compute canada | calcul canada

Using multiple GPUs for Machine Learning

Isaac Ye, HPTC @ York University

Isaac@sharcnet.ca

SHARCNET™

Objectives

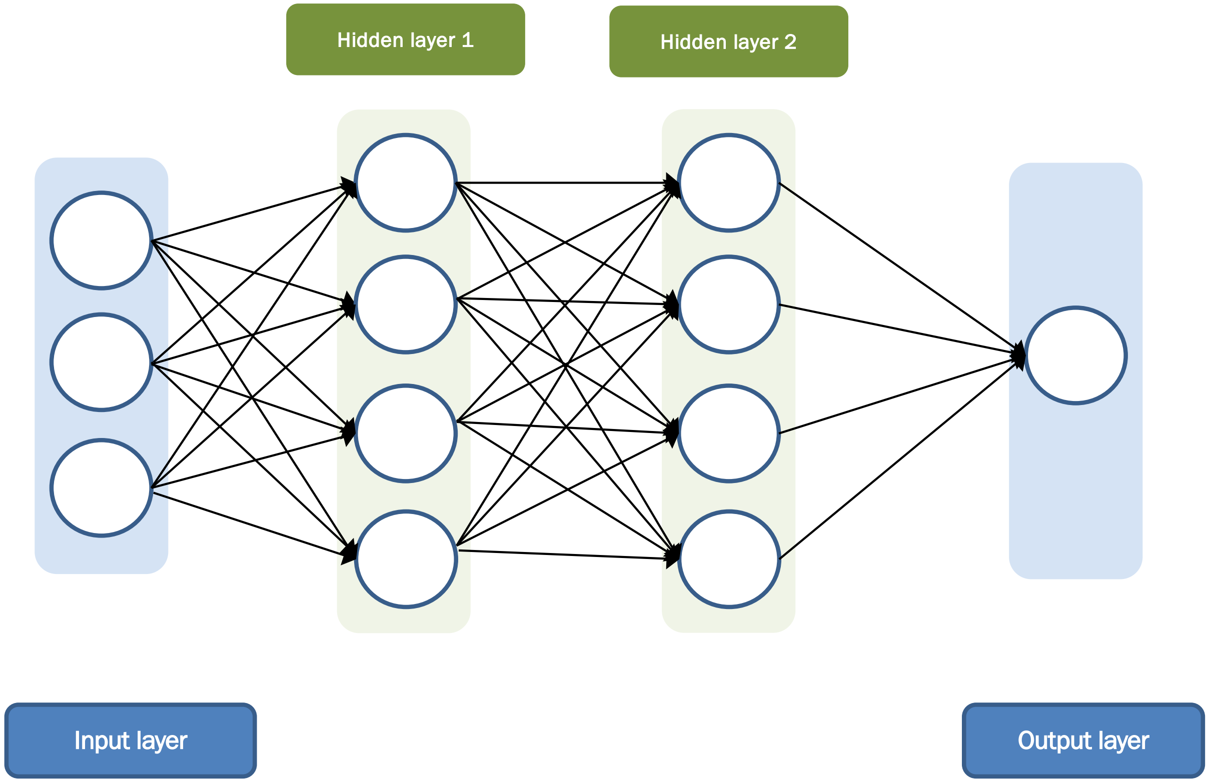
- ✓ Guide a beginner user to run his/her codes using GPU on Graham system
- ✓ Introduce how to setup a job submission script for different ML frameworks (TensorFlow, PyTorch)
- ✓ Introduce several approaches in using multiple GPUs + multiple nodes
- ✓ Show how to use Tensorboard for PyTorch

SHARCNET™
GENERAL INTEREST SEMINAR

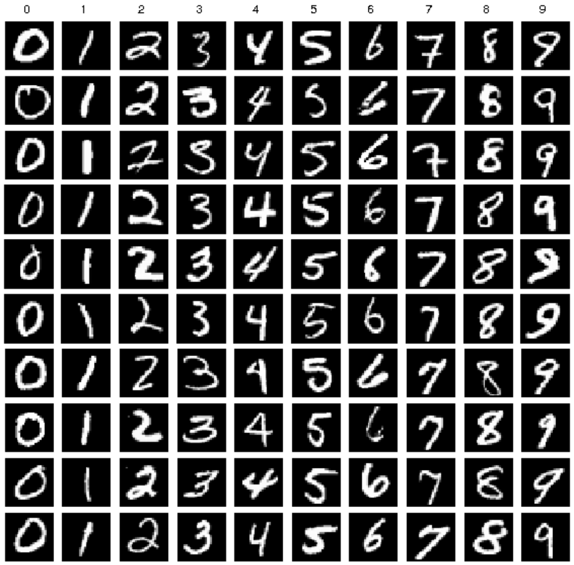
Outline

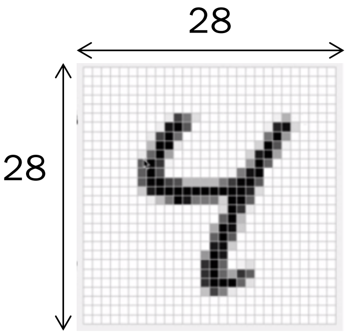
- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

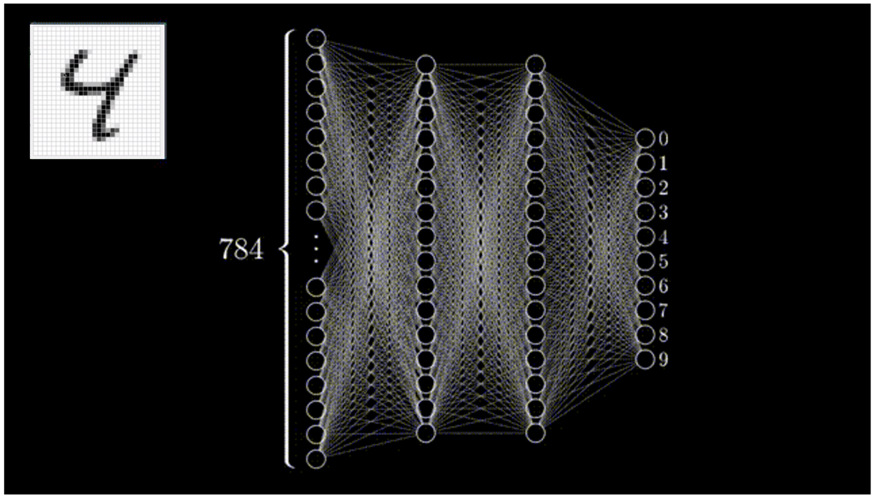
Deep Neural Network



Classification problem: MNIST







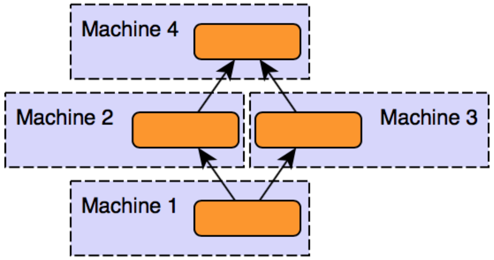
Handwritten data
60K train set and 10K test set
Each image has a size of 28x28 (=784)

Parallelism

Model parallelism

Use the **same data** for every process but **split the model** among processes

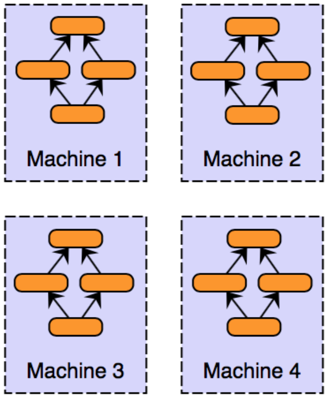
Model Parallelism



Data parallelism

Use the **same model** for every process but feed it with **split data**

Data Parallelism



Outline

- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

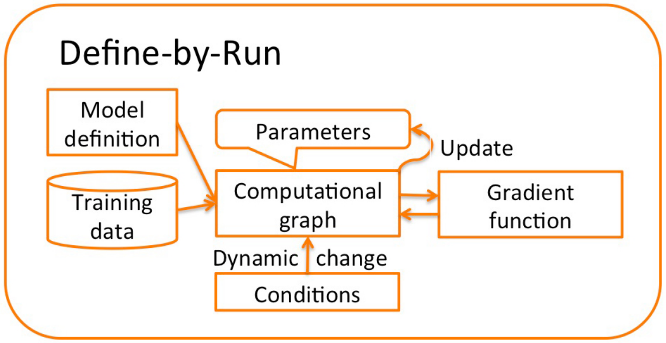
PyTorch

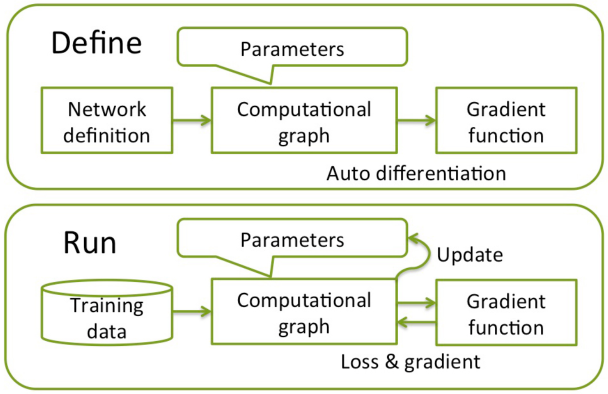
- Rapidly growing in research community developed by Facebook
- A Python adaptation of Torch
- Caffe2 has been merged to PyTorch
- Define-by-Run type for neural networks
- Ease of expression and use
- https://github.com/pytorch/pytorch
- Version 1.7.1 is available in Graham

TensorFlow

- The most widely used framework open-sourced by Google
- Runs on almost all architectures (CPU/GPU/TPU/etc)
- Define-and-Run type for neural networks
- Version 2.0+ has Define-by-Run component (Eager execution)
- https://github.com/tensorflow/tensorflow
- Version 2.3.0 is available in Graham

	PyTorch	TensorFlow
Pros	Easy to use (Python support) Intuitive Dynamic graphs Research community prefers	Large community Heterogeneous architecture TF 2.0: Eager execution(Define-by-Run) Tensorboard (visualizing), Keras
Cons	Small community Less additional tools	Verbose Static graphs





https://www.oreilly.com/content/complex-neural-networks-made-easy-by-chainer/

Outline

- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

GPU available in Graham

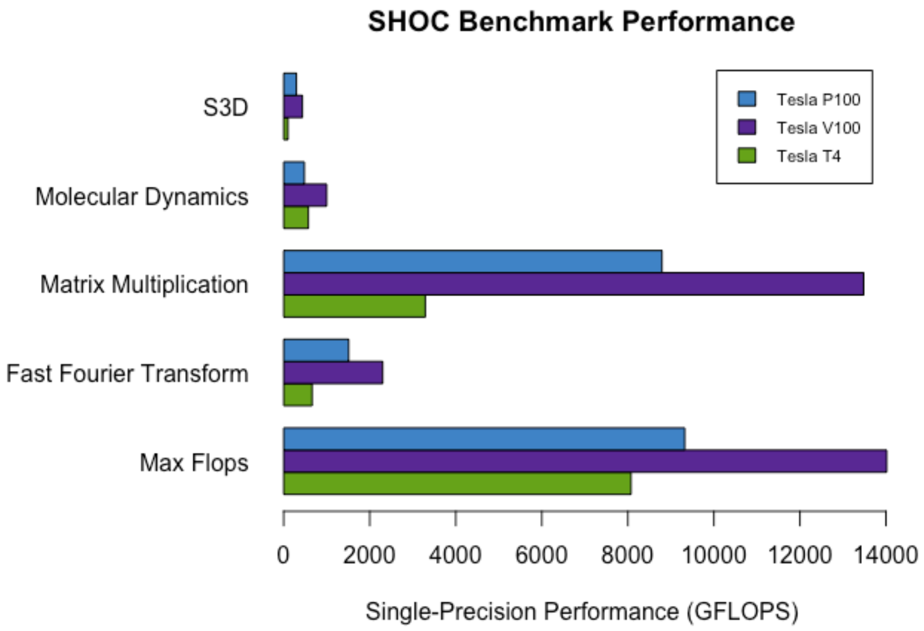
As of Feb, 2021

	# of nodes	# of GPU/node	GPU type	Note
Graham	320	2	P100 Pascal	--gres=gpu:1
	70	8	V100 Volta	CPU/GPU ≤ 3.5 --gres=gpu:v100:1
	144	4	**T4 Turing (DL target)**	CPU/GPU ≤ 3.5 --gres=gpu:t4:1

Which GPUs?

Available GPUs in Graham

	P100	V100	T4
Availability	Best	Good	Better
Double Pre.	5.3 TF	7.8 TF	N/A
Single Pre.	10.6 TF	15.7 TF	8.1 TF
Tensor core	N/A	620	320

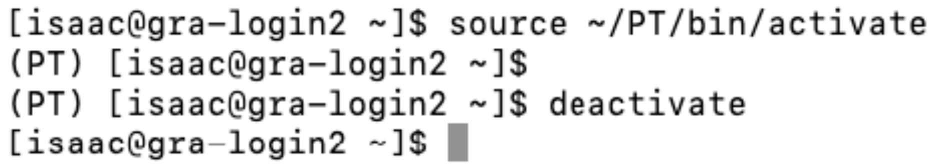


https://www.microway.com/hpc-tech-tips/nvidia-turing-tesla-t4-hpc-performance-benchmarks/tesla_comparison_t4-p100-v100/

Virtual environment

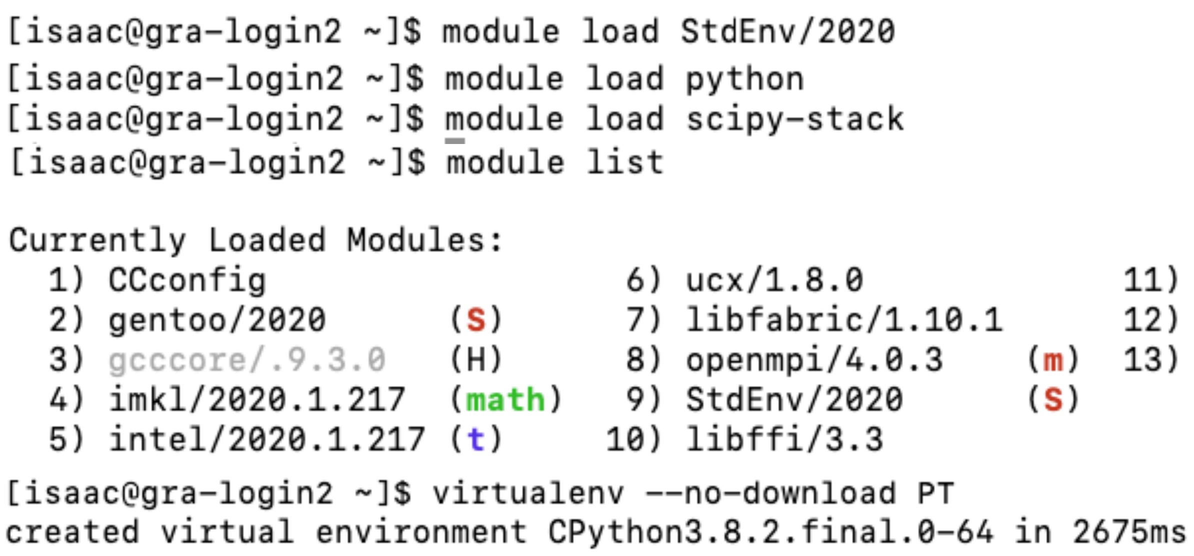
Allows users to create virtual environments so that one can install Python modules easily

Many versions of same module are possible



```
[isaac@gra-login2 ~]$ source ~/PT/bin/activate
(PT) [isaac@gra-login2 ~]$
(PT) [isaac@gra-login2 ~]$ deactivate
[isaac@gra-login2 ~]$ 
```

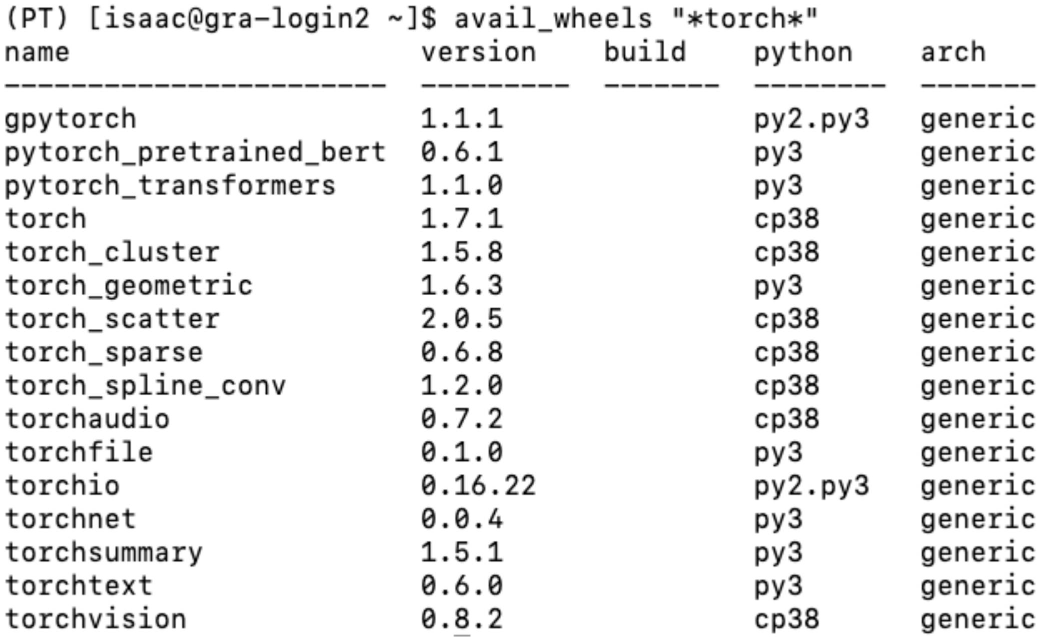
Virtual env. PyTorch



```
[isaac@gra-login2 ~]$ module load StdEnv/2020
[isaac@gra-login2 ~]$ module load python
[isaac@gra-login2 ~]$ module load scipy-stack
[isaac@gra-login2 ~]$ module list

Currently Loaded Modules:
  1) CCconfig                   6) ucx/1.8.0                11) python/3.8.2      (t)
  2) gentoo/2020      (S)       7) libfabric/1.10.1         12) ipykernel/2020b
  3) gcccore/.9.3.0   (H)       8) openmpi/4.0.3      (m)   13) scipy-stack/2020b (math)
  4) imkl/2020.1.217  (math)    9) StdEnv/2020        (S)
  5) intel/2020.1.217 (t)      10) libffi/3.3

[isaac@gra-login2 ~]$ virtualenv --no-download PT
created virtual environment CPython3.8.2.final.0-64 in 2675ms
```

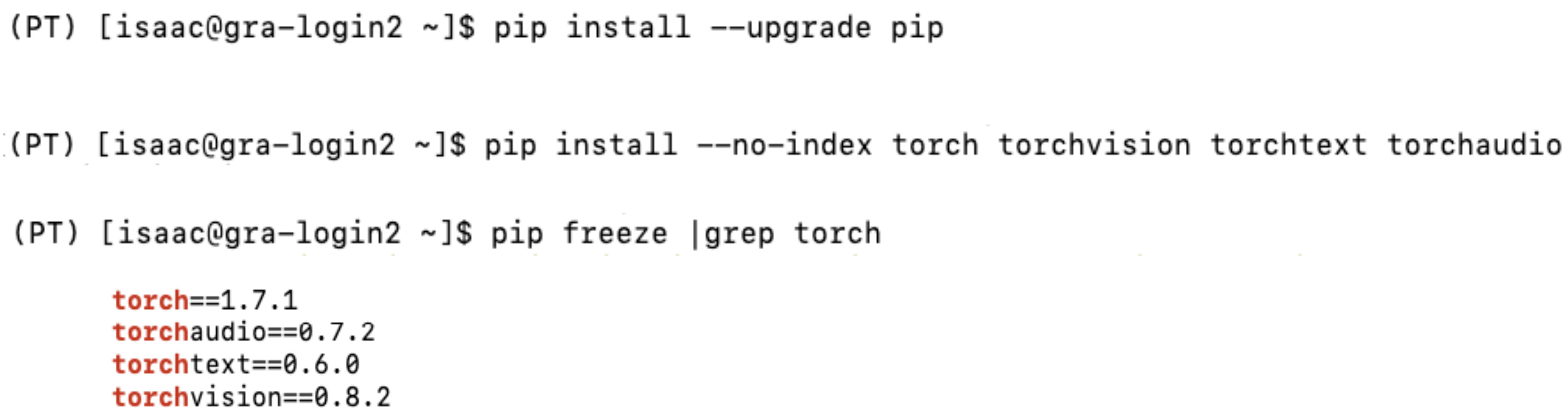
Wheels PyTorch

Graham supports ‘wheels’ for Python package installation.



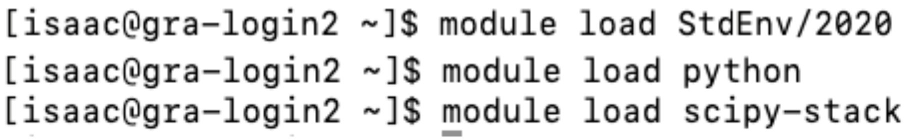
```
(PT) [isaac@gra-login2 ~]$ avail_wheels "*torch*"
name                       version    build    python    arch
-------------------------  ---------  -------  --------  -------
gpytorch                   1.1.1               py2.py3   generic
pytorch_pretrained_bert    0.6.1               py3       generic
pytorch_transformers       1.1.0               py3       generic
torch                      1.7.1               cp38      generic
torch_cluster              1.5.8               cp38      generic
torch_geometric            1.6.3               py3       generic
torch_scatter              2.0.5               cp38      generic
torch_sparse               0.6.8               cp38      generic
torch_spline_conv          1.2.0               cp38      generic
torchaudio                 0.7.2               cp38      generic
torchfile                  0.1.0               py3       generic
torchio                    0.16.22             py2.py3   generic
torchnet                   0.0.4               py3       generic
torchsummary               1.5.1               py3       generic
torchtext                  0.6.0               py3       generic
torchvision                0.8.2               cp38      generic
```

Virtual env. PyTorch

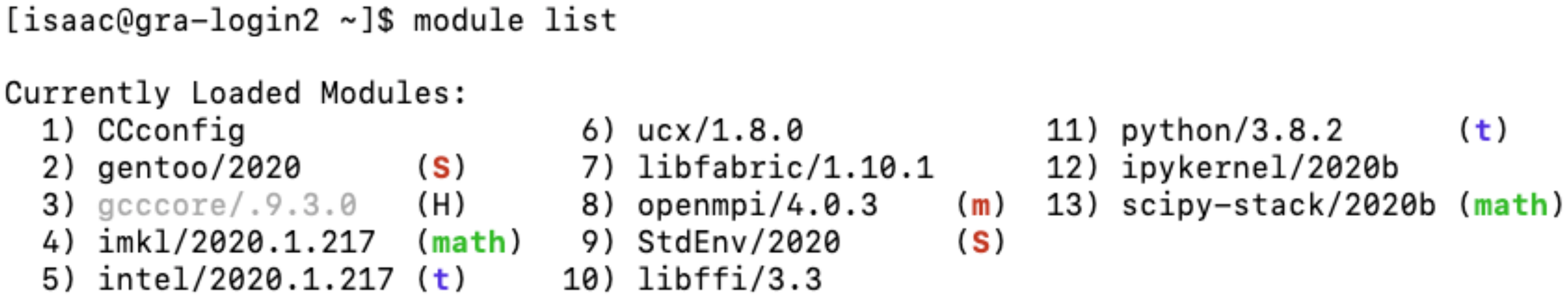


```
(PT) [isaac@gra-login2 ~]$ pip install --upgrade pip


(PT) [isaac@gra-login2 ~]$ pip install --no-index torch torchvision torchtext torchaudio

(PT) [isaac@gra-login2 ~]$ pip freeze |grep torch

      torch==1.7.1
      torchaudio==0.7.2
      torchtext==0.6.0
      torchvision==0.8.2
```

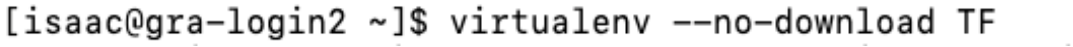
Virtual env. TensorFlow



```
[isaac@gra-login2 ~]$ module load StdEnv/2020
[isaac@gra-login2 ~]$ module load python
[isaac@gra-login2 ~]$ module load scipy-stack
```



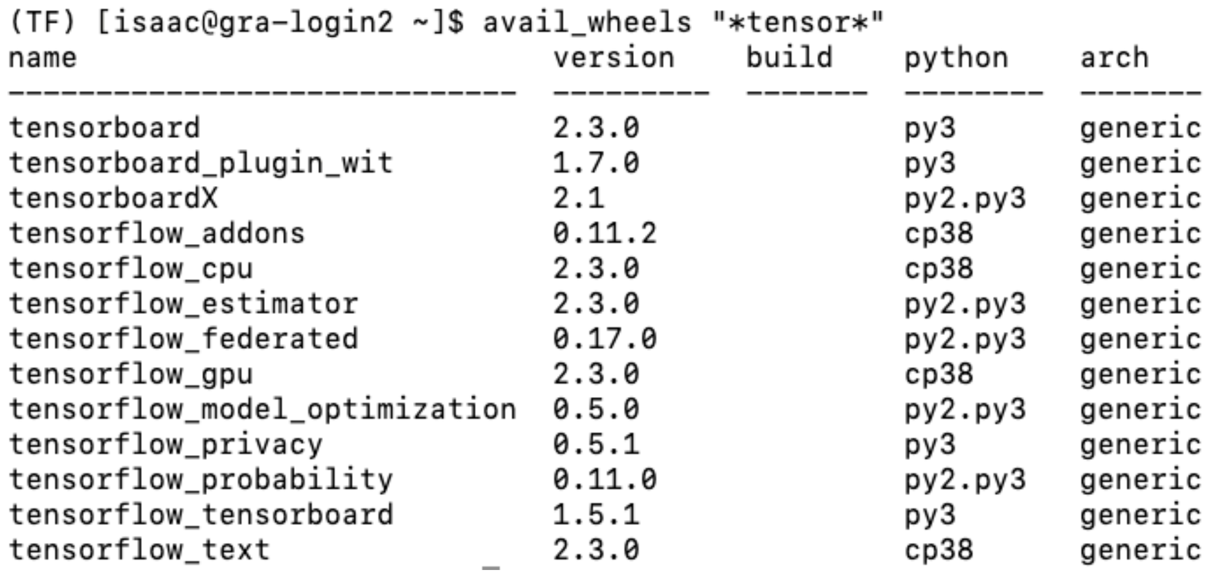
```
[isaac@gra-login2 ~]$ module list

Currently Loaded Modules:
  1) CCconfig                     6) ucx/1.8.0             11) python/3.8.2      (t)
  2) gentoo/2020       (S)        7) libfabric/1.10.1      12) ipykernel/2020b
  3) gcccore/.9.3.0    (H)        8) openmpi/4.0.3    (m)  13) scipy-stack/2020b (math)
  4) imkl/2020.1.217   (math)     9) StdEnv/2020      (S)
  5) intel/2020.1.217  (t)       10) libffi/3.3
```



```
[isaac@gra-login2 ~]$ virtualenv --no-download TF
```

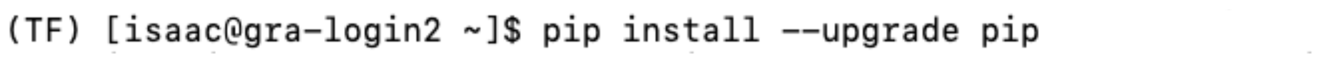
Wheels TensorFlow

Graham supports ‘wheels’ for Python package installation.

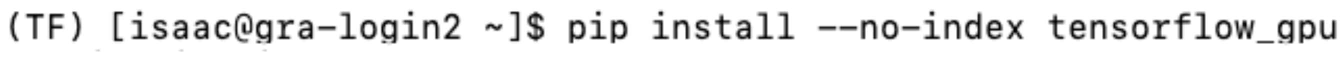


```
(TF) [isaac@gra-login2 ~]$ avail_wheels "*tensor*"
name                           version    build    python    arch
-----------------------------  ---------  -------  --------  -------
tensorboard                    2.3.0               py3       generic
tensorboard_plugin_wit         1.7.0               py3       generic
tensorboardX                   2.1                 py2.py3   generic
tensorflow_addons              0.11.2              cp38      generic
tensorflow_cpu                 2.3.0               cp38      generic
tensorflow_estimator           2.3.0               py2.py3   generic
tensorflow_federated           0.17.0              py2.py3   generic
tensorflow_gpu                 2.3.0               cp38      generic
tensorflow_model_optimization  0.5.0               py2.py3   generic
tensorflow_privacy             0.5.1               py3       generic
tensorflow_probability         0.11.0              py2.py3   generic
tensorflow_tensorboard         1.5.1               py3       generic
tensorflow_text            _   2.3.0               cp38      generic
```

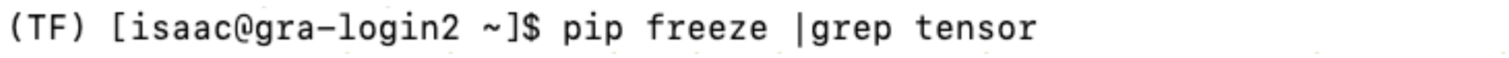
Virtual env. TensorFlow



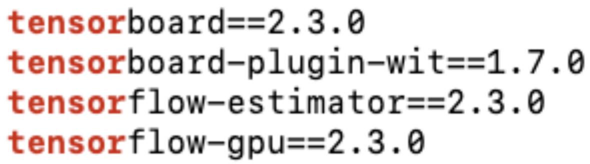
```
(TF) [isaac@gra-login2 ~]$ pip install --upgrade pip
```



```
(TF) [isaac@gra-login2 ~]$ pip install --no-index tensorflow_gpu
```



```
(TF) [isaac@gra-login2 ~]$ pip freeze |grep tensor
```

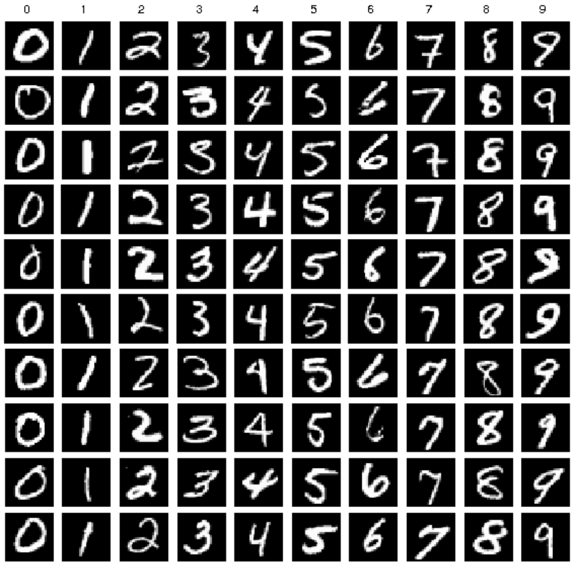


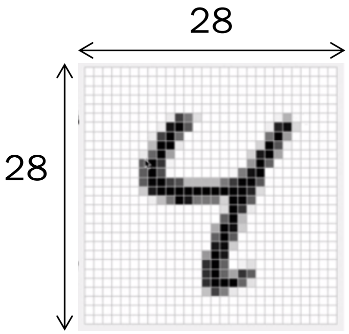
```
tensorboard==2.3.0
tensorboard-plugin-wit==1.7.0
tensorflow-estimator==2.3.0
tensorflow-gpu==2.3.0
```

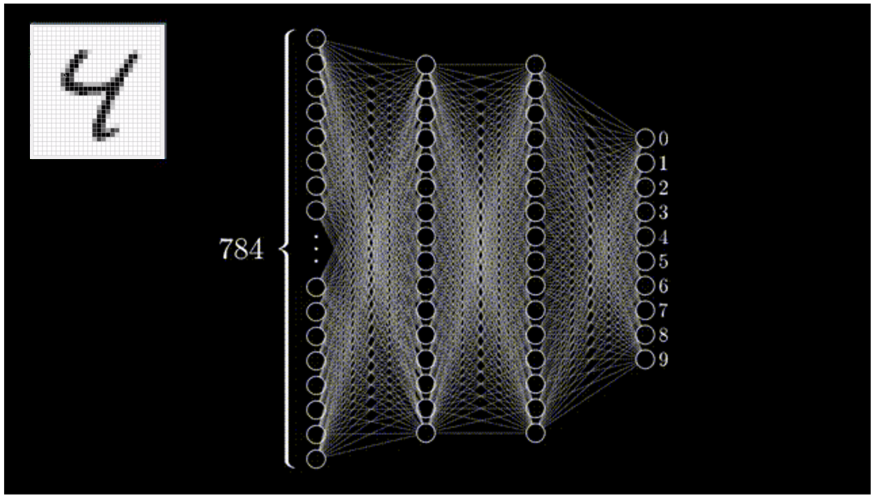
Outline

- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

Classification problem: MNIST







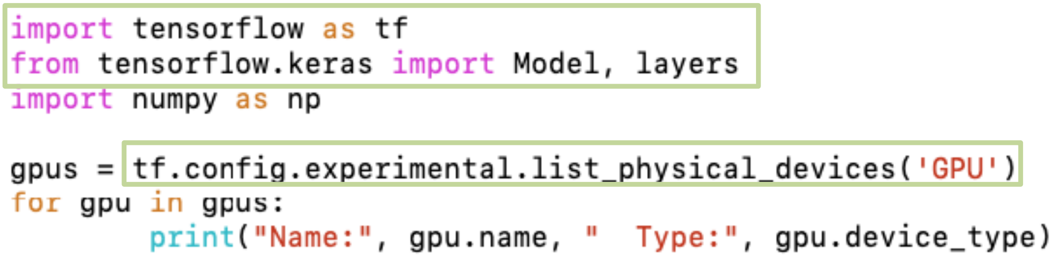
Handwritten data

60K train set and 10K test set

Each image has a size of 28x28 (=784)

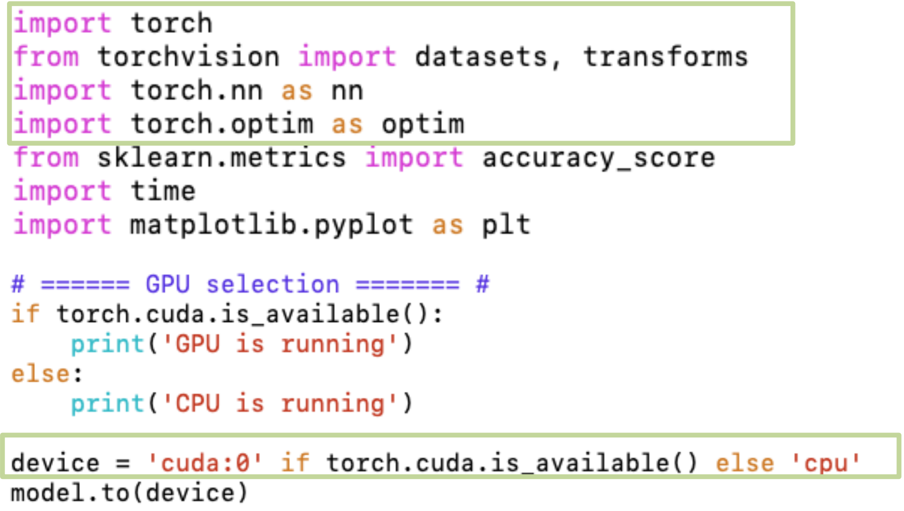
A little peek in the code

TensorFlow



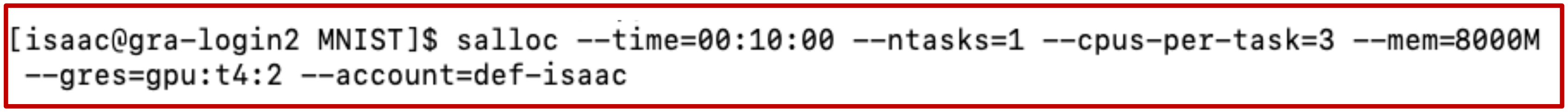
```
import tensorflow as tf
from tensorflow.keras import Model, layers
import numpy as np

gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
        print("Name:", gpu.name, "  Type:", gpu.device_type)
```

PyTorch



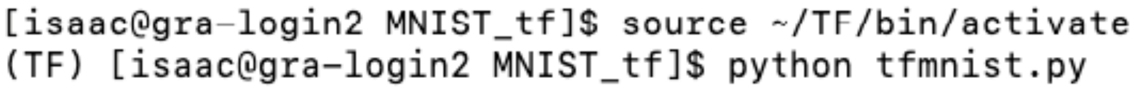
```
import torch
from torchvision import datasets, transforms
import torch.nn as nn
import torch.optim as optim
from sklearn.metrics import accuracy_score
import time
import matplotlib.pyplot as plt

# ====== GPU selection ======= #
if torch.cuda.is_available():
    print('GPU is running')
else:
    print('CPU is running')

device = 'cuda:0' if torch.cuda.is_available() else 'cpu'
model.to(device)
```

Running interactively



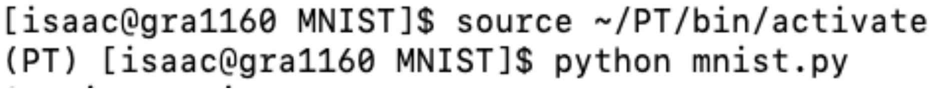
```
[isaac@gra-login2 MNIST]$ salloc --time=00:10:00 --ntasks=1 --cpus-per-task=3 --mem=8000M
 --gres=gpu:t4:2 --account=def-isaac
```

TensorFlow

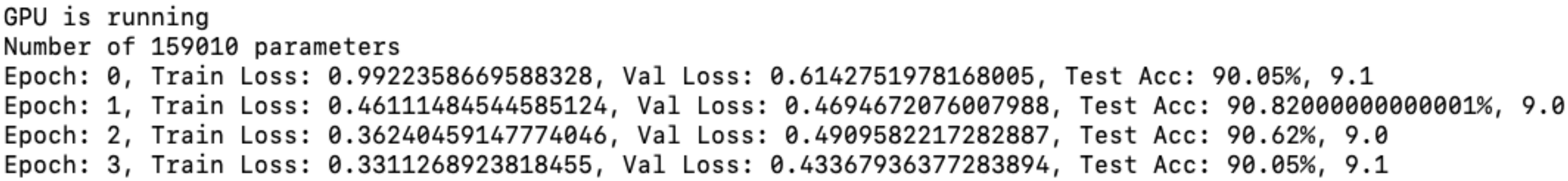


```
[isaac@gra-login2 MNIST_tf]$ source ~/TF/bin/activate
(TF) [isaac@gra-login2 MNIST_tf]$ python tfmnist.py
```

PyTorch



```
[isaac@gra1160 MNIST]$ source ~/PT/bin/activate
(PT) [isaac@gra1160 MNIST]$ python mnist.py
```



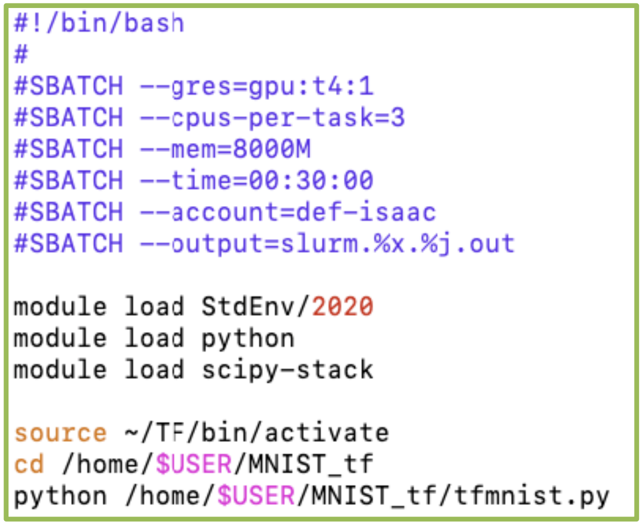
```
GPU is running
Number of 159010 parameters
Epoch: 0, Train Loss: 0.9922358669588328, Val Loss: 0.6142751978168005, Test Acc: 90.05%, 9.1
Epoch: 1, Train Loss: 0.46111484544585124, Val Loss: 0.4694672076007988, Test Acc: 90.82000000000001%, 9.0
Epoch: 2, Train Loss: 0.36240459147774046, Val Loss: 0.4909582217282887, Test Acc: 90.62%, 9.0
Epoch: 3, Train Loss: 0.3311268923818455, Val Loss: 0.43367936377283894, Test Acc: 90.05%, 9.1
```

Outline

- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

Running in scheduler (SLURM)

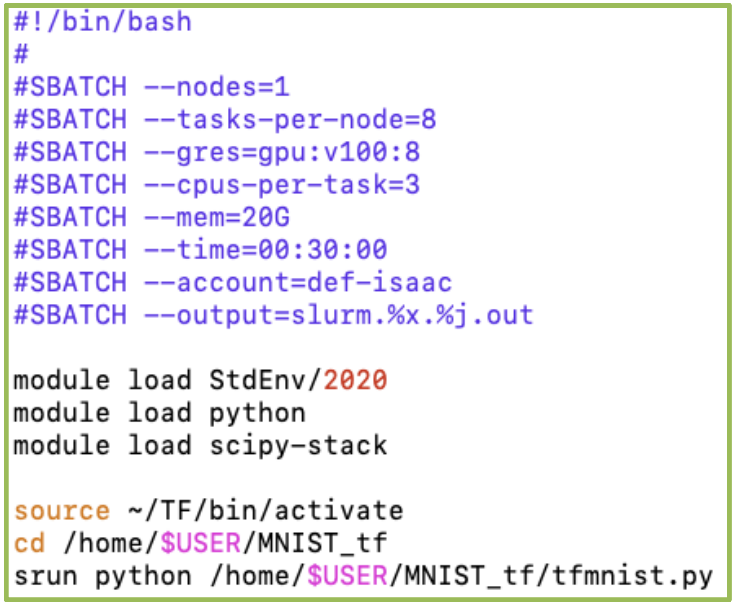
Single GPU in Single Node



```
#!/bin/bash
#
#SBATCH --gres=gpu:t4:1
#SBATCH --cpus-per-task=3
#SBATCH --mem=8000M
#SBATCH --time=00:30:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/TF/bin/activate
cd /home/$USER/MNIST_tf
python /home/$USER/MNIST_tf/tfmnist.py
```

TensorFlow

Multi-GPUs in Single Node

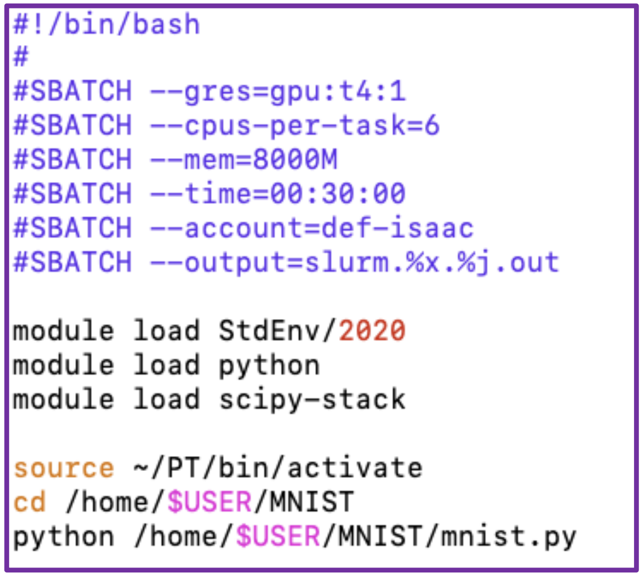


```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=8
#SBATCH --gres=gpu:v100:8
#SBATCH --cpus-per-task=3
#SBATCH --mem=20G
#SBATCH --time=00:30:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/TF/bin/activate
cd /home/$USER/MNIST_tf
srun python /home/$USER/MNIST_tf/tfmnist.py
```

Note: **CPU to GPU** ratio recommended to have **less than 3.5**

Running in scheduler (SLURM)

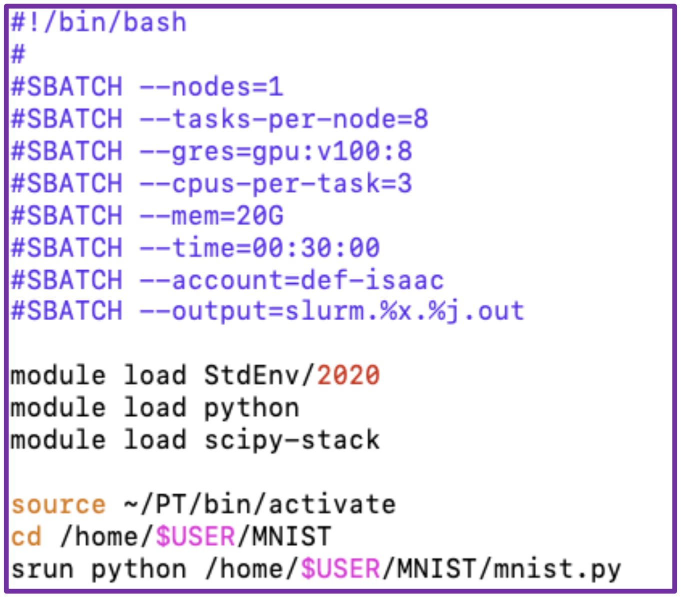
Single GPU in Single Node



```
#!/bin/bash
#
#SBATCH --gres=gpu:t4:1
#SBATCH --cpus-per-task=6
#SBATCH --mem=8000M
#SBATCH --time=00:30:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/PT/bin/activate
cd /home/$USER/MNIST
python /home/$USER/MNIST/mnist.py
```

PyTorch

Multi-GPUs in Single Node



```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=8
#SBATCH --gres=gpu:v100:8
#SBATCH --cpus-per-task=3
#SBATCH --mem=20G
#SBATCH --time=00:30:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/PT/bin/activate
cd /home/$USER/MNIST
srun python /home/$USER/MNIST/mnist.py
```

Note: **CPU to GPU** ratio recommended to have **less than 3.5**

SHARCNET™
GENERAL INTEREST SEMINAR

Outline

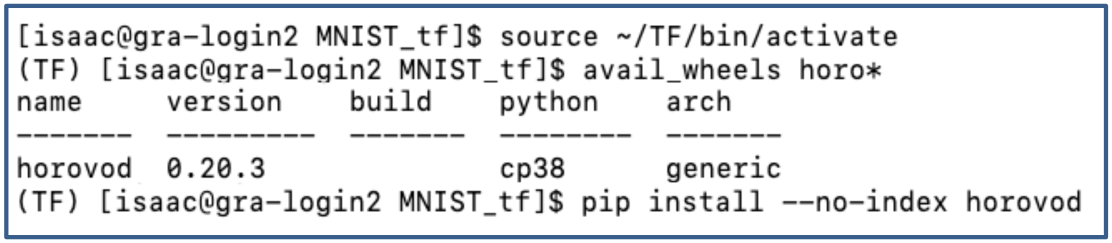
- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- **Running in SLURM (Multi-GPUs in multi-nodes)**
- Tensorboard

compute canada | calcul canada

HOROVOD

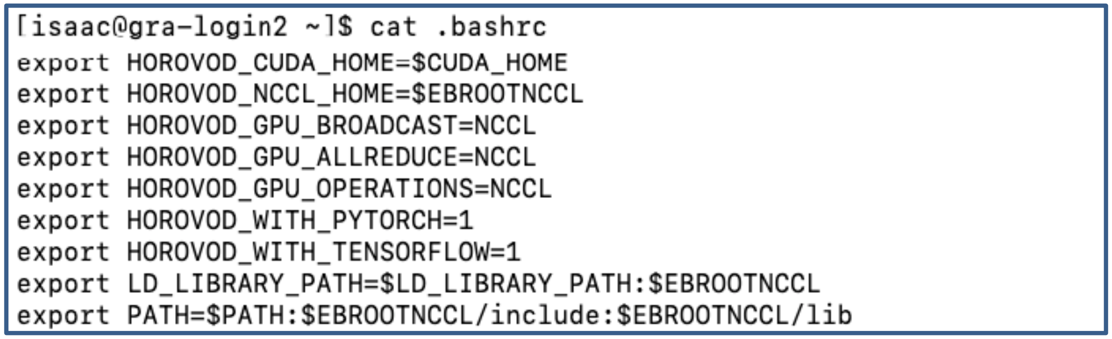
Distributed deep learning training framework

Installation



```
[isaac@gra-login2 MNIST_tf]$ source ~/TF/bin/activate
(TF) [isaac@gra-login2 MNIST_tf]$ avail_wheels horo*
name     version    build    python    arch
-------  ---------  -------  --------  -------
horovod  0.20.3              cp38      generic
(TF) [isaac@gra-login2 MNIST_tf]$ pip install --no-index horovod
```

Environment



```
[isaac@gra-login2 ~]$ cat .bashrc
export HOROVOD_CUDA_HOME=$CUDA_HOME
export HOROVOD_NCCL_HOME=$EBROOTNCCL
export HOROVOD_GPU_BROADCAST=NCCL
export HOROVOD_GPU_ALLREDUCE=NCCL
export HOROVOD_GPU_OPERATIONS=NCCL
export HOROVOD_WITH_PYTORCH=1
export HOROVOD_WITH_TENSORFLOW=1
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$EBROOTNCCL
export PATH=$PATH:$EBROOTNCCL/include:$EBROOTNCCL/lib
```

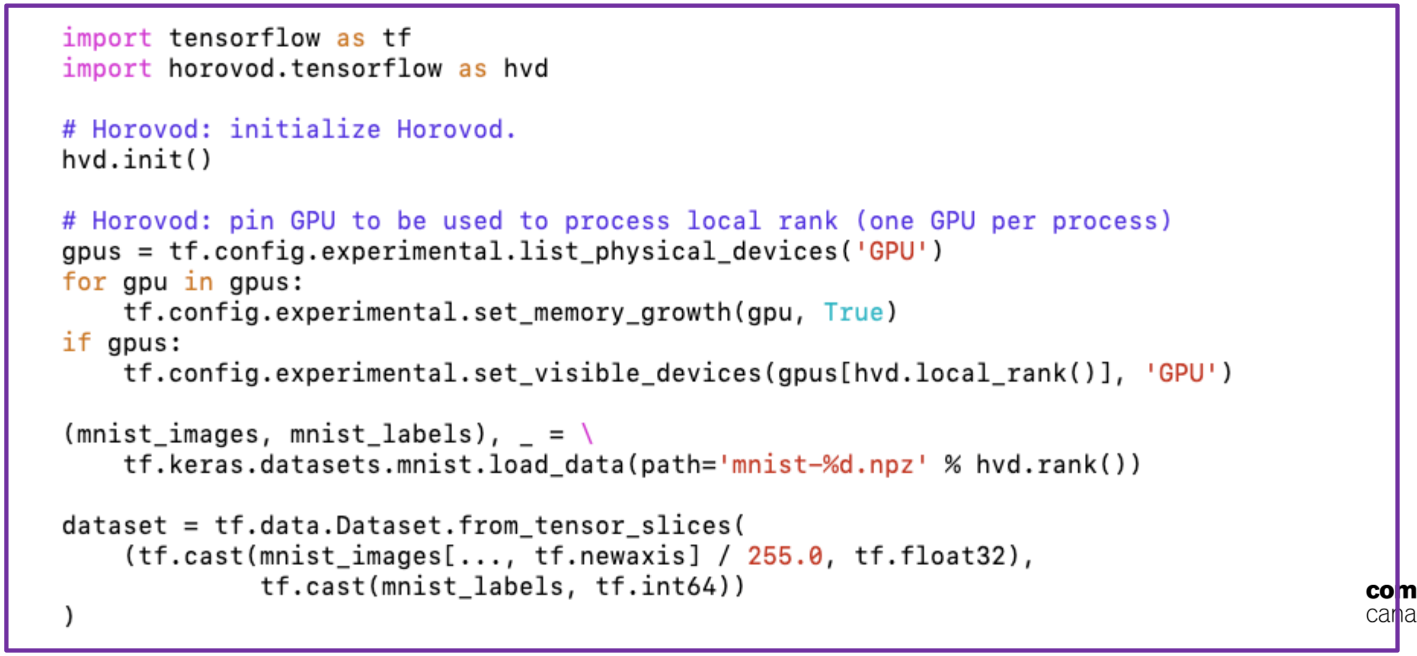
TensorFlow

PyTorch

TensorFlow + HOROVOD

Multiple GPUs in Multi-nodes

HOROVOD



```
import tensorflow as tf
import horovod.tensorflow as hvd

# Horovod: initialize Horovod.
hvd.init()

# Horovod: pin GPU to be used to process local rank (one GPU per process)
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)
if gpus:
    tf.config.experimental.set_visible_devices(gpus[hvd.local_rank()], 'GPU')

(mnist_images, mnist_labels), _ = \
    tf.keras.datasets.mnist.load_data(path='mnist-%d.npz' % hvd.rank())

dataset = tf.data.Dataset.from_tensor_slices(
    (tf.cast(mnist_images[..., tf.newaxis] / 255.0, tf.float32),
             tf.cast(mnist_labels, tf.int64))
)
```

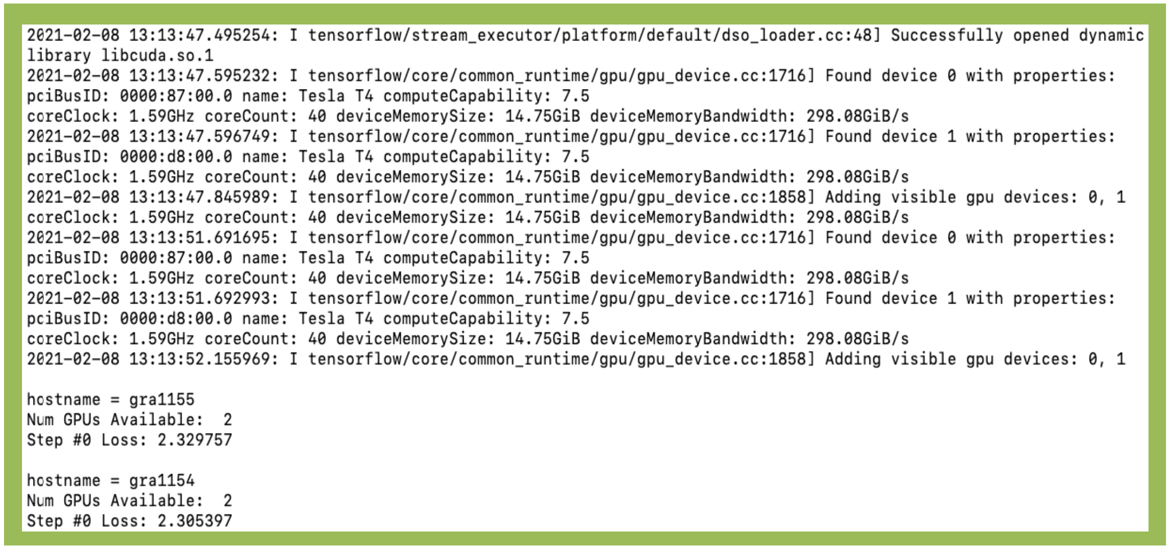
TensorFlow + HOROVOD

Multi-GPU in Multi-Node

HOROVOD

```
#!/bin/bash
#
#SBATCH --nodes=2
#SBATCH --gres=gpu:t4:2
#SBATCH --tasks-per-node=2
#SBATCH --mem=10G
#SBATCH --cpus-per-task=3

#SBATCH --time=00:10:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python scipy-stack
module load cuda cudnn
module load nccl

source ~/.bashrc
source ~/TF/bin/activate
cd /home/$USER/MNIST_tf

srun python /home/$USER/MNIST_tf/mnisthor.py --log-dir distributed
--variable_update horovod
```

TensorFlow

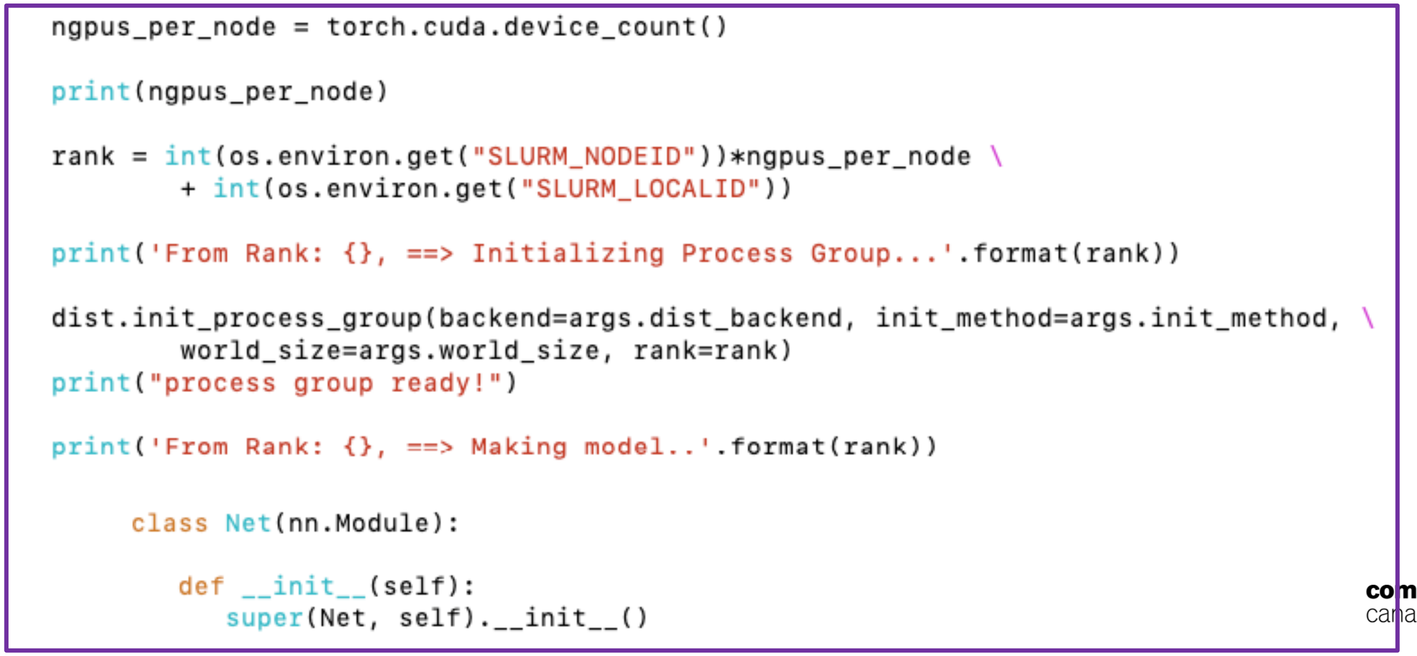


```
2021-02-08 13:13:47.495254: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic
library libcuda.so.1
2021-02-08 13:13:47.595232: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1716] Found device 0 with properties:
pciBusID: 0000:87:00.0 name: Tesla T4 computeCapability: 7.5
coreClock: 1.59GHz coreCount: 40 deviceMemorySize: 14.75GiB deviceMemoryBandwidth: 298.08GiB/s
2021-02-08 13:13:47.596749: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1716] Found device 1 with properties:
pciBusID: 0000:d8:00.0 name: Tesla T4 computeCapability: 7.5
coreClock: 1.59GHz coreCount: 40 deviceMemorySize: 14.75GiB deviceMemoryBandwidth: 298.08GiB/s
2021-02-08 13:13:47.845989: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Adding visible gpu devices: 0, 1
coreClock: 1.59GHz coreCount: 40 deviceMemorySize: 14.75GiB deviceMemoryBandwidth: 298.08GiB/s
2021-02-08 13:13:51.691695: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1716] Found device 0 with properties:
pciBusID: 0000:87:00.0 name: Tesla T4 computeCapability: 7.5
coreClock: 1.59GHz coreCount: 40 deviceMemorySize: 14.75GiB deviceMemoryBandwidth: 298.08GiB/s
2021-02-08 13:13:51.692993: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1716] Found device 1 with properties:
pciBusID: 0000:d8:00.0 name: Tesla T4 computeCapability: 7.5
coreClock: 1.59GHz coreCount: 40 deviceMemorySize: 14.75GiB deviceMemoryBandwidth: 298.08GiB/s
2021-02-08 13:13:52.155969: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Adding visible gpu devices: 0, 1

hostname = gra1155
Num GPUs Available:  2
Step #0 Loss: 2.329757

hostname = gra1154
Num GPUs Available:  2
Step #0 Loss: 2.305397
```

PyTorch + DDP

Multiple GPUs in Multi-nodes

Distributed Data Parallel (DDP)



```
ngpus_per_node = torch.cuda.device_count()

print(ngpus_per_node)

rank = int(os.environ.get("SLURM_NODEID"))*ngpus_per_node \
        + int(os.environ.get("SLURM_LOCALID"))

print('From Rank: {}, ==> Initializing Process Group...'.format(rank))

dist.init_process_group(backend=args.dist_backend, init_method=args.init_method, \
        world_size=args.world_size, rank=rank)
print("process group ready!")

print('From Rank: {}, ==> Making model..'.format(rank))


     class Net(nn.Module):

        def __init__(self):
           super(Net, self).__init__()
```

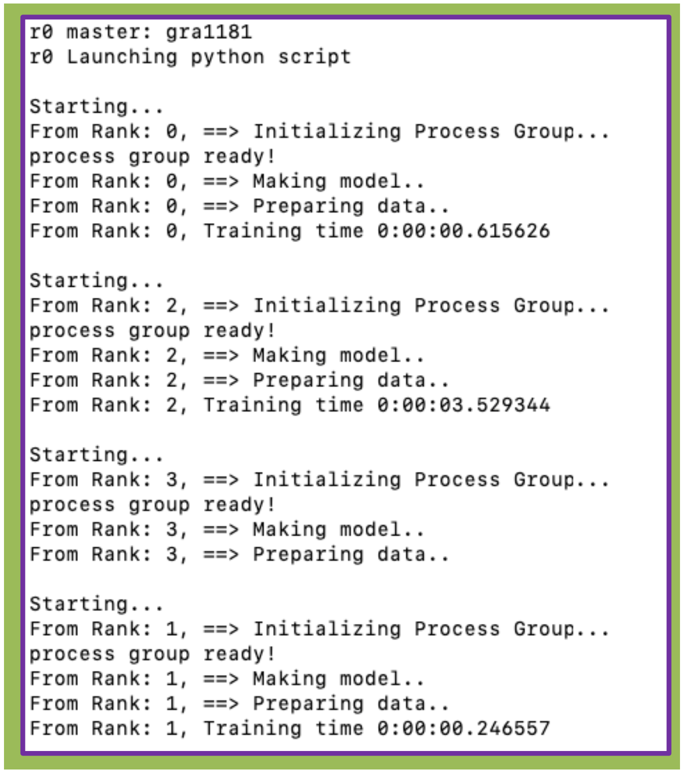
PyTorch + DDP

Multi-GPU in Multi-Node

- Distributed Data Parallel (DDP)

```
#!/bin/bash
#
#SBATCH --nodes=2
#SBATCH --gres=gpu:t4:2
#SBATCH --tasks-per-node=2
#SBATCH --mem=15G
#SBATCH --cpus-per-task=3

#SBATCH --time=00:30:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/PT/bin/activate
cd /home/$USER/MNIST

export MASTER_ADDR=$(hostname)
echo "r$SLURM_NODEID master: $MASTER_ADDR"
echo "r$SLURM_NODEID Launching python script"

srun python /home/$USER/MNIST/mnistddp.py --init_method tcp://$MASTER_ADDR:3456
--world_size $SLURM_NTASKS
```

PyTorch



```
r0 master: gra1181
r0 Launching python script

Starting...
From Rank: 0, ==> Initializing Process Group...
process group ready!
From Rank: 0, ==> Making model..
From Rank: 0, ==> Preparing data..
From Rank: 0, Training time 0:00:00.615626

Starting...
From Rank: 2, ==> Initializing Process Group...
process group ready!
From Rank: 2, ==> Making model..
From Rank: 2, ==> Preparing data..
From Rank: 2, Training time 0:00:03.529344

Starting...
From Rank: 3, ==> Initializing Process Group...
process group ready!
From Rank: 3, ==> Making model..
From Rank: 3, ==> Preparing data..

Starting...
From Rank: 1, ==> Initializing Process Group...
process group ready!
From Rank: 1, ==> Making model..
From Rank: 1, ==> Preparing data..
From Rank: 1, Training time 0:00:00.246557
```

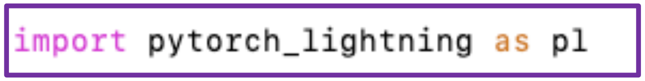
SHARCNET™
GENERAL INTEREST SEMINAR

PyTorch + PyTorch Lighting

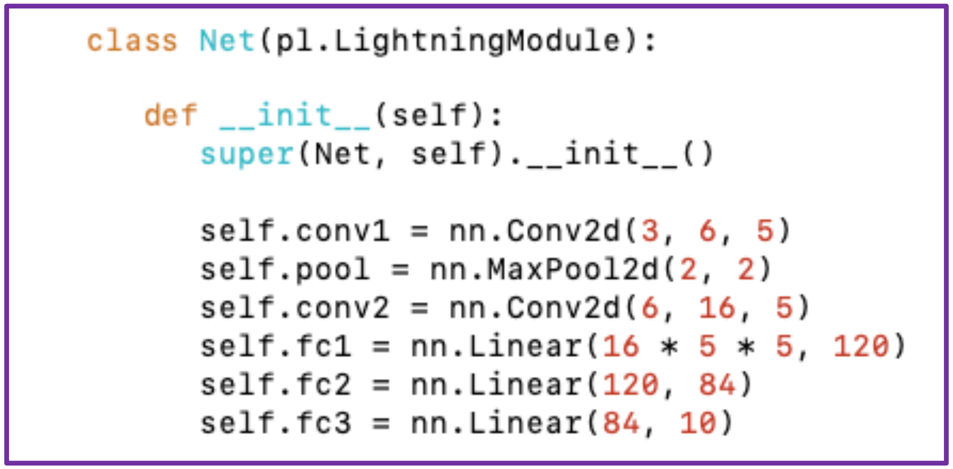
Multiple GPUs in Multi-nodes

Note: pytorch-lightning is currently not available in wheels. Please install it manually by
'pip install pytorch-lightning'

PyTorch Lightning



```
import pytorch_lightning as pl
```



```
class Net(pl.LightningModule):

    def __init__(self):
        super(Net, self).__init__()

        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)
```

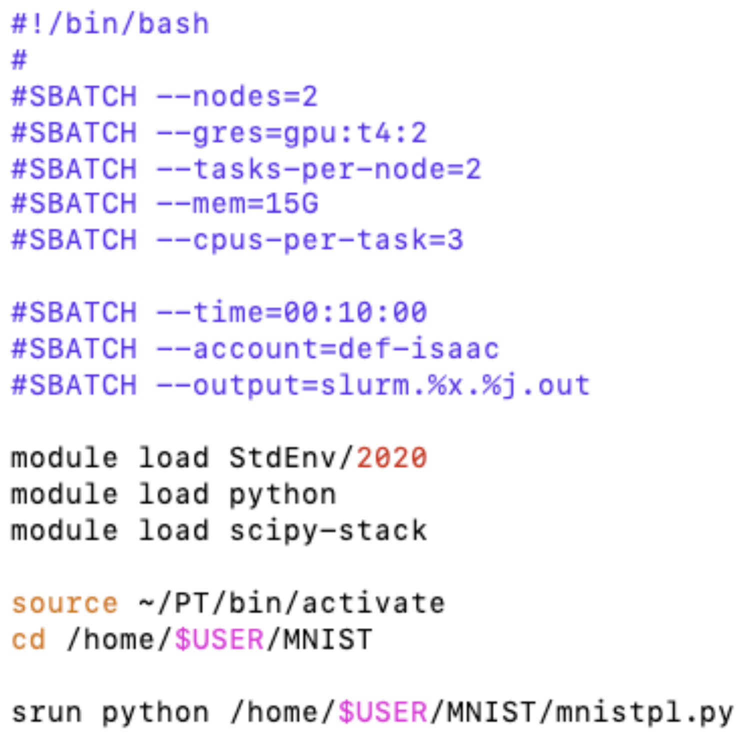
PyTorch

compute canada | calcul canada

PyTorch + PyTorch Lighting

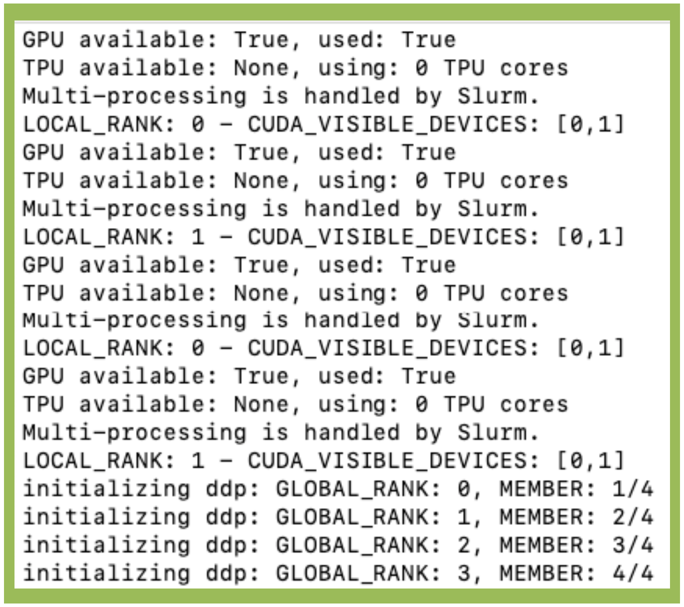
Multi-GPU in Multi-Node

- PyTorch Lightning



```
#!/bin/bash
#
#SBATCH --nodes=2
#SBATCH --gres=gpu:t4:2
#SBATCH --tasks-per-node=2
#SBATCH --mem=15G
#SBATCH --cpus-per-task=3

#SBATCH --time=00:10:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack

source ~/PT/bin/activate
cd /home/$USER/MNIST

srun python /home/$USER/MNIST/mnistpl.py
```

PyTorch



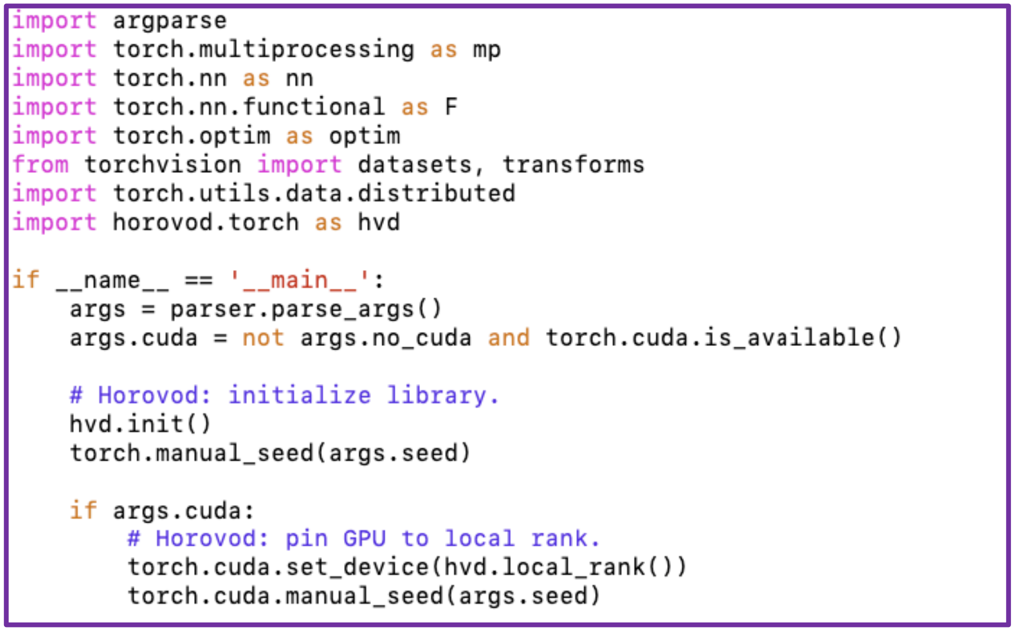
```
GPU available: True, used: True
TPU available: None, using: 0 TPU cores
Multi-processing is handled by Slurm.
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0,1]
GPU available: True, used: True
TPU available: None, using: 0 TPU cores
Multi-processing is handled by Slurm.
LOCAL_RANK: 1 - CUDA_VISIBLE_DEVICES: [0,1]
GPU available: True, used: True
TPU available: None, using: 0 TPU cores
Multi-processing is handled by Slurm.
LOCAL_RANK: 0 - CUDA_VISIBLE_DEVICES: [0,1]
GPU available: True, used: True
TPU available: None, using: 0 TPU cores
Multi-processing is handled by Slurm.
LOCAL_RANK: 1 - CUDA_VISIBLE_DEVICES: [0,1]
initializing ddp: GLOBAL_RANK: 0, MEMBER: 1/4
initializing ddp: GLOBAL_RANK: 1, MEMBER: 2/4
initializing ddp: GLOBAL_RANK: 2, MEMBER: 3/4
initializing ddp: GLOBAL_RANK: 3, MEMBER: 4/4
```

PyTorch + HOROVOD

Multiple GPUs in Multi-nodes

Note: You need to install horovod in your virtual environment
'pip install –no-index horovod'

Horovod



```python
import argparse
import torch.multiprocessing as mp
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
from torchvision import datasets, transforms
import torch.utils.data.distributed
import horovod.torch as hvd

if __name__ == '__main__':
    args = parser.parse_args()
    args.cuda = not args.no_cuda and torch.cuda.is_available()

    # Horovod: initialize library.
    hvd.init()
    torch.manual_seed(args.seed)

    if args.cuda:
        # Horovod: pin GPU to local rank.
        torch.cuda.set_device(hvd.local_rank())
        torch.cuda.manual_seed(args.seed)
```

PyTorch

compute | calcul
canada | canada

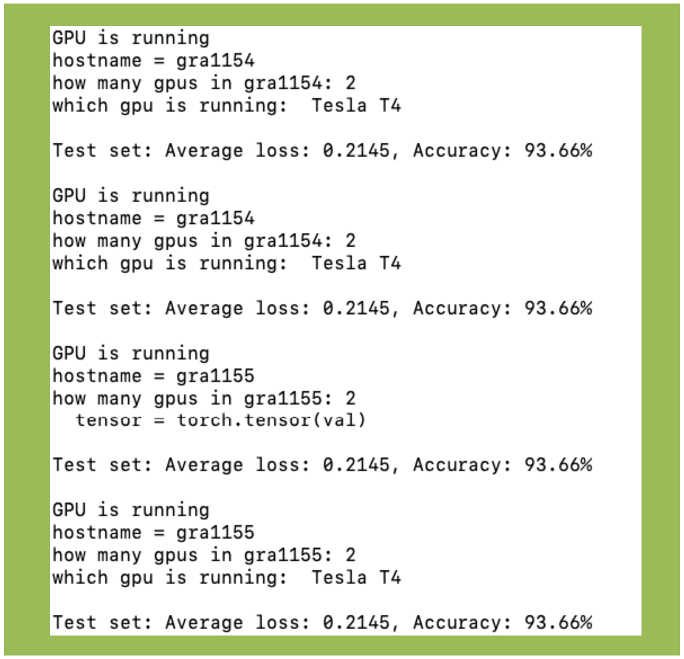
GENERAL INTEREST SEMINAR

PyTorch + HOROVOD

Multi-GPU in Multi-Node

- PyTorch Lightning

```
#!/bin/bash
#
#SBATCH --nodes=2
#SBATCH --gres=gpu:t4:2
#SBATCH --tasks-per-node=2
#SBATCH --mem=10G
#SBATCH --cpus-per-task=3

#SBATCH --time=00:10:00
#SBATCH --account=def-isaac
#SBATCH --output=slurm.%x.%j.out

module load StdEnv/2020
module load python
module load scipy-stack
module load cuda cudnn
module load nccl

source ~/.bashrc
source ~/PT/bin/activate
cd /home/$USER/MNIST

srun python /home/$USER/MNIST/mnisthor.py
```



```
GPU is running
hostname = gra1154
how many gpus in gra1154: 2
which gpu is running:  Tesla T4

Test set: Average loss: 0.2145, Accuracy: 93.66%

GPU is running
hostname = gra1154
how many gpus in gra1154: 2
which gpu is running:  Tesla T4

Test set: Average loss: 0.2145, Accuracy: 93.66%

GPU is running
hostname = gra1155
how many gpus in gra1155: 2
  tensor = torch.tensor(val)

Test set: Average loss: 0.2145, Accuracy: 93.66%

GPU is running
hostname = gra1155
how many gpus in gra1155: 2
which gpu is running:  Tesla T4

Test set: Average loss: 0.2145, Accuracy: 93.66%
```

SHARCNET™
GENERAL INTEREST SEMINAR

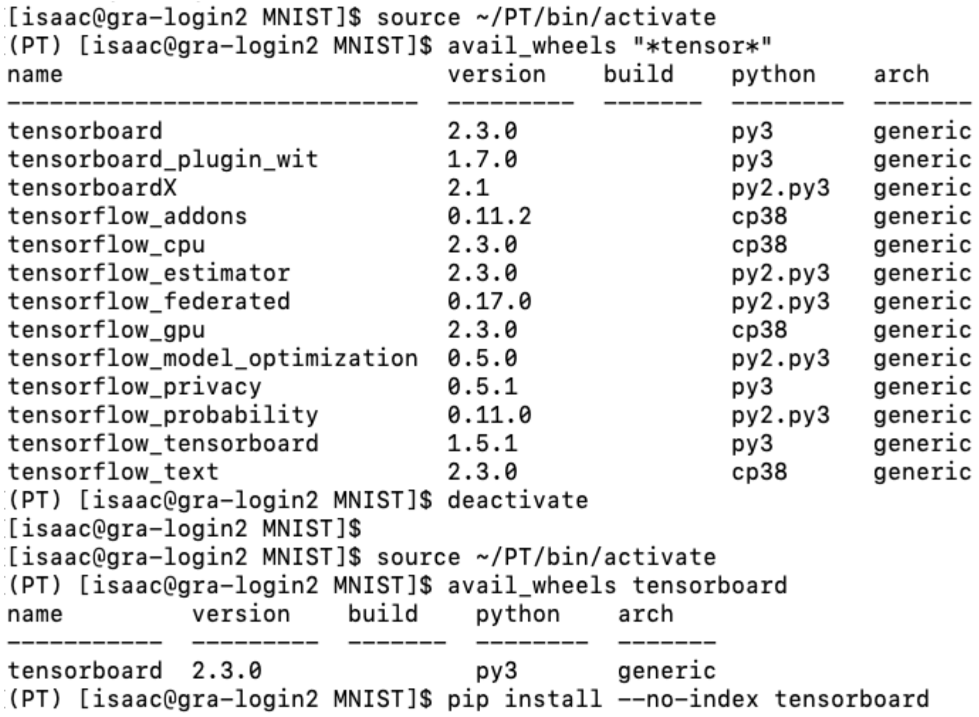
Outline

- DNN & Parallelism (Data vs Model)
- TensorFlow vs PyTorch
- GPUs and Virtual Environment
- Running interactively
- Running in SLURM (Multi-GPUs in single node)
- Running in SLURM (Multi-GPUs in multi-nodes)
- Tensorboard

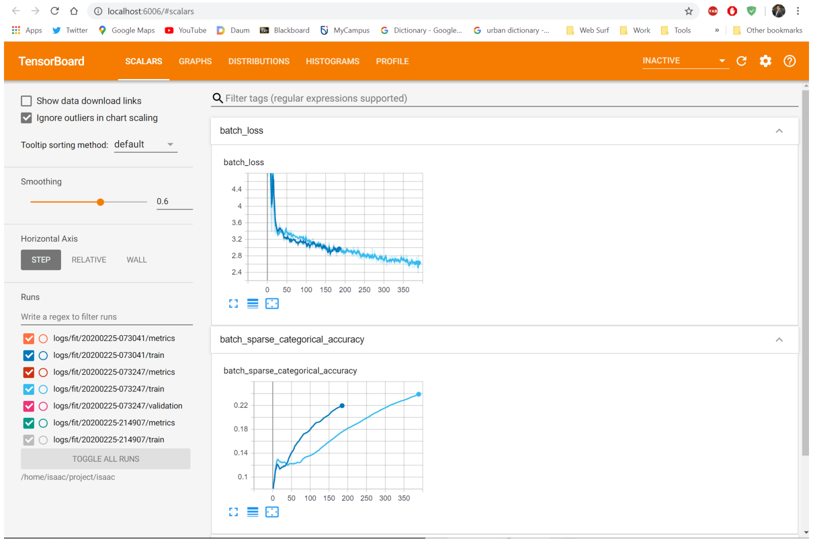
compute canada | calcul canada

SHARCNET™
GENERAL INTEREST SEMINAR

Tensorboard + PyTorch



```
[isaac@gra-login2 MNIST]$ source ~/PT/bin/activate
(PT) [isaac@gra-login2 MNIST]$ avail_wheels "*tensor*"
name                           version    build    python    arch
-----------------------------  ---------  -------  --------  -------
tensorboard                    2.3.0               py3       generic
tensorboard_plugin_wit         1.7.0               py3       generic
tensorboardX                   2.1                 py2.py3   generic
tensorflow_addons              0.11.2              cp38      generic
tensorflow_cpu                 2.3.0               cp38      generic
tensorflow_estimator           2.3.0               py2.py3   generic
tensorflow_federated           0.17.0              py2.py3   generic
tensorflow_gpu                 2.3.0               cp38      generic
tensorflow_model_optimization  0.5.0               py2.py3   generic
tensorflow_privacy             0.5.1               py3       generic
tensorflow_probability         0.11.0              py2.py3   generic
tensorflow_tensorboard         1.5.1               py3       generic
tensorflow_text                2.3.0               cp38      generic
(PT) [isaac@gra-login2 MNIST]$ deactivate
[isaac@gra-login2 MNIST]$
[isaac@gra-login2 MNIST]$ source ~/PT/bin/activate
(PT) [isaac@gra-login2 MNIST]$ avail_wheels tensorboard
name         version    build    python    arch
-----------  ---------  -------  --------  -------
tensorboard  2.3.0               py3       generic
(PT) [isaac@gra-login2 MNIST]$ pip install --no-index tensorboard
```



localhost:6006/#scalars

☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: STEP

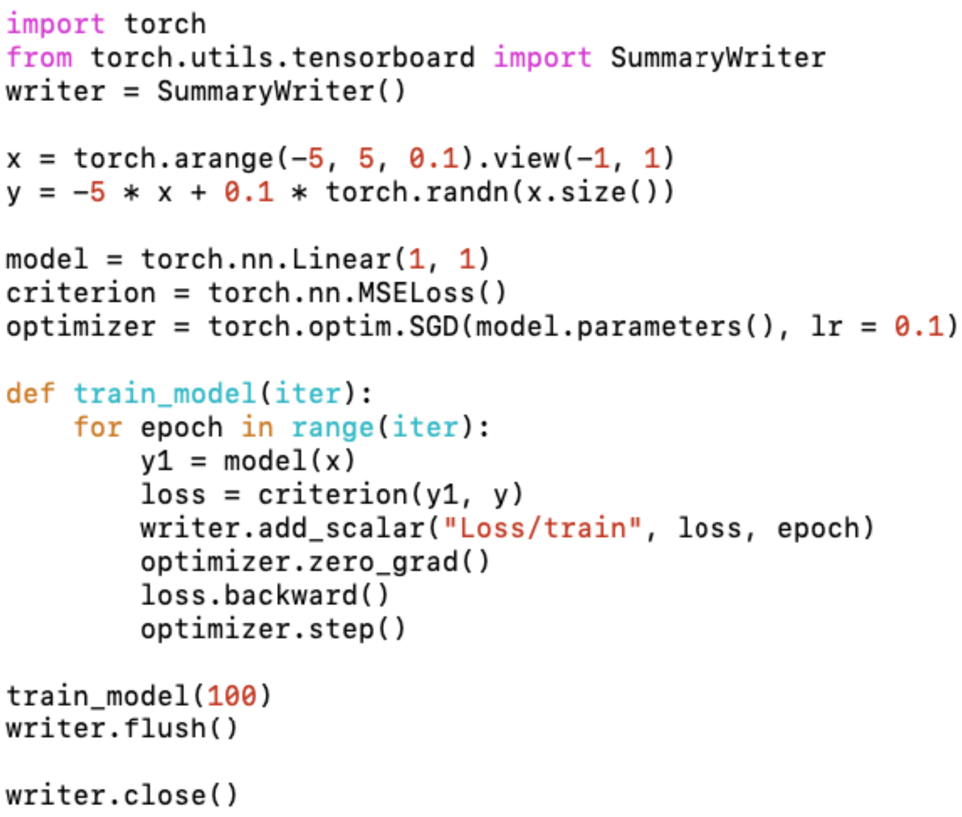
Runs

☑ ○ logs/fit/20200225-073041/metrics
☑ ○ logs/fit/20200225-073041/train
☑ ○ logs/fit/20200225-073247/metrics
☑ ○ logs/fit/20200225-073247/train
☑ ○ logs/fit/20200225-073247/validation
☑ ○ logs/fit/20200225-214907/metrics
☑ ○ logs/fit/20200225-214907/train

/home/isaac/project/isaac

compute | calcul
canada | canada

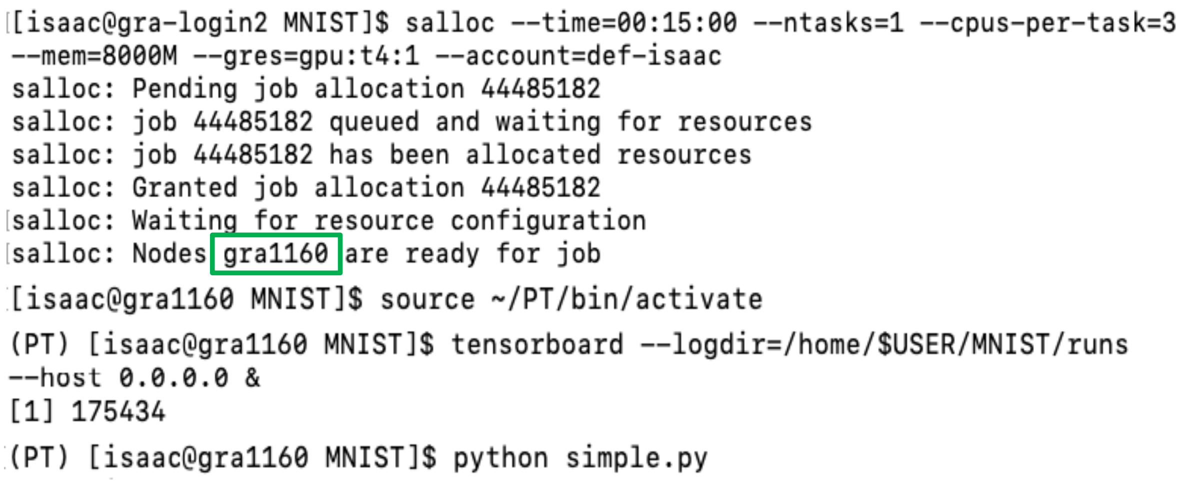
Test code



```
import torch
from torch.utils.tensorboard import SummaryWriter
writer = SummaryWriter()

x = torch.arange(-5, 5, 0.1).view(-1, 1)
y = -5 * x + 0.1 * torch.randn(x.size())

model = torch.nn.Linear(1, 1)
criterion = torch.nn.MSELoss()
optimizer = torch.optim.SGD(model.parameters(), lr = 0.1)

def train_model(iter):
    for epoch in range(iter):
        y1 = model(x)
        loss = criterion(y1, y)
        writer.add_scalar("Loss/train", loss, epoch)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

train_model(100)
writer.flush()

writer.close()
```

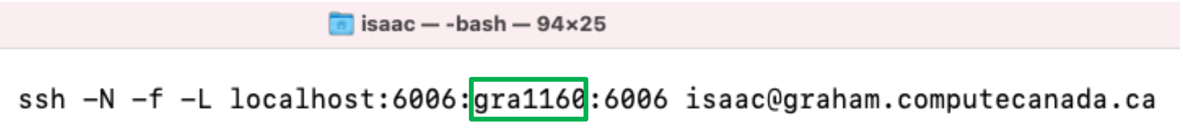
How to use ?

In Graham (Interactive or Slurm)



```
[isaac@gra-login2 MNIST]$ salloc --time=00:15:00 --ntasks=1 --cpus-per-task=3
--mem=8000M --gres=gpu:t4:1 --account=def-isaac
salloc: Pending job allocation 44485182
salloc: job 44485182 queued and waiting for resources
salloc: job 44485182 has been allocated resources
salloc: Granted job allocation 44485182
salloc: Waiting for resource configuration
salloc: Nodes gra1160 are ready for job
[isaac@gra1160 MNIST]$ source ~/PT/bin/activate
(PT) [isaac@gra1160 MNIST]$ tensorboard --logdir=/home/$USER/MNIST/runs
--host 0.0.0.0 &
[1] 175434
(PT) [isaac@gra1160 MNIST]$ python simple.py
```

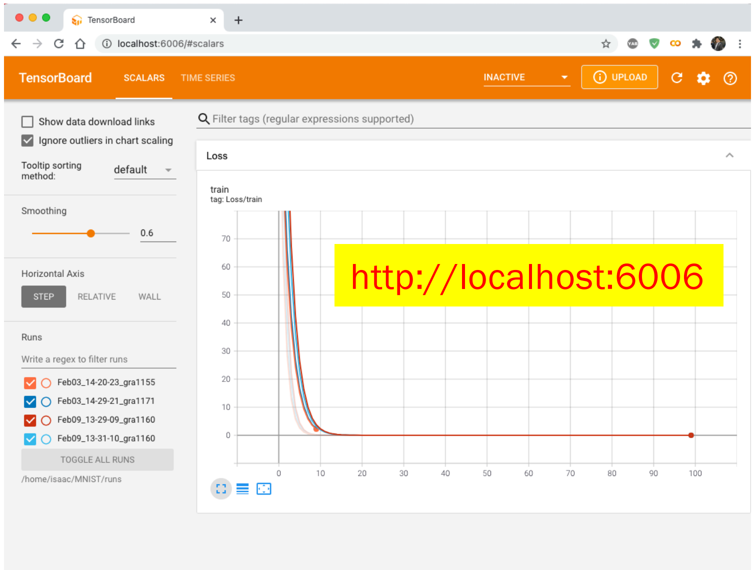
In your PC



isaac — -bash — 94×25

```
ssh -N -f -L localhost:6006:gra1160:6006 isaac@graham.computecanada.ca
```

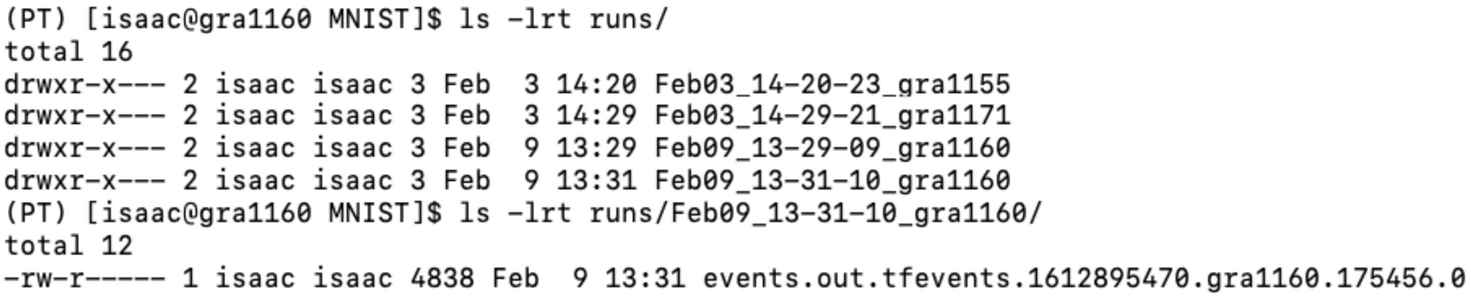
In your PC



☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP
Runs
☑ ○ Feb03_14-20-23_gra1155
☑ ○ Feb03_14-29-21_gra1171
☑ ○ Feb09_13-29-09_gra1160
☑ ○ Feb09_13-31-10_gra1160
/home/isaac/MNIST/runs

http://localhost:6006

Logs



```
(PT) [isaac@gra1160 MNIST]$ ls -lrt runs/
total 16
drwxr-x--- 2 isaac isaac 3 Feb  3 14:20 Feb03_14-20-23_gra1155
drwxr-x--- 2 isaac isaac 3 Feb  3 14:29 Feb03_14-29-21_gra1171
drwxr-x--- 2 isaac isaac 3 Feb  9 13:29 Feb09_13-29-09_gra1160
drwxr-x--- 2 isaac isaac 3 Feb  9 13:31 Feb09_13-31-10_gra1160
(PT) [isaac@gra1160 MNIST]$ ls -lrt runs/Feb09_13-31-10_gra1160/
total 12
-rw-r----- 1 isaac isaac 4838 Feb  9 13:31 events.out.tfevents.1612895470.gra1160.175456.0
```

Thanks!

Q & A

You can find all testing files in this seminar here:

https://sharcnet.ca/~isaac/GIS2020Feb10.tar.gz