

## 1. Graphic driver download (use terminal)

Check your Nvidia graphic driver version : <http://www.nvidia.fr/Download/index.aspx>

if graphic driver version is 375.xx : **sudo apt-get install nvidia-375-dev**

2. CUDA 8.0 download (더욱 자세한 설명을 보고 싶다면 <http://pythonkim.tistory.com/71> 여기로 들어가면 된다. 다만 CUDA, cuDNN 까지만 따라하도록 한다. CUDA의 경우 패치파일이 없으므로 생략하고 환경설정하면 된다.)

<https://developer.nvidia.com/cuda-downloads>

(회원가입 필수)

• Review the CUDA 8 Performance Overview [webinar](#), [Slides](#)

IBM Power8 Users: Download CUDA 8 from the [CUDA Toolkit for IBM Power 8 Page](#).

For Linux users upgrading from previous versions of the CUDA Toolkit, click to see instructions in this section before proceeding.

Select Target Platform ⓘ

Click on the green buttons that describe your target platform. Only supported platforms will be shown.

Operating System

Windows Linux Mac OSX

Architecture ⓘ

x86\_64 ppc64le

Distribution

Fedora OpenSUSE RHEL CentOS SLES Ubuntu

Version

16.04 14.04

Installer Type ⓘ

runfile (local) deb (local) deb (network) cluster (local)

Related Links

[CUDA Quick Start Guide](#)  
[Release Notes](#)  
[EULA](#)  
[Online Documentation](#)  
[CUDA Toolkit Overview](#)  
[Installer Checksums](#)  
[Open Source Packages](#)  
[Legacy CUDA Toolkits](#)

Download Installer for Linux Ubuntu 16.04 x86\_64

The base installer is available for download below.

Base Installer

Download [1.4 GB]

Installation Instructions:

1. Run `sudo sh cuda_8.0.61_375.26_linux.run`  
2. Follow the command-line prompts

The CUDA Toolkit contains Open-Source Software. The source code can be found [here](#).  
The checksums for the installer and patches can be found in [Installer Checksums](#).  
For further information, see the [Installation Guide for Linux](#) and the [CUDA Quick Start Guide](#).

터미널창에 다음과 같이 입력한다. (터미널은 다운받은 run file 위치에서 실행)

```
$ sudo sh cuda_8.0.61_375.26_linux.run --override
```

다음과 같은 설치과정에서 빨간색 글씨처럼 입력한다.

```
# ----- 설치 내용 ----- #
```

```
Do you accept the previously read EULA? accept/decline/quit: accept
```

```
Install NVIDIA Accelerated Graphics Driver for Linux-x86_64 361.77? (y)es/(n)o/(q)uit: n
```

```
Install the CUDA 8.0 Toolkit? (y)es/(n)o/(q)uit: y
```

```
Enter Toolkit Location [ default is /usr/local/cuda-8.0 ]: 엔터
```

```
Do you want to install a symbolic link at /usr/local/cuda? (y)es/(n)o/(q)uit: y
```

```
Install the CUDA 8.0 Samples? (y)es/(n)o/(q)uit: y
```

```
Enter CUDA Samples Location [ default is /home/python-kim ]: 엔터
```

Installing the CUDA Toolkit in /usr/local/cuda-8.0 ...

Missing recommended library: libGLU.so.

Missing recommended library: libX11.so

Missing recommended library: libXi.so

Missing recommended library: libXmu.so

Installing the CUDA Samples in /home/python-kim ...

Copying samples to /home/python-kim/NVIDIA\_CUDA-8.0\_Samples now...

Finished copying samples.

===== Summary =====

Driver: Not Selected

Toolkit: Installed in /usr/local/cuda-8.0

Samples: Installed in /home/python-kim, but missing recommended libraries

이렇게 되면 설치 완료!

그 후 환경구성을 해야한다.

라이브러리와 CUDA 를 사용할 수 있도록 경로를 추가한다. 먼저 환경 파일을 연다.

**\$ sudo gedit ~/.bashrc**

아래 내용은 .bashrc 파일의 마지막에 추가한다.

**export CUDA\_HOME=/usr/local/cuda-8.0**

**export PATH=/usr/local/cuda-8.0/bin\${PATH:+:\${PATH}}**

**export**

**LD\_LIBRARY\_PATH=/usr/local/cuda-8.0/lib64\${LD\_LIBRARY\_PATH:+:\${LD\_LIBRARY\_PATH}}**

Gedit 에서 저장 후 gedit 을 종료한다.

추가한 내용을 즉각 반영한다.

**\$ source ~/.bashrc**

환경설정까지 완료 그 후 sample 을 구동해서 정상적으로 CUDA 가 설치되었는지 확인한다.

**\$ cd NVIDIA\_CUDA-8.0\_Samples/1\_Uutilities/bandwidthTest/**

**\$ make**

```
"/usr/local/cuda-8.0"/bin/nvcc -ccbin g++ -I../common/inc -m64 -gencode
arch=compute_20,code=sm_20 -gencode arch=compute_30,code=sm_30 -gencode
arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm_37 -gencode
arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52 -gencode
arch=compute_60,code=sm_60 -gencode arch=compute_60,code=compute_60 -o
bandwidthTest.o -c bandwidthTest.cu
"/usr/local/cuda-8.0"/bin/nvcc -ccbin g++ -m64 -gencode
arch=compute_20,code=sm_20 -gencode arch=compute_30,code=sm_30 -gencode
arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm_37 -gencode
arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52 -gencode
arch=compute_60,code=sm_60 -gencode arch=compute_60,code=compute_60 -o
bandwidthTest bandwidthTest.o
mkdir -p ../bin/x86_64/linux/release
cp bandwidthTest ../bin/x86_64/linux/release
```

**\$ ./bandwidthTest**

[CUDA Bandwidth Test] - Starting...

Running on...

Device 0: GeForce GTX 1060 6GB

Quick Mode

Host to Device Bandwidth, 1 Device(s)

PINNED Memory Transfers

Transfer Size (Bytes)	Bandwidth(MB/s)
33554432	12542.1

Device to Host Bandwidth, 1 Device(s)

PINNED Memory Transfers

Transfer Size (Bytes)	Bandwidth(MB/s)
33554432	12322.1

Device to Device Bandwidth, 1 Device(s)

PINNED Memory Transfers

Transfer Size (Bytes)	Bandwidth(MB/s)
33554432	141467.7

Result = PASS

NOTE: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost is enabled.

다음과 같이 나오면 성공한 것이다.

### 3.CUDNN 5.1 download

<https://developer.nvidia.com/rdp/cudnn-download> 회원가입 필수

## cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☒ I Agree To the Terms of the [cuDNN Software License Agreement](#)

Please check your framework documentation to determine the recommended version of cuDNN.  
If you are using cuDNN with a Pascal (GTX 1080, GTX 1070), version 5 or later is required.

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 8.0

[cuDNN User Guide](#)

[cuDNN Install Guide](#)

[cuDNN v5.1 Library for Linux](#)

[cuDNN v5.1 Library for Power8](#)

[cuDNN v5.1 Library for Windows 7](#)

[cuDNN v5.1 Library for Windows 10](#)

[cuDNN v5.1 Library for OSX](#)

[cuDNN v5.1 Release Notes](#)

#### QUICKLINKS

[Accelerated Computing - Training](#)

[CUDA GPUs](#)

[Tools & Ecosystem](#)

[OpenACC: More Science Less Programming](#)

[CUDA FAQ](#)

#### GPU Computing

[Follow](#)

CUDA, GPU Computing Retweeted

 **Mark Harris**  
@harris

[buff.ly/2mDkLm](#) Full details on Jetson TX2 Developer Kit. 2x [#deeplearning](#) efficiency for Edge computing.



cuDNN v5.1 library for Linux 를 받으면 된다.

이건 설치할 게 없다. 다운로드한 파일을 압축을 풀어서 복사해서 붙여넣기만 하면 된다.  
앞에서 CUDA 샘플을 구동하기 위해 샘플 폴더로 이동했기 때문에 홈 폴더로 이동하는 것까지 포함한다.

```
$ cd ~
```

```
$ tar xvzf cudnn-8.0-linux-x64-v5.1.tgz
```

```
$ sudo cp cuda/include/cudnn.h /usr/local/cuda-8.0/include/
```

```
$ sudo cp cuda/lib64/* /usr/local/cuda-8.0/lib64/
```

다음과 같이 터미널에서 순차적으로 입력하면 된다.

### 4.Pip download(use terminal)

#### Pip3 download(use terminal)

Pip 과 pip3 은 각각 python2.xx 버전과 python3.xx 버전이라고 보면 된다.

Python2.xx 버전과 python3.xx 버전 모두를 받는것이 좋을 것 같다.

파이썬 3.5

```
$ sudo apt-get install python3-pip python3-numpy swig python3-dev python3-wheel
```

파이썬 2.7

```
$ sudo apt-get install python-pip python-numpy swig python-dev python-wheel
```

### 6.Tensorflow download(use terminal)

파이썬 3.5

```
$ sudo pip install tensorflow-gpu
```

파이썬 2.7

```
$ sudo pip3 install tensorflow-gpu
```

다 완료했으면 terminal 에서

\$ python

\$ import tensorflow

```
luislee@luislee-Precision-WorkStation-T5500: ~  
luislee@luislee-Precision-WorkStation-T5500:~$ pip3 install urllib  
Collecting urllib  
  Could not find a version that satisfies the requirement urllib (from versions: )  
No matching distribution found for urllib  
luislee@luislee-Precision-WorkStation-T5500:~$ pip3 install xrange  
Collecting xrange  
  Could not find a version that satisfies the requirement xrange (from versions: )  
No matching distribution found for xrange  
luislee@luislee-Precision-WorkStation-T5500:~$ python  
Python 2.7.12 (default, Nov 19 2016, 06:48:10)  
[GCC 5.4.0 20160609] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import tensorflow  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcublas.so.8.0 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcudnn.so.5 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcufft.so.8.0 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcuda.so.1 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcurand.so.8.0 locally  
>>>
```

\$ python3

\$ import tensorflow

```
luislee@luislee-Precision-WorkStation-T5500: ~  
luislee@luislee-Precision-WorkStation-T5500:~$ python3  
Python 3.5.2 (default, Nov 17 2016, 17:05:23)  
[GCC 5.4.0 20160609] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import tensorflow  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcublas.so.8.0 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcudnn.so.5 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcufft.so.8.0 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcuda.so.1 locally  
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcurand.so.8.0 locally  
>>>
```

다음과 같이 나오면 된다.