**u-net setup on science cloud**

On ubuntu 18.04

1. **Install Cuda 10**

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/**3bf863cc**.pub

Many forums have 7fa2af80.pub as the key, this no longer works.

DW: Yes, indeed, this is what was suggested to me inline, as well:

The public CUDA GPG key does not appear to be installed.

To install the key, run this command:

sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86\_64/**7fa2af80**.pub

DW: Then, this command works (and is required for sudo apt-get update to work):

sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86\_64/**3bf863cc**.pub

sudo apt-get update

sudo apt-get install cuda-10-0

nano ~/.bashrc

In the .bashrc file, insert these lines on the very top:

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/cuda-10.0/lib64

export PATH=$PATH:/usr/local/cuda-10.0/bin

exit and save the file

reload the .bashrc file

source ~/.bashrc

verify installation of CUDA 10.0

nvcc --version

DW:

nvcc: NVIDIA (R) Cuda compiler driver

Copyright (c) 2005-2018 NVIDIA Corporation

Built on Sat\_Aug\_25\_21:08:01\_CDT\_2018

Cuda compilation tools, release 10.0, V10.0.130

1. **Install CudNN7 and NCCL2 (I think NCCL2 isn’t necessary)**

wget https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86\_64/nvidia-machine-learning-repo-ubuntu1804\_1.0.0-1\_amd64.deb

sudo dpkg -i nvidia-machine-learning-repo-ubuntu1804\_1.0.0-1\_amd64.deb

sudo apt update

DW, to actually install the updates:

sudo apt upgrade

sudo apt install -y libcudnn7 libcudnn7-dev libnccl2 libc-ares-dev

sudo mkdir -p /usr/local/cuda-10.0/nccl/lib

sudo ln -s /usr/lib/x86\_64-linux-gnu/libnccl.so.2 /usr/local/cuda/nccl/lib/

sudo ln -s /usr/lib/x86\_64-linux-gnu/libcudnn.so.7 /usr/local/cuda-10.0/lib64/

1. **Follow the ‘Setup on own server (from source)’ part of the u-net tutorial. The following changes apply for cuda 10:**

<https://lmb.informatik.uni-freiburg.de/resources/opensource/unet/>

DW: Specifically, <https://lmb.informatik.uni-freiburg.de/resources/opensource/unet/#installation-backend-fromsource>, there, they say the use the ubuntu username ‘unetuser’ and the host ‘unetserver’. In our (lienkamplab) case, the username is ‘ubuntu’ and the host is some IP address, both are to be changed in this guide to whatever values/names are given by the ubuntu image from the Science Cloud when creating/launching a new instance. The host will become relevant during the fiji unet plugin setup.

sudo apt-get install -y wget git unzip build-essential cmake libboost-system-dev libboost-thread-dev libboost-filesystem-dev libprotobuf-dev protobuf-compiler libhdf5-serial-dev libatlas-base-dev libgoogle-glog-dev

The cmake version you get from this is too old for cuda 10. I uninstalled it with sudo apt remove cmake but you can also just remove it from the command above.

Cmake 3.12.2 is needed for cuda10:

wget https://cmake.org/files/v3.12/cmake-3.12.2.tar.gz

tar -xzvf cmake-3.12.2.tar.gz

cd cmake-3.12.2

sudo ./bootstrap

sudo make

sudo make install

Verify that cmake 3.12.2 is installed

cmake --version

Continue with the tutorial as it is until here:

cmake -DCMAKE\_BUILD\_TYPE=Release -DCMAKE\_INSTALL\_PREFIX=/home/unetuser/u-net -DUSE\_OPENCV=OFF -DUSE\_LEVELDB=OFF -DUSE\_LMDB=OFF -DBUILD\_python=OFF -DBUILD\_python\_layer=OFF -DCUDA\_ARCH\_NAME=Manual -DCUDA\_ARCH\_BIN="20 30 35 50 60 61" -DCUDA\_ARCH\_PTX="35" ..

DW: Be sure to check your username (we used ‘ubuntu’ at the lienkamplab, because that is the username given when choosing Science Cloud’s pre-built Ubuntu images when creating/launching new (virtual computer) instances there):

cmake -DCMAKE\_BUILD\_TYPE=Release -DCMAKE\_INSTALL\_PREFIX=/home/**ubuntu**/u-net -DUSE\_OPENCV=OFF -DUSE\_LEVELDB=OFF -DUSE\_LMDB=OFF -DBUILD\_python=OFF -DBUILD\_python\_layer=OFF -DCUDA\_ARCH\_NAME=Manual -DCUDA\_ARCH\_BIN="30 35 50 60 61 **75**" -DCUDA\_ARCH\_PTX="35" ..

Check the architecture of your GPU here (<https://en.wikipedia.org/wiki/CUDA> (DW: specifically, here: <https://en.wikipedia.org/wiki/CUDA#GPUs_supported>)). You’ll have to add the ‘Compute capability (version)’ to the list in -DCUDA\_ARCH\_BIN="20 30 35 50 60 61" . So in my case I used a Tesla T4 with compute capability 7.5. Also, cuda 10 no longer supports compute capabilities 2.x, so the 20 has to be removed from the list. Resulting in -DCUDA\_ARCH\_BIN="30 35 50 60 61 75". The rest of the cmake command can stay the same.

To edit the ~/.bashrc file the command nano ~/.bashrc can be used again. Instead of starting a new shell, just refresh ~/.bashrc with source ~/.bashrc

**DW**: The last command in the tutorial does not work:

“Test whether execution over ssh works:”

ssh localhost caffe\_unet

ubuntu@localhost: Permission denied (publickey).

“A usage message should appear on the screen.”  
I do not know why this is. I will try to run caffe\_unet from outside the science cloud instance and test the execution over ssh that way (windows powershell):

PS C:\Users\<my\_windows\_username>> ssh -i .\.ssh\id\_rsa\_tower\_sciencecloud ubuntu@172.23.51.207 caffe\_unet

caffe\_unet: command line brew

usage: caffe\_unet <command> <args>

commands:

tiled\_predict score a model in overlap-tile strategy

check\_model\_and\_weights\_h5 check given model (hdf5) and weights

device\_query show GPU diagnostic information

Flags from /home/ubuntu/caffe/tools/caffe\_unet.cpp:

-average\_mirror (Use the average over mirrored versions of the input tiles

for the final segmentation. Cannot be used in conjunction with

…

…

I worked. Therefore, the setup is complete and successful.

1. **fiji u-net plugin**

On first launch, it will ask for the location of caffe\_unet. For me it was

/home/unetuser/u-net/bin/caffe\_unet

DW: Maybe the user being ‘unetuser’ on Taiyo’s instance caused some problems for me when running the fiji unet plugin from my home PC. The appropriate counterpart on my own source-built instance should be:

/home/**ubuntu**/u-net/bin/caffe\_unet

It did (not) work (TBD).