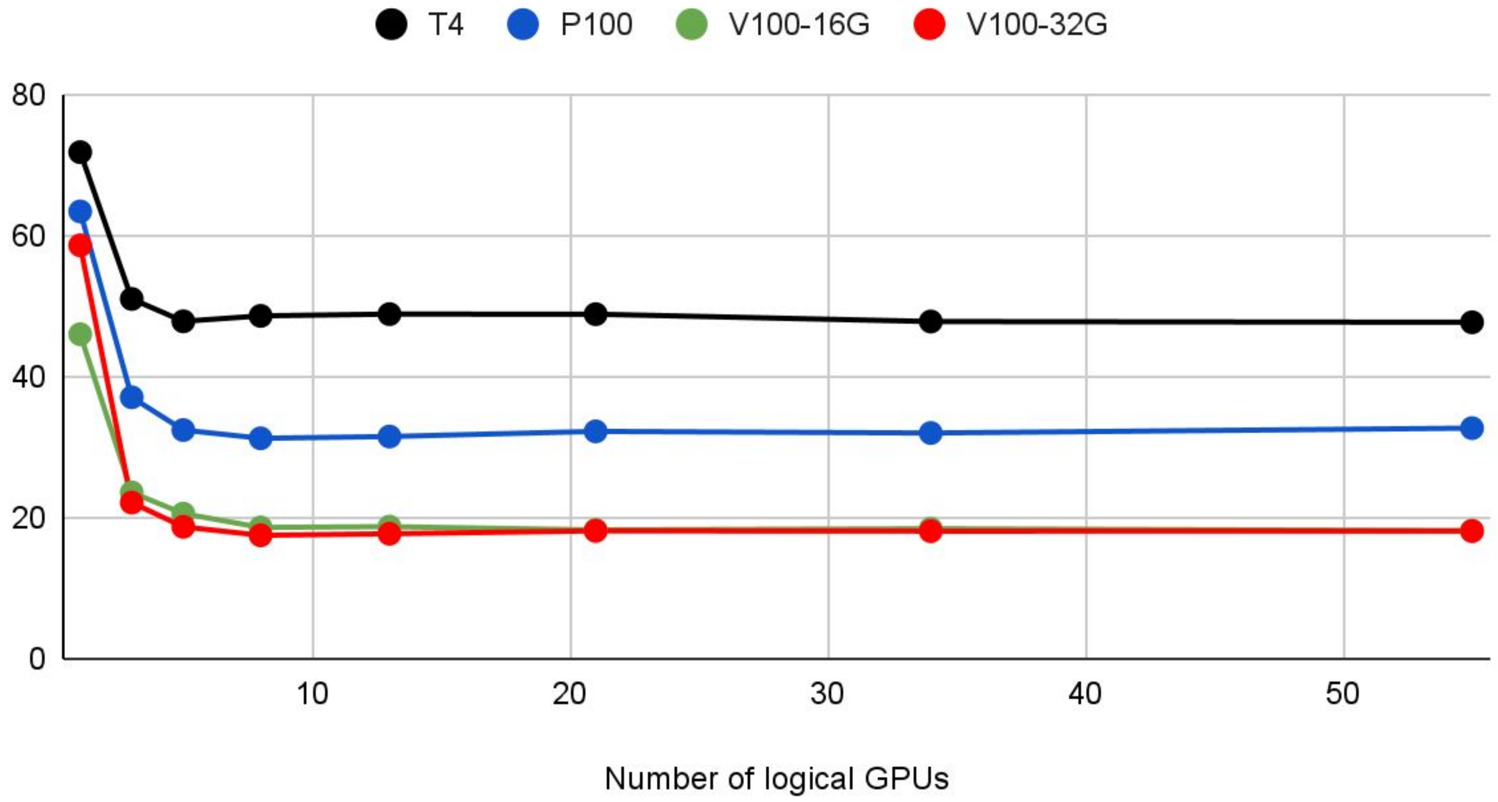


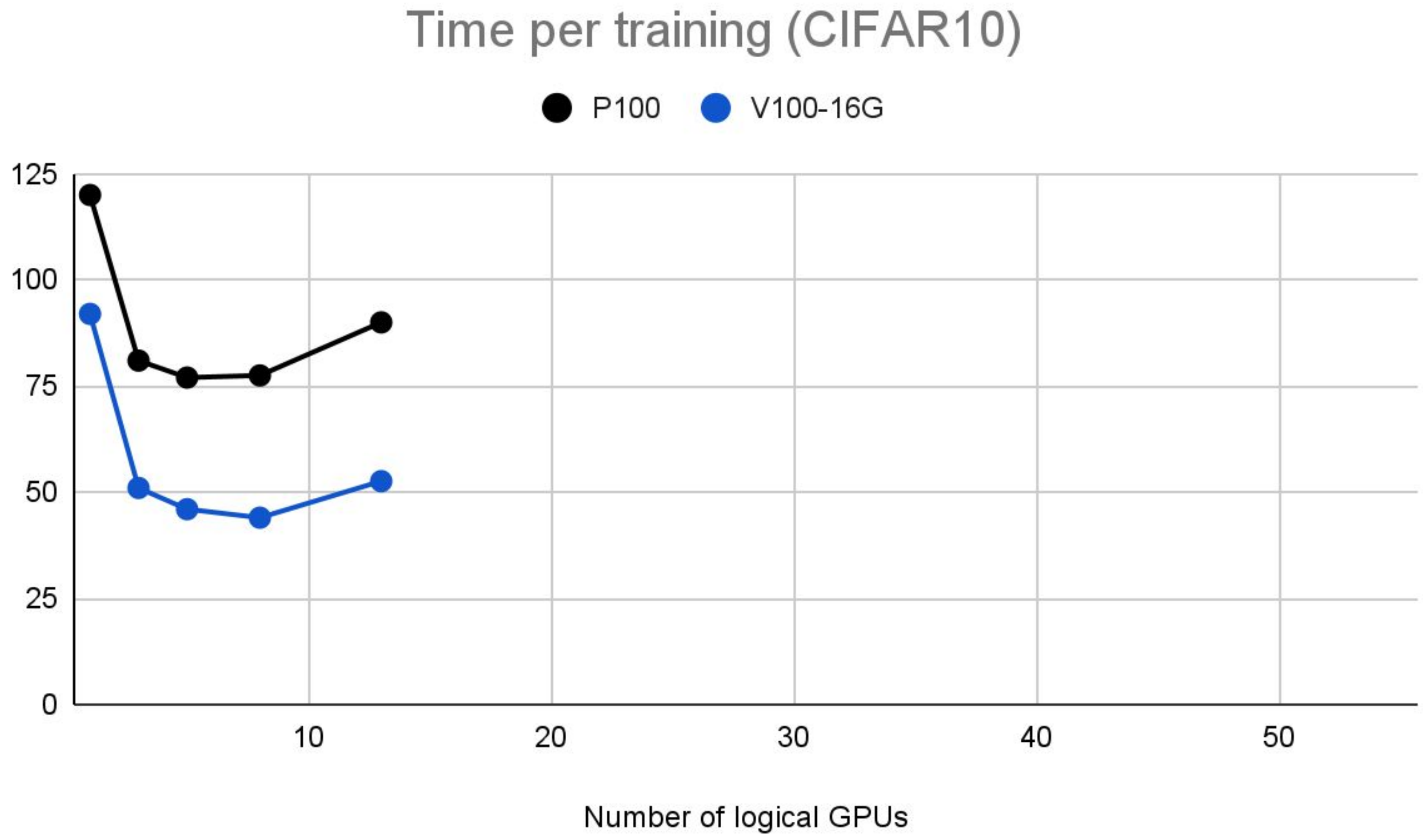
Let’s take a look at the results!



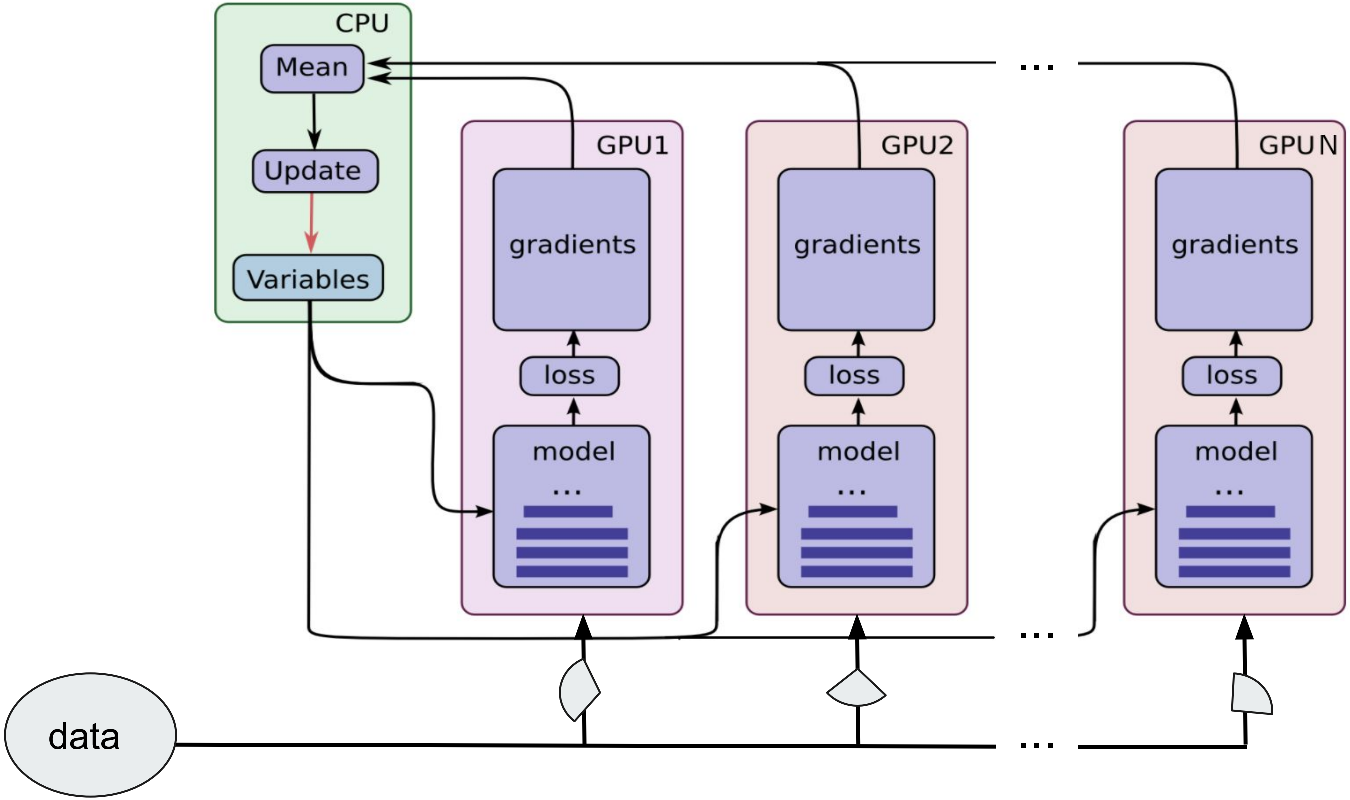
Time per training (MNIST) on P100
no split
Split
80
60
40
20
0
10
20
30
40
50
Number of training processes

Time per training (MNIST)



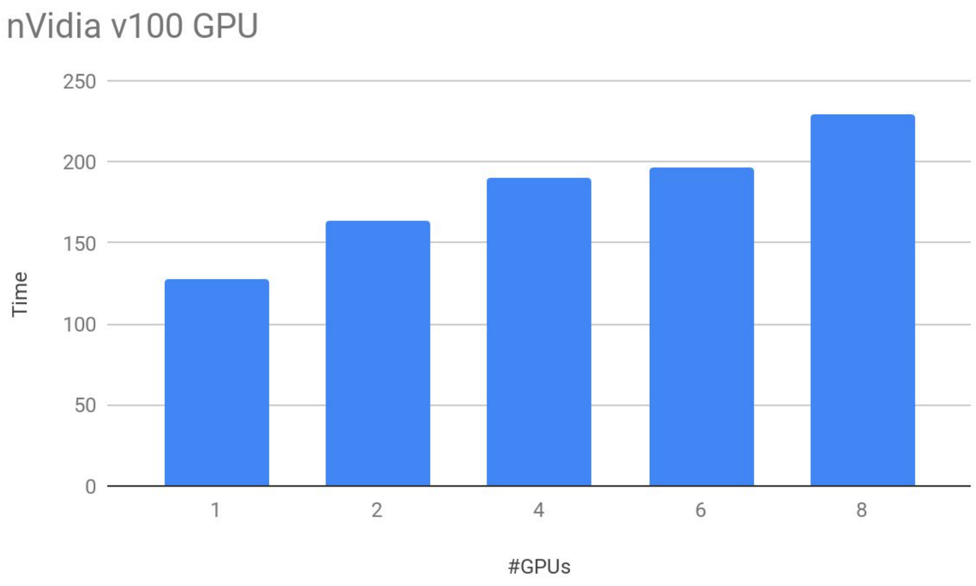

Time per training (CIFAR10)
P100
V100-16G
125
100
75
50
25
0
10
20
30
40
50
Number of logical GPUs

Can Mirrored strategy help?



Mirrored strategy test

Batch size = 256 Iterations = 20000		
# of GPUs	Time (sec)	
	p100	v100
1	135	128
2	161	164
4		190
6		196
8		229



Conclusion

- GPU is under-utilized when used to train small NN. One can find the utilization by command *nvidia-smi* or by testing
- We can get better throughput by simultaneously running multiple training processes on a single GPU
- One needs to find the optimal split of a single GPU to reach maximal throughput by experiment.