
Using Tensorboard to debug and profile NN training

Weiguang Guan (guanw@sharcnet.ca)
SHARNet/Compute Canada

Goals

- How to use Tensorboard on CC clusters
- How to use the Debugger V2 to detect/fix anomalies (NaN, Infinity ...)
- How to use profiler to find performance bottlenecks and improve the performance issues

Reference

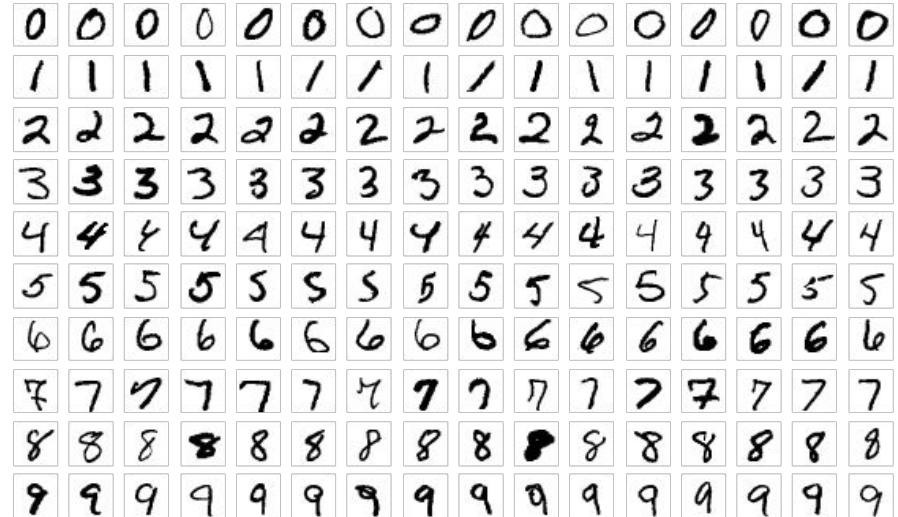
- Tutorials and guide on Tensorflow official website
[\(https://www.tensorflow.org/\)](https://www.tensorflow.org/)
- Wiki <https://docs.computecanada.ca/wiki/TensorFlow>
- Code: <https://staff.sharcnet.ca/guanw/2022/2022.tar.gz>

Datasets used in this seminar

- MNIST
- CIFAR10

MNIST dataset

- 60000 training samples + 10000 testing samples
- 28x28 gray scale image with values ranging [0, 255]
- Label is an integer [0, 9]



MNIST dataset (cont.)

Different ways of loading MNIST data

- `tf.keras.datasets.mnist.load_data(...)`
Default location: `~/keras/datasets`
- `tensorflow_datasets.load('mnist', ...)`
Default location: `~/tensorflow_datasets/`

Pre-download the dataset on a login node

- `tf.keras.datasets.mnist.load_data()`
- Import `tensorflow_datasets` as `tfds`
`tfds.load('mnist', split=['train', 'test'],
shuffle_files=True, as_supervised=True)`

Other datasets

Dataset repository on Graham

- /datashare on Graham
- Create a symbolic link
 - cd ~/.keras/datasets
 - ln -s /datashare/CIFAR-10/cifar-10-python.tar.gz
cifar-10-batches-py.tar.gz

Dataset catalog

- https://www.tensorflow.org/api_docs/python/tf/keras/datasets
- https://www.tensorflow.org/datasets/catalog/overview#all_datasets

Use Tensorboard on a CC cluster

- Get on a compute node (GPU node is preferred)
`salloc --account=def-guanw --time=01:10:0 --cpus-per-task=4
--mem=16G --gres=gpu:1`
- On a compute node:
 - Load modules and activate virtual environment
 - `python abc.py`
 - `tensorboard --logdir=your_log_dir --host 0.0.0.0 --load_fast false &`
- On a local terminal:
`ssh -L 6006:computenode:6006 userid@cluster.computecanada.ca`
- On a local web browser
`http://localhost:6006/`

Use Tensorboard on a CC cluster (cont.)

In abc.py

...

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(  
    log_dir='tmp',  
    histogram_freq=1,  
    profile_batch = '500,520'  
)  
model.fit(x=x_train,  
          y=y_train,  
          epochs=5,  
          validation_data=(x_test, y_test),  
          callbacks=[tensorboard_callback])  
...
```



The Debugger V2

Debugging neural network model training

Examples of anomalies:

- Infinity: $\log(x)$ when $x=0$
- NaN: \sqrt{x} when $x<0$

Debugging neural network model training

- `tf.debugging.enable_check_numerics(...)`
- `tf.debugging.experimental.enable_dump_debug_info(`
 `dump_root,`
 `tensor_debug_mode = "FULL_HEALTH"`
 `...`
 `)`



The Profiler

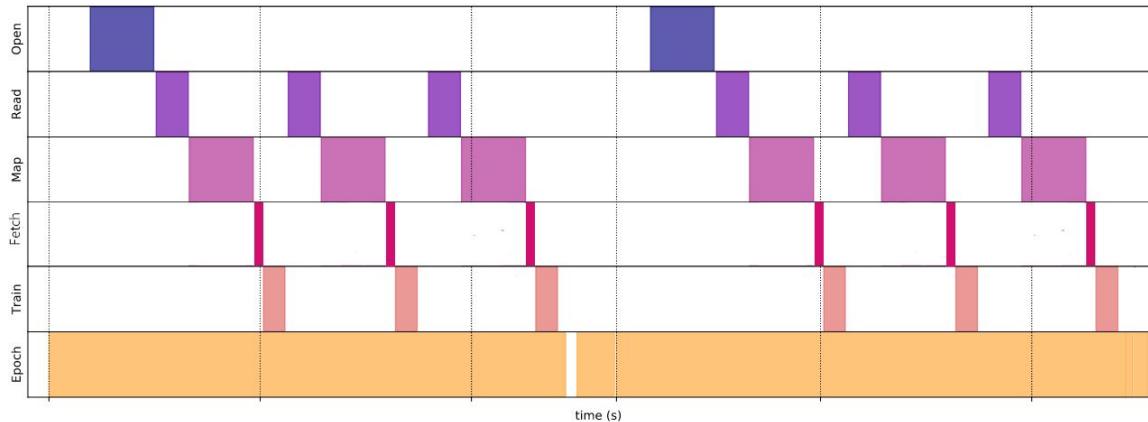
Profiling neural network model training

Example: https://www.tensorflow.org/tensorboard/tensorboard_profiling_keras

```
...
tensorboard_callback = tf.keras.callbacks.TensorBoard(
    log_dir='tmp',
    histogram_freq=1,
    profile_batch = '500,520'
)
...
model.fit(x=x_train,
    y=y_train,
    epochs=5,
    validation_data=(x_test, y_test),
    callbacks=[tensorboard_callback])
```

Input pipeline (overview)

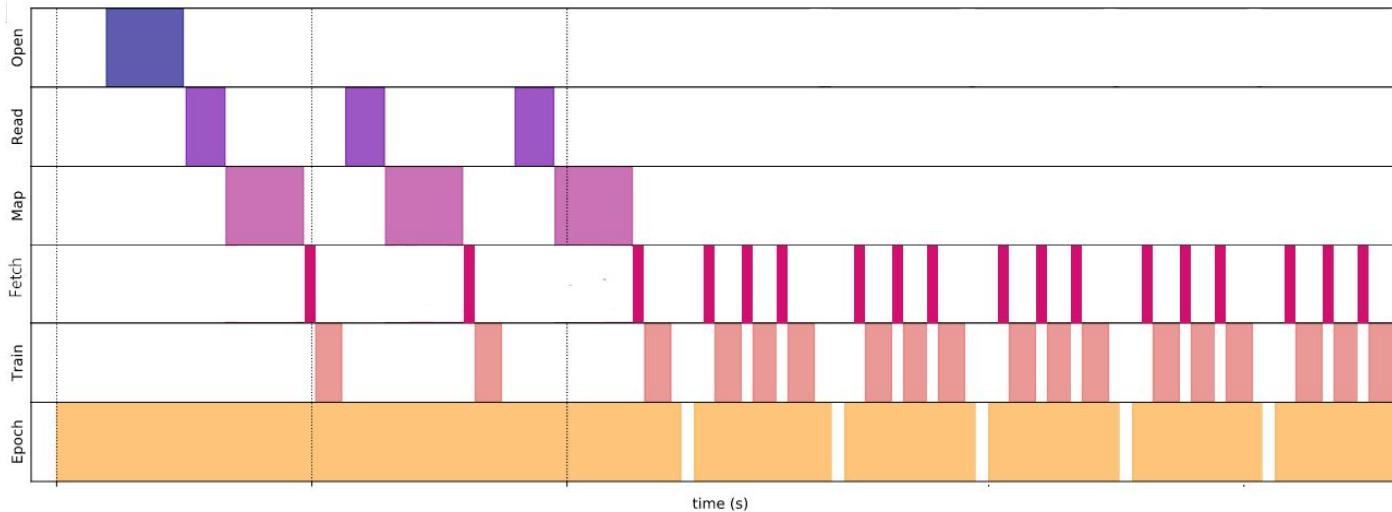
- Open data file
- Read data into memory
- Transform data
- Fetch a batch of data from memory to GPU
- Training



Input pipeline (caching)

Caching

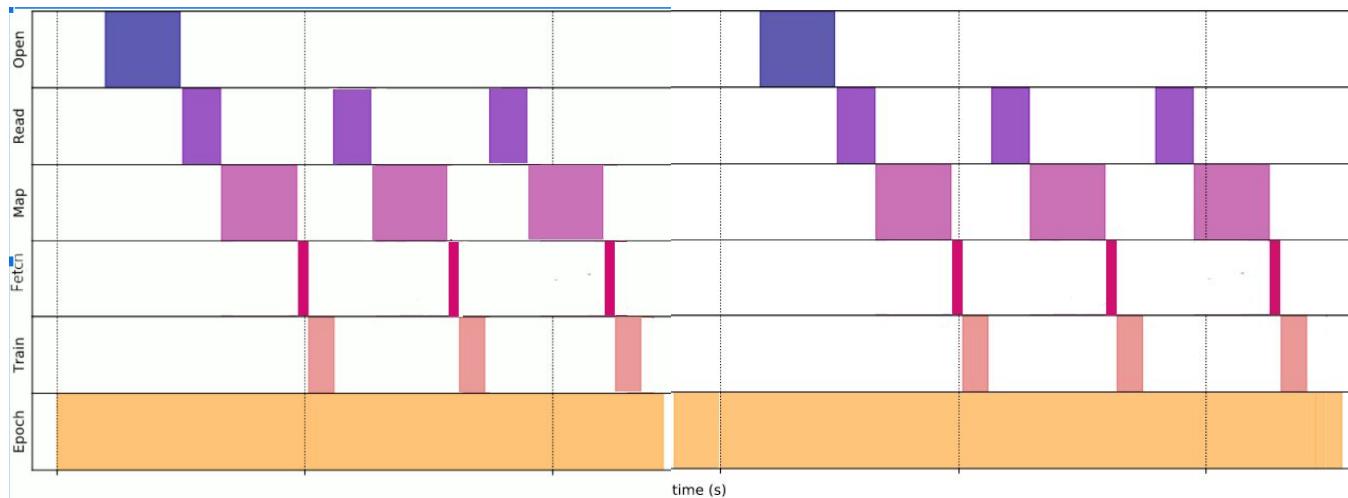
- Need to load/map the data in the first epoch
- The data are kept in the memory for rest of the training epochs



Input pipeline (prefetching)

Prefetching

- overlaps the preprocessing (read, map, fetching) and model execution of a training step



Input pipeline (prefetching) cont.

Prefetching

- overlaps the preprocessing (read, map, fetching) and model execution of a training step

