WHITE PAPER–JULY 2020

Running TensorFlow on vSphere Bitfusion Example Guide

vSphere Bitfusion Guides

vmware®

Table of Contents

Running TensorFlow on vSphere Bitfusion Example Guide

This document describes the steps to install TensorFlow and benchmarks with all their pre-requisites, and then run the benchmarks on a Bitfusion client. For completeness, it also describes some configuration options for fresh linux VMs and the steps to install Bitfusion client software. These steps should serve as a basis for running other versions of TensorFlow, running other TensorFlow applications and even running entirely different AI/ML applications and frameworks under vSphere Bitfusion.

Introduction

We will describe how to make clients for Ubuntu 16.04, Ubuntu 18.04, RHEL 7, and CentOS 7 VMs, then install TensorFlow and its benchmarks. Some steps are unique to particular Linux distributions, others are the same for all.

The major steps are:

- Create a VM (or select appropriate existing VM)
- Enable VM for Bitfusion
- Install Bitfusion
- Install CUDA 10.0
- Install CuDNN 7
- Install python3, if needed (CentOS)
- Install TensorFlow 1.13.1
- Install TensorFlow benchmarks (branch cnn_tf_v1.13_compatible)
- Run TensorFlow benchmarks

Prerequisites

- We are going to assume that you can mount or copy a directory with an ML dataset. We will use the /data directory for this purpose and assume it has 8 GBs of files from the ImageNet dataset.
- One or more Bitfusion servers need to be up and running

1. Create or select a VM

This is not the proper document to go through the complete procedure for bringing up a new VM. We will, instead, highlight a few choices you should make.

- Select the same switched network used by the Bitfusion servers
- The supported OSes are Ubuntu 16.04, Ubuntu 18.04, RHEL 7, and CentOS 7
- The proper client core count is application-dependent and usually determined with empirical experiments. If you know how many cores it takes to properly run an app with local GPUs, use that number. Bitfusion introduces little compute overhead. If you don't know, start with four and begin exploring.
- The proper amount of client memory is determined by summing the memory size of all the GPUs you want your application to use and then multiplying by 1.5. For example, if you want to use two 16 GB cards, then you would assign a minimum of 48 GB = 1.5 (16GB + 16GB).
- The drive size – 25 GB will suffice for this example flow and leave lots of headroom. This is all subject, of course, to the size of your dataset.
- Configure the network and make sure you can access the Bitfusion appliance

Check the Logical Volume Size

In Ubuntu 18.04, for example, you may, by default, launch a system with a small logical volume (use df -h to see if the space in the root, "/", directory is under 4 GB). You will need more than 4 GB of space for everything you will install. The following steps will make the full disk size available to you.

Your base VM is ready.

BEST PRACTICE: Once you are satisfied with your base VM, take a snapshot so you can recover from any future missteps.

```
# Grow the 3rd partition (see the partition size increase)
sudo parted -l
sudo growpart /dev/sda 3
sudo parted -l

# Grow the physical volume (watch "Free PE" grow)
sudo vgdisplay
sudo pvresize /dev/sda3
sudo vgdisplay

# Extend the root logical volume (watch "LV Size" grow)
sudo lvdisplay
sudo lvresize -l +100%FREE /dev/ubuntu-vg/ubuntu-lv
sudo lvdisplay

# Resize the ext4 file system (watch "Avail" space on / grow) df -h
sudo resize2fs /dev/ubuntu-vg/ubuntu-lv df -h
```

2. Enable Bitfusion

Use vCenter to configure and authorize the client VM:

- Take vCenter to the "Hosts and Clusters" GUI, and on the new client VM(s) rightclick → Bitfusion → select "Enable Bitfusion" In the dialog box, select the "Client" choice then confirm with the "Enable" button

3. Install Bitfusion

As a prerequisite upload the Bitfusion-client deb or rpm package into your VM. You will install the package and add users into the "Bitfusion" group. You can download the bitfusion client software from https://my.vmware.com/group/vmware/get-download?downloadGroup=BITFUSION-200, or from the repositories https://packages.vmware.com/bitfusion/ubuntu/ and https://packages.vmware.com/bitfusion/centos.

A. Ubuntu
(Using Ubuntu 18.04 Bitfusion deb package as an example. The name will be different for 16.04)

```
sudo apt-get update
sudo apt-get install -y  ./bitfusion-client-ubuntu1804_2.0.0-11_amd64.deb

bitfusion version

sudo usermod -aG bitfusion <uname> # Add user to bitfusion group

# log out and back in
# Confirm user, <uname>, belongs to bitfusion group
groups

#test Bitfusion
bitfusion list_gpus
```

B. CentOS 7 and RHEL 7

```
# CentOS only, RHEL does not need the epel-release package
sudo yum install -y epel-release

sudo yum install -y bitfusion-client-centos7-2.0.0-11.x86_64.rpm

bitfusion version
sudo usermod -aG bitfusion <uname> # Add user to bitfusion group

# log out and back in
# Confirm user, <uname>, belongs to bitfusion group
groups

#test Bitfusion
bitfusion list_gpus
```

Continue for all systems (Ubuntu and CentOS and RHEL)

Proceed if the output of the last command, bitfusion list_gpus, looks something like this:

```
- server 0 [172.16.31.222:56001]: running 0 tasks
|- GPU 0: free memory 32510 MiB / 32510 MiB
|- GPU 1: free memory 32510 MiB / 32510 MiB
```

BEST PRACTICE: This is good spot to take another snapshot.

4. Install CUDA 10.0

CUDA is the NVIDIA library allows programmatic access to their GPUs. It will be used by the TensorFlow benchmarks you'll run at the end of this document.

A. Ubuntu 16.04

```
cd <some download directory>

wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-
ubuntu1604_10.0.130-1_amd64.deb

sudo dpkg -i cuda-repo-ubuntu1604_10.0.130-1_amd64.deb
sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/cuda/repos/
ubuntu1604/x86_64/7fa2af80.pub
```

```
sudo apt-get update
sudo apt-get install cuda-10.0

bitfusion run -n 1 nvidia-smi

cd /usr/local/cuda/samples/0_Simple/matrixMul
sudo make
bitfusion run -n 1 ./matrixMul
```

B. Ubuntu 18.04

```
cd <some download directory> wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-
repo-ubuntu1804_10.0.130-1_amd64.deb

sudo dpkg -i cuda-repo-ubuntu1804_10.0.130-1_amd64.deb
sudo apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/7fa2af80.pub

sudo apt-get update
sudo apt-get install cuda-10.0

bitfusion run -n 1 nvidia-smi

cd /usr/local/cuda/samples/0_Simple/matrixMul
sudo make
bitfusion run -n 1 ./matrixMul
```

C. RHEL 7 and CentOS 7

```
cd <some download directory>

wget
https://developer.download.nvidia.com/compute/cuda/repos/rhel7/x86_64/cuda-repo-
rhel7-10.0.130-1.x86_64.rpm
sudo rpm -i cuda-repo-rhel7-10.0.130-1.x86_64.rpm
sudo yum clean all
sudo yum -y install cuda-10-0

bitfusion run -n 1 nvidia-smi

cd /usr/local/cuda/samples/0_Simple/matrixMul
sudo make
bitfusion run -n 1 ./matrixMul

# command in full
cd ~/bitfusion/batch-scripts
bitfusion run -n 1 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
```

BEST PRACTICE: Take a snapshot.

5. Install CuDNN 7

CuDNN is the Deep Neural Network library from NVIDIA. The TensorFlow benchmarks you will run later will require this library too.

As a prerequisite you will need to create an NVIDIA developer account to download onto your VM the CuDNN package matching your CUDA version and appropriate for your OS and version. Go to https://developer.nvidia.com/cudnn

A. Ubuntu (either version 16.04 or 18.04)

You can run the benchmarks by typing out the command in full, with a parameterized convenience script, or with a single-purpose script. Examples shown below.

```
sudo dpkg -i libcudnn7_7.6.5.32-1+cuda10.0_amd64.deb
ldconfig -p | grep cudnn # to see if it is installed
```

B. RHEL 7 and CentOS 7

You can run the benchmarks by typing out the command in full, with a parameterized convenience script, or with a single-purpose script. Examples shown below.

```
sudo rpm -ivh libcudnn7-7.6.5.32-1.cuda10.0.x86_64.rpm
sudo ldconfig # update libraries list
ldconfig -p | grep cudnn # to see if it is installed
```

BEST PRACTICE: Take a snapshot.

6. Install Python3 (CentOS and RHEL)

On RHEL and CentOS systems you may need to install python3. If so, follow the steps below.

A. RHEL 7 and CentOS 7

```
sudo yum update

# Every time you want to use python 3 you have to

# bring up an scl environment with the new python first
sudo yum install -y centos-release-scl
sudo yum install -y rh-python36

# Now `python -V will still report version 2.7. So...
scl enable rh-python36 bash
python -V # will report version 3.6
exit # to get back to outer shell
```

BEST PRACTICE: Take a snapshot.

7. Install pip3 and TensorFlow

We will use pip3 to install TensorFlow. TensorFlow is the ML framework we will be using.

A. Ubuntu (either version 16.04 or 18.04)

```
sudo apt-get install -y python3-pip
sudo pip3 install absl-py
sudo pip3 install tensorflow-gpu==1.13.1
pip3 list
```

B. RHEL 7 and CentOS 7

```
sudo yum install -y python36-devel
sudo yum install -y python36-pip
```

```
# At this point you now have file /usr/bin/python3
# available and you can avoid using scl and just
# invoke python3

sudo pip3 install tensorflow-gpu==1.13.1
pip3 list
```

BEST PRACTICE: Take a snapshot.

8. Install TensorFlow Benchmarks

The benchmarks are open source ML applications designed to test performance on the TensorFlow framework.

We keep things organized working in the subdirectory we set up earlier.

These steps are the same on all our OSes

```
cd ~/bitfusion
git clone https://github.com/tensorflow/benchmarks.git
cd benchmarks
git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/cnn_tf_v1.10_compatible
  ...
  remotes/origin/cnn_tf_v1.13_compatible
  ...
$ git checkout cnn_tf_v1.13_compatible
Branch cnn_tf_v1.13_compatible set up to track remote branch
cnn_tf_v1.13_compatible
from origin.
Switched to a new branch 'cnn_tf_v1.13_compatible'
$ git branch
* cnn_tf_v1.13_compatible
  master
```

BEST PRACTICE: Take a snapshot.

9. Run TensorFlow Benchmarks

Run the benchmarks under Bitfusion. Examples shown below. Again, we are assuming our dataset is in the /data directory

These steps are the same on all the OSes

```
# Run on full GPU, using real data in /data/ dir
cd ~/bitfusion/
bitfusion run -n 1 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
--model=resnet50 \
--variable_update=replicated \
--local_parameter_device=gpu \
--nodistortions \
--num_gpus=1 \
--num_batches=100 \
--data_dir=/data \
--data_name=imagenet \
--use_fp16=False

# Run on 2/3rd-sized partial GPU
# with real data in /data/ dir
cd ~/bitfusion/
bitfusion run -n 1 -p 0.67 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
--model=resnet50 \
--variable_update=replicated \
--local_parameter_device=gpu \
--nodistortions \
--num_gpus=1 \
--num_batches=100 \
--data_dir=/data \
--data_name=imagenet \
--use_fp16=False

# Run with single purpose script using synthesized data
cd ~/bitfusion/
bitfusion run -n 1 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
--model=resnet50 \
--variable_update=replicated \
--local_parameter_device=gpu \
--nodistortions \
--num_gpus=1 \
--num_batches=100 \
--use_fp16=False
```

Last Comments

Now you can run the TensorFlow benchmarks with Bitfusion providing shared GPUs from a remote server. The benchmarks support many models and parameters to help you explore a large space in the machine learning universe. And you now have knowledge that will help you to install and run other frameworks and ML applications.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 vmware.com
Item No: VMW-0520-2213_VMware vSphere Bitfusion_Running TensorFlow Example Guide_1.4_MMT 5/20