Rahul Bhadani Follow

May 4, 2021 · 4 min read ★ · Listen

Installing cuDNN and CUDA Toolkit on Ubuntu 20.04 for Machine Learning Tasks

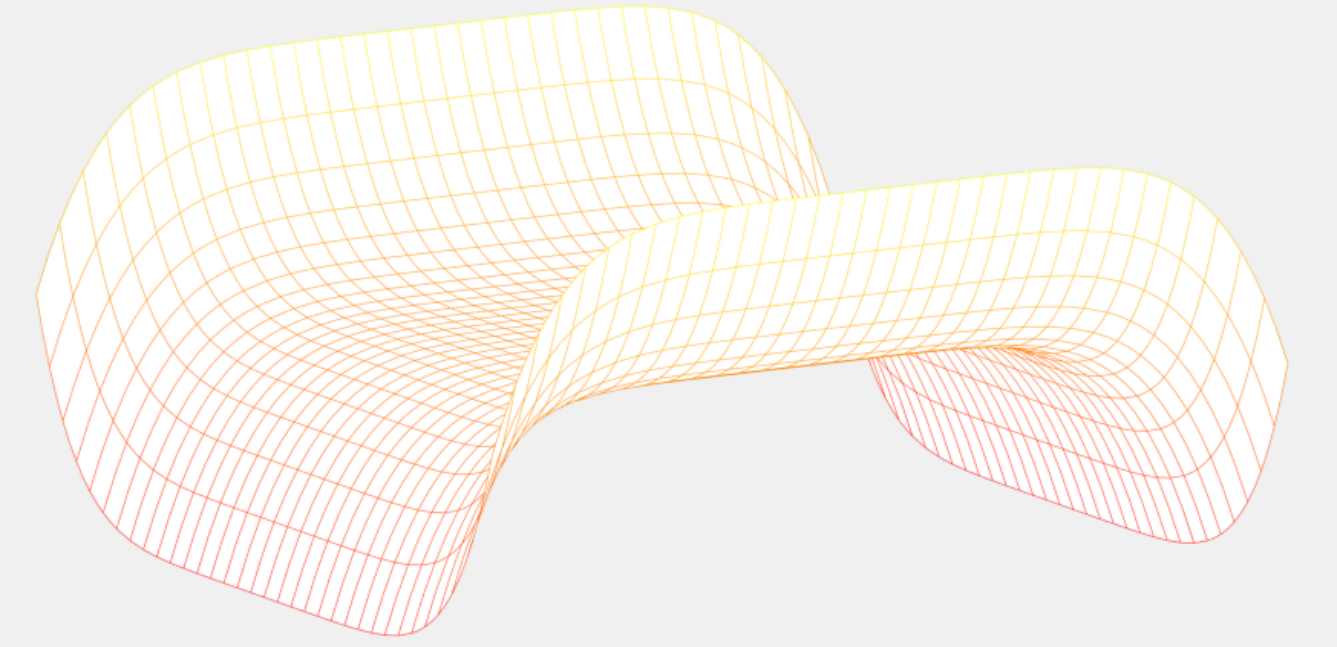

Image by the author. $Z = Y^6 — X^6$

It is always convoluted and challenging to install a CUDA toolkit and library that needs to interact with your NVIDIA GPU on an Ubuntu machine. However, if done right, the CUDA toolkit with your NVIDIA GPU can be a great tool that can harness the power of GPU to produce fast applications.

1650 Ti.

Further, at the time of writing this article, I installed the latest version of the CUDA toolkit which was CUDA Toolkit 11.3.

The other requirement is the NVIDIA developer account that can be obtained at https://developer.nvidia.com/login free of charge.

CUDA Toolkit Installation

I decided to install a .deb file of the CUDA toolkit following the instruction I reproduced below:

```
wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-
pin-600

wget
https://developer.download.nvidia.com/compute/cuda/11.3.0/local_inst
allers/cuda-repo-ubuntu2004-11-3-local_11.3.0-465.19.01-1_amd64.deb
sudo dpkg -i cuda-repo-ubuntu2004-11-3-local_11.3.0-465.19.01-
1_amd64.deb

sudo apt-key add /var/cuda-repo-ubuntu2004-11-3-local/7fa2af80.pub
sudo apt-get update

sudo apt-get -y install cuda
export PATH=/usr/local/cuda-11.3/bin${PATH:+:${PATH}}
export LD_LIBRARY_PATH=/usr/local/cuda-
11.3/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}
```

A few checks can be performed once the CUDA toolkit is installed:

```
systemctl status nvidia-persistenced
```

From its documentation, nvidia-persistenced is intended to be run as a daemon from system initialization and is generally designed as a tool for compute-only platforms

```
sudo systemctl enable nvidia-persistenced
```

To get the version of the NVIDIA driver, type

```
cat /proc/driver/nvidia/version
```

which, in my case, gave output as

```
NVRM version: NVIDIA UNIX x86_64 Kernel Module  465.19.01  Fri Mar
19 07:44:41 UTC 2021
GCC version:  gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
```

Now, we move to download cuDNN, a deep-learning library for writing applications for ML task using NVIDIA GPU:

Installing cuDNN

In order to download cuDNN libraries, you need to go to https://developer.nvidia.com/cudnn and click on the **Download cuDNN** button. The webpage will ask you to login into the NVIDIA developer account. After logging in and accepting their terms and conditions, you should click on the following three links:

cuDNN Runtime Library for Ubuntu20.04 x86_64 (Deb)

cuDNN Developer Library for Ubuntu20.04 x86_64 (Deb)

cuDNN Code Samples and User Guide for Ubuntu20.04 x86_64 (Deb)

which is relevant to Ubuntu 20.04 LTS. After downloading, you should have the following three .deb files:

1. libcudnn8-samples_8.2.0.53–1+cuda11.3_amd64.deb

that you can install using `dpkg` commands:

```
sudo dpkg -i  libcudnn8_8.2.0.53-1+cuda11.3_amd64.deb
sudo dpkg -i libcudnn8-dev_8.2.0.53-1+cuda11.3_amd64.deb
sudo dpkg -i libcudnn8-samples_8.2.0.53-1+cuda11.3_amd64.deb
```

Next, you should type the following to see if your cuDNN communicates with the NVIDIA driver:

```
nvidia-smi
```

If you get an error and a message saying **unable to communicate**, you probably have secure boot enabled that can be turned off by typing:

```
sudo mokutil --disable-validation
```

Typing the above command will ask you to enter a password. Reboot. Then enter the password. **And note down the password.** Usually, the system will ask you to enter specific letters of your password. Once you successfully verify your password, the secure boot will be disabled. Now, reboot your computer and in the terminal, type as follows

```
nvidia-smi
```

and you should see the following output:

```
Fri Apr 30 21:19:53 2021
+-----------------------------------------------------------------------------
----------+
| NVIDIA-SMI 465.19.01    Driver Version: 465.19.01    CUDA Version:
```


```
Compute M. |
|                               |                      |
MIG M. |
|===============================+======================+============
==========|
|   0  NVIDIA GeForce ...  On   | 00000000:01:00.0 Off |
N/A |
| N/A   41C    P8     3W /  N/A |    825MiB /  3914MiB |      7%
Default |
|                               |                      |
N/A |
+-------------------------------+----------------------+------------
----------+

+-----------------------------------------------------------------------
----------+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name
GPU Memory |
|        ID   ID
Usage      |
|=======================================================================
==========|
|    0   N/A  N/A      1084      G   /usr/lib/xorg/Xorg
133MiB |
|    0   N/A  N/A      1720      G   /usr/lib/xorg/Xorg
339MiB |
|    0   N/A  N/A      1892      G   /usr/bin/gnome-shell
68MiB |
|    0   N/A  N/A      3439    C+G   ...R2021a/bin/glnxa64/MATLAB
226MiB |
|    0   N/A  N/A      3720      G   ...1531959360FF99E4245683A9B
5MiB |
|    0   N/A  N/A      4966      G   ...AAAAAAAAA= --shared-files
33MiB |
+-----------------------------------------------------------------------
----------+
```

Testing the installation of cuDNN

To test the installation of cuDNN, copy cuDNN samples into your home directory,

```
cp -r /usr/src/cudnn_sample
```

```
./mnistCUDNN
```

You should some outputs with some texts at the end stating `Tests passed.`

You are now ready to write deep learning applications using NVIDIA and cuDNN.

Reference

1. https://developer.nvidia.com/cuda-downloads?target_os=Linux&target_arch=x86_64&=Ubuntu&target_version=20.04&target_type=deb_local
2. https://developer.nvidia.com/cudnn
3. https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html