How to run tf2-cosmoflow on Summit

Sunwoo Lee sunwoolee1.2014@u.northwestern.edu

On Summit, IBM Watson's Machine Learning module should be loaded to use TensorFlow. The module contains TensorFlow, caffe, pyTorch, and other packages for ML applications. However, the module does not support mpi4py. So, the tf2-cosmoflow should be slightly modified to run on Summit GPU nodes without direct calls of mpi4py.

1. Clone the code from github repository.

```
[slz839@login2.summit cosmo]$ git clone https://swblaster@github.com/NU-CUCIS/tf2-cosmoflow
Cloning into 'tf2-cosmoflow'...
Password for 'https://swblaster@github.com':
remote: Enumerating objects: 631, done.
remote: Total 631 (delta 0), reused 0 (delta 0), pack-reused 631
Receiving objects: 100% (631/631), 118.76 KiB | 0 bytes/s, done.
Resolving deltas: 100% (443/443), done.
```

2. Modify ***feeder_async.py.***
    a. Switch line 12 and 13.

```
 1 '''
 2 Copyright (C) 2020, Northwestern University and Lawrence Berkeley National Laboratory
 3 See COPYRIGHT notice in top-level directory.
 4 '''
 5 import os
 6 import time
 7 import tensorflow as tf
 8 import yaml
 9 import numpy as np
10 import h5py
11 import math
12 #from mpi4py import MPI
13 import horovod.tensorflow as hvd
14 import multiprocessing as mp
```

    b. Switch line 22~24 and 25~26.

```
22        #self.comm = MPI.COMM_WORLD
23        #self.size = self.comm.Get_size()
24        #self.rank = self.comm.Get_rank()
25        self.size = hvd.size()
26        self.rank = hvd.rank()
```

c. Comment out line 125.

```
125       #self.comm.Bcast(self.shuffled_file_index, root = 0)
```

3. Modify ***feeder_sync.py*** (same as ***feeder_sync.py***).
   a. Switch line 12 and 13.

```
 1 '''
 2 Copyright (C) 2020, Northwestern University and Lawrence Berkeley National Laboratory
 3 See COPYRIGHT notice in top-level directory.
 4 '''
 5 import os
 6 import time
 7 import tensorflow as tf
 8 import yaml
 9 import numpy as np
10 import h5py
11 import math
12 #from mpi4py import MPI
13 import horovod.tensorflow as hvd
14 import multiprocessing as mp
```

   b. Switch line 23 ~ 25 and 26 ~ 27.

```
22        #self.comm = MPI.COMM_WORLD
23        #self.size = self.comm.Get_size()
24        #self.rank = self.comm.Get_rank()
25        self.size = hvd.size()
26        self.rank = hvd.rank()
```

   c. Comment out line 126.

```
126       #self.comm.Bcast(self.shuffled_file_index, root = 0)
```

4. Modify ***train.py***.
   a. Comment out line 8.

```
1 '''
2 Copyright (C) 2020, Northwestern University and Lawrence Berkeley National Laboratory
3 See COPYRIGHT notice in top-level directory.
4 '''
5 import time
6 import tensorflow as tf
7 import multiprocessing as mp
8 #from mpi4py import MPI
9 import horovod.tensorflow as hvd
```

b. Switch line 101 and 102. Each process will print its local validation loss.

```
101             #average_loss = MPI.COMM_WORLD.allreduce(valid_loss_np, MPI.SUM) /
MPI.COMM_WORLD.Get_size()
102             average_loss = valid_loss_np
```

5. Modify ***main.py***.
    a. Switch line 56 and 57.

```
53    if gpus:
54        # On Summit, each resource set can view its own GPUs only.
55        # So, the visible devices should be set to gpu:0 for every process.
56        tf.config.experimental.set_visible_devices(gpus[0], 'GPU')
57        #tf.config.experimental.set_visible_devices(gpus[hvd.local_rank()], 'GPU')
```

6. Load the IBM Watson module.

```
[slz839@login2.summit tf2-cosmoflow]$ module load ibm-wml-ce/1.7.0-3
(ibm-wml-ce-1.7.0-3) [slz839@login2.summit tf2-cosmoflow]$
```

7. Specify path to the input files and a few parameters in ***test_summit.yaml*** file. The below is an example of the ***test_summit.yaml*** file.

```
 1 frameCnt: 128
 2 numPar: 4
 3 sourceDir: {
 4   prj: /gpfs/alpine/ast153/scratch/slz839/
 5 }
 6 subDir: 1/multiScale_tryG
 7 fnameTmpl: PeterA_2019_05_4parE-rec*.h5
 8 splitIdx:
 9   val: [100, 101, 102, 103, 104, 105, 106, 107]
10   train: [20, 21, 22, 23, 24, 25, 26, 110,
11           30, 31, 32, 33, 34, 35, 36, 111,
12           40, 41, 42, 43, 44, 45, 46, 112,
13           50, 51, 52, 53, 54, 55, 56, 113,
14           60, 61, 62, 63, 64, 65, 66, 114,
15           70, 71, 72, 73, 74, 75, 76, 115,
16           80, 81, 82, 83, 84, 85, 86, 116,
```

8. Submit a job using ***myjob.lsf***. The below is an example script.

```
#!/bin/bash
#BSUB -P AST153
#BSUB -W 00:10
#BSUB -nnodes 11
#BSUB -J sunwoo
#BSUB -o sunwoo.%J
#BSUB -e sunwoo.%J

jsrun -n64 -a1 -c4 -g1 python3 main.py --epochs=3 \
                       --batch_size=4 \
                       --overlap=1 \
                       --checkpoint=0 \
                       --cache_size=0 \
                       --file_shuffle=1 \
                       --buffer_size=8 \
                       --record_acc=0 \
                       --config="test_summit.yaml" \
                       --evaluate=0 \
                       --async_io=1
```

9. The output log looks like the below.

```
Epoch: 0 lr: 0.002
Epoch 1 waiting time = 0 training loss = 1.8676238 training timing: 30.020379295921884 sec
Epoch: 1 lr: 0.002
Epoch 2 waiting time = 0 training loss = 0.31578225 training timing: 9.695642819046043 sec
Epoch: 2 lr: 0.002
Epoch 3 waiting time = 0 training loss = 0.2627585 training timing: 9.665014030993916 sec
All done!
```