

Learn OpenCV

Installing Deep Learning Frameworks on Ubuntu with CUDA support

SEPTEMBER 11, 2017 BY [VIKAS GUPTA \(HTTPS://WWW.LEARNOPENCV.COM/AUTHOR/VIKAS/\)](https://www.learnopencv.com/author/vikas/)



</wp-content/uploads/2017/09/feature-image-dl-installation-e1505140356872.jpg>

In this article, we will learn how to install Deep Learning Frameworks like TensorFlow, Theano, Keras and PyTorch on a machine having a NVIDIA graphics card.

If you have a brand new computer with a graphics card and you don't know what libraries to install to start your deep learning journey, this article will help you.

We will install CUDA, cuDNN, Python 2, Python 3, TensorFlow, Theano, Keras, Pytorch,

Processor : Intel core i7 6850K with 6 cores and 40 PCIe lines

Motherboard : Gigabyte X99P – SLI

RAM : 32 GB

Graphics Card : Zotac GeForce GTX 1080 Ti with 11 GB RAM

We will be assuming a fresh Ubuntu 16.04 installation. i.e nothing has been installed on the system earlier.

Step 1 : Install Prerequisites

Before installing anything, let us first update the information about the packages stored on the computer and upgrade the already installed packages to their latest versions.

```
1 | sudo apt-get update
2 | sudo apt-get upgrade
```

Next, we will install some basic packages which we might need during the installation process as well in future. Also, remove the packages which are not needed.

```
1 | sudo apt-get install -y build-essential cmake gfortran git pkg-config
2 | sudo apt-get install -y python-dev software-properties-common wget vim
3 | sudo apt-get autoremove
```

Step 2 : Install CUDA

If you do not have a NVIDIA CUDA supported Graphics Card, then you can skip this step. and go to Step 4.

Download the CUDA driver from the [official nvidia website](https://developer.nvidia.com/cuda-downloads) (<https://developer.nvidia.com/cuda-downloads>). We recommend you download the deb (local) version from Installer type as shown in the screenshot below.

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 \(local\)](#)

Download Installers for Linux Ubuntu 16.04 x86_64

The base installer is available for download below.

There is 1 patch available. This patch requires the base installer to be installed first.

> Base Installer

[Download \(1.9 GB\) !\[\]\(654d8e30dc2e8e002b21c7dff500ad96_img.jpg\)](#)

Installation Instructions:

1. ``sudo dpkg -i cuda-repo-ubuntu1604-8-0-local-ga2_8.0.61-1_amd64.deb``
2. ``sudo apt-get update``
3. ``sudo apt-get install cuda``

[\(/wp-content/uploads/2017/09/nvidia-download-page.png\)](/wp-content/uploads/2017/09/nvidia-download-page.png)

After downloading the file, go to the folder where you have downloaded the file and run the following commands from the terminal to install the CUDA drivers.

Please make sure that the filename used in the command below is the same as the downloaded file.

```
1 | sudo dpkg -i cuda-repo-ubuntu1604-8-0-local-ga2_8.0.61-1_amd64.deb
2 | sudo apt-get update
3 | sudo apt-get install -y cuda-8.0
```

Run the following command to check whether the driver has installed successfully by running NVIDIA's System Management Interface (nvidia-smi). It is a tool used for monitoring the state of the GPU.

```
1 | nvidia-smi
```

You should get an output as shown below.

```
@vikas-gpu: ~
vikas@vikas-gpu:~$ nvidia-smi
Mon Sep 11 13:01:09 2017

+-----+-----+
| NVIDIA-SMI 375.82                Driver Version: 375.82          |
+-----+-----+
| GPU   Name           Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|=====+=====+
|   0   GeForce GTX 108...    Off   | 0000:4B:00.0     On   |           N/A       |
|  0%   32C    P8      13W / 320W | 288MiB / 11171MiB |           0%      Default |
+-----+-----+

+-----+-----+
| Processes:                        GPU Memory Usage |
| GPU       PID    Type    Process name                  |
|=====+=====+
|   0        972    G       /usr/lib/xorg/Xorg              | 178MiB |
|   0       1792    G       compiz                          |  43MiB |
+-----+-----+
```

```
| 0 2158 G ...el-token=3B7E63559B0772E7093383DC4E9A535C 64MiB |
+-----+
vikas@vikas-gpu:~$
```


(/wp-content/uploads/2017/09/nvidia-smi-output.png)

As a side note, I found that apart from getting better resolution options for display, installing the CUDA driver lowers the power consumption of the graphics card from 71W to 16W for a NVIDIA GTX 1080 Ti GPU attached via PCIe x16.

Step 3 : Install cuDNN

CUDA Deep Neural Network (cuDNN) is a library used for further optimizing neural network computations. It is written using the CUDA API.

Go to [official cudnn website \(https://developer.nvidia.com/rdp/form/cudnn-download-survey\)](https://developer.nvidia.com/rdp/form/cudnn-download-survey) and fill out the form for downloading the cuDNN library. After you get to the download link (sample shown below), you should download the “cuDNN v6.0 Library for Linux” from the options.

 **NVIDIA DEVELOPER** COMPUTEWORKS GAMEWORKS JETPACK DESIGNWORKS DRIVE vikas ▾

Home > ComputeWorks > Deep Learning > Software > cuDNN Download

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 GPU, version 5 or later is required.

For access to cuDNN user guide, API reference and release notes, please visit the [cuDNN product documentation webpage](#).

[Download cuDNN v7.0.2 \(Sept 7, 2017\), for CUDA 9.0 RC](#)

[Download cuDNN v7.0.2 \(Sept 7, 2017\), for CUDA 8.0](#)

[Download cuDNN v6.0 \(April 27, 2017\), for CUDA 8.0](#)

Download packages updated April 27, 2017 to resolve issues related to dilated convolution on Kepler Architecture GPUs.

[cuDNN User Guide](#)[cuDNN Install Guide](#)[cuDNN v6.0 Library for Linux](#)[cuDNN v6.0 Library for Power8](#)[cuDNN v6.0 Library for Windows 7](#)[cuDNN v6.0 Library for Windows 10](#)[cuDNN v6.0 Library for OSX](#)[cuDNN v6.0 Release Notes](#)[cuDNN v6.0 Runtime Library for Ubuntu16.04 \(Deb\)](#)

[\(/wp-content/uploads/2017/09/cudnn-download-page.png\)](#)

Now, go to the folder where you have downloaded the “.tgz” file and from the command line execute the following.

```
1 | tar xvf cudnn-8.0-linux-x64-v6.0.tgz
2 | sudo cp -P cuda/lib64/* /usr/local/cuda/lib64/
3 | sudo cp cuda/include/* /usr/local/cuda/include/
```

Next, update the paths for CUDA library and executables.

```
1 | echo 'export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/usr/local/cuda/lib64:/usr/local/cuda/ex'
2 | echo 'export CUDA_HOME=/usr/local/cuda' >> ~/.bashrc
3 | echo 'export PATH="/usr/local/cuda/bin:$PATH"' >> ~/.bashrc
4 | source ~/.bashrc
```

This should get everything sorted out with respect to CUDA and cuDNN

Step 4 : Install requirements for DL Frameworks

NOTE : If you get a warning saying
/usr/lib/nvidia-375/libEGL.so.1 not a symbolic link
Then execute the following commands.

```
1 sudo mv /usr/lib/nvidia-375/libEGL.so.1 /usr/lib/nvidia-375/libEGL.so.1.org
2 sudo mv /usr/lib32/nvidia-375/libEGL.so.1 /usr/lib32/nvidia-375/libEGL.so.1.org
3 sudo ln -s /usr/lib/nvidia-375/libEGL.so.375.82 /usr/lib/nvidia-375/libEGL.so.1
4 sudo ln -s /usr/lib32/nvidia-375/libEGL.so.375.82 /usr/lib32/nvidia-375/libEGL.so.1
```

Next, we install python 2 and 3 along with other important packages like boost, lmbd, glog, blas etc.

```
1 sudo apt-get install -y --no-install-recommends libboost-all-dev doxygen
2 sudo apt-get install -y libgflags-dev libgoogle-glog-dev liblmbd-dev libblas-dev
3 sudo apt-get install -y libatlas-base-dev libopenblas-dev libgphoto2-dev libeigen3-dev
4
5 sudo apt-get install -y python-dev python-pip python-nose python-numpy python-scipy
6 sudo apt-get install -y python3-dev python3-pip python3-nose python3-numpy python3-scipy
```

Step 5 : Enable Virtual Environments

Most of us work on different projects and like to keep the settings for these projects separate too. This can be done using Virtual environments in Python. In a virtual environment, you can install any python library without affecting the global installation or other virtual environments. This way, even if you damage the libraries in one virtual environment, your rest of the projects remain safe. It is highly recommended to use virtual environments.

Install the virtual environment wrapper which enables us to create and work on virtual environments in python.

Step 6 : Install Deep Learning frameworks

Now, we install Tensorflow, Keras, PyTorch, dlib along with other standard Python ML libraries like numpy, scipy, sklearn etc.

We will create virtual environments and install all the deep learning frameworks inside them. We create separate environments for Python 2 and 3.

NOTE that PyTorch is in beta at the time of writing this article. So, the download link for PyTorch can change in future. You can visit [this link \(http://pytorch.org/\)](http://pytorch.org/) to get the correct download link according to your desktop configuration.

For Python 2

```
1 # create a virtual environment for python 2
2 mkvirtualenv virtual-py2 -p python2
3 # Activate the virtual environment
4 workon virtual-py2
5
6 pip install numpy scipy matplotlib scikit-image scikit-learn ipython protobuf jupyter
7
8 # If you do not have CUDA installed
9 pip install tensorflow
10 # If you have CUDA installed
11 pip install tensorflow-gpu
12
13 pip install Theano
14 pip install keras
15 pip install http://download.pytorch.org/whl/cu80/torch-0.2.0.post3-cp27-cp27mu-manylini
16 pip install dlib
17
18 deactivate
```

For Python 3


```

/
8 # If you do not have CUDA installed
9 pip install tensorflow
10 # If you have CUDA installed
11 pip install tensorflow-gpu
12
13 pip install Theano
14 pip install keras
15 pip install http://download.pytorch.org/whl/cu80/torch-0.2.0.post3-cp35-cp35m-manylinux
16 pip install dlib
17
18 deactivate

```

Check Installation of Frameworks

```

1 workon virtual-py2
2 python
3 import numpy
4 numpy.__version__
5 import theano
6 theano.__version__
7 import tensorflow
8 tensorflow.__version__
9 import keras
10 keras.__version__
11 import torch
12 torch.__version__
13 import cv2
14 cv2.__version__

```

You should get an output similar to the figure shown below

```

(virtual-py2) vikas@vikas-gpu:~/softwares/opencv/build$ clear
(virtual-py2) vikas@vikas-gpu:~/softwares/opencv/build$ 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 numpy
>>> numpy.__version__
'1.13.1'
>>> import theano
>>> theano.__version__
'0.9.0'
>>> import tensorflow
>>> tensorflow.__version__
'1.3.0'
>>> import keras
Using TensorFlow backend.
>>> keras.__version__
'2.0.8'
>>> import torch
>>> torch.__version__
'0.2.0_3'
>>> import cv2
Traceback (most recent call last):

```

```
File "<stdin>", line 1, in <module>
ImportError: No module named cv2
>>>
```

[\(/wp-content/uploads/2017/09/library-versions.png\)](/wp-content/uploads/2017/09/library-versions.png)

If you want to install OpenCV 3.3, follow along

Step 7 : Install OpenCV 3.3

First we will install the dependencies

```
1  sudo apt-get remove x264 libx264-dev
2  sudo apt-get install -y checkinstall yasm
3  sudo apt-get install -y libjpeg8-dev libjasper-dev libpng12-dev
4
5  # If you are using Ubuntu 14.04
6  sudo apt-get install -y libtiff4-dev
7
8  # If you are using Ubuntu 16.04
9  sudo apt-get install -y libtiff5-dev
10 sudo apt-get install -y libavcodec-dev libavformat-dev libswscale-dev libdc1394-22-dev
11
12 sudo apt-get install -y libxine2-dev libv4l-dev
13 sudo apt-get install -y libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev
14 sudo apt-get install -y libqt4-dev libgtk2.0-dev libtbb-dev
15 sudo apt-get install -y libfaac-dev libmp3lame-dev libtheora-dev
16 sudo apt-get install -y libvorbis-dev libxvidcore-dev
17 sudo apt-get install -y libopencore-amrnb-dev libopencore-amrwb-dev
18 sudo apt-get install -y x264 v4l-utils
```

Download OpenCV and OpenCV-contrib

```
1  git clone https://github.com (http://github.com)/opencv/opencv.git
2  cd opencv
3  git checkout 3.3.0
4  cd ..
```

Configure and generate the Makefile

```

1 | cd opencv
2 | mkdir build
3 | cd build
4 | #Remove the line WITH_CUDA=ON if you dont have CUDA in your system
5 |
6 | cmake -D CMAKE_BUILD_TYPE=RELEASE \
7 |       -D CMAKE_INSTALL_PREFIX=/usr/local \
8 |       -D INSTALL_C_EXAMPLES=ON \
9 |       -D INSTALL_PYTHON_EXAMPLES=ON \
10 |      -D WITH_TBB=ON \
11 |      -D WITH_V4L=ON \
12 |      -D WITH_QT=ON \
13 |      -D WITH_OPENGL=ON \
14 |      -D WITH_CUDA=ON \
15 |      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib/modules \
16 |      -D BUILD_EXAMPLES=ON ..

```

Compile and Install

NOTE : *The make operation takes quite a long time, almost an hour using 12 cores on an i7 processor. Also, it might get stuck for long at some places, but don't worry unless it is stuck for more than an hour.*

```

1 | make -j4
2 | sudo make install
3 | sudo sh -c 'echo "/usr/local/lib" >> /etc/ld.so.conf.d/opencv.conf'
4 | sudo ldconfig

```

Link OpenCV to your virtual environments

```

1 | # Link the opencv to python virtual environment
2 | find /usr/local/lib/ -type f -name "cv2*.so"

```

It should give an output similar to the one shown below

```
/usr/local/lib/python3.5/dist-packages/cv2.cpython-35m-x86_64-linux-gnu.so
```

```
/usr/local/lib/python2.7/dist-packages/cv2.so
```

Note the exact path of the cv2.so file. In my system, it is located in dist-packages. But in most systems, it is located in site-packages directory.

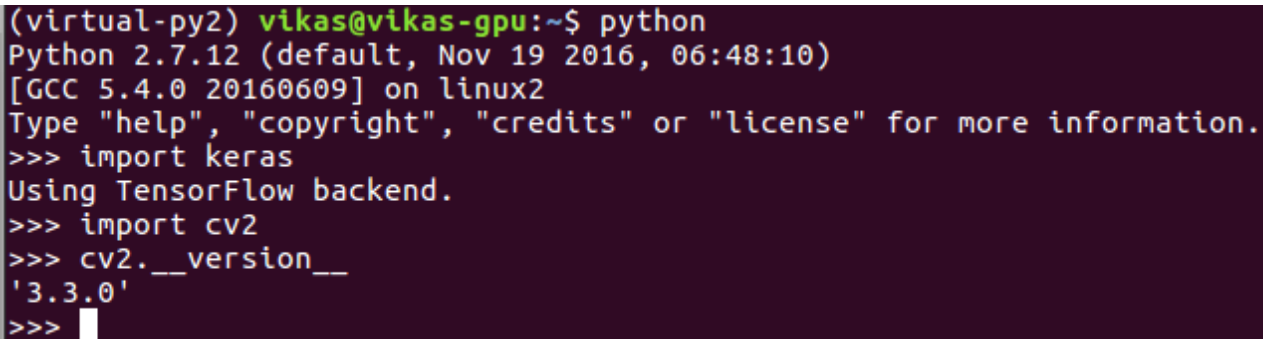
The creation of symlinks is done as follows

```
1 ##### For Python 2 #####
2 cd ~/.virtualenvs/virtual-py2/lib/python2.7/site-packages
3 ln -s /usr/local/lib/python2.7/dist-packages/cv2.so cv2.so
4
5 ##### For Python 3 #####
6 cd ~/.virtualenvs/virtual-py3/lib/python3.5/site-packages
7 ln -s /usr/local/lib/python3.5/dist-packages/cv2.cpython-35m-x86_64-linux-gnu.so cv2.so
```

Check OpenCV Installation

```
1 workon virtual-py2
2 python
3 import cv2
4 cv2.__version__
```

You should get an output similar to the figure given below

A terminal window with a dark background and light-colored text. The prompt is (virtual-py2) vikas@vikas-gpu:~\$. The user enters 'python', and the terminal shows 'Python 2.7.12 (default, Nov 19 2016, 06:48:10) [GCC 5.4.0 20160609] on linux2'. It then shows 'Type "help", "copyright", "credits" or "license" for more information.' followed by a series of prompts '>>>' where the user enters 'import keras', 'import cv2', and 'cv2.__version__'. The output for the last command is '3.3.0'.

```
(virtual-py2) vikas@vikas-gpu:~$ 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 keras
Using TensorFlow backend.
>>> import cv2
>>> cv2.__version__
'3.3.0'
>>> █
```

[\(/wp-content/uploads/2017/09/cv2-version.png\)](/wp-content/uploads/2017/09/cv2-version.png)

What next?

Subscribe & Download Code

If you liked this article and would like to download code (C++ and Python) and example images used all the posts of this blog, please [subscribe](#)

(<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>) to our newsletter. You will also receive a free [Computer Vision Resource](#) (<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>) Guide. In our newsletter, we share OpenCV tutorials and examples written in C++/Python, and Computer Vision and Machine Learning algorithms and news.

Subscribe Now

(<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>)

References

Here is a list of other resources you may find useful.

<https://github.com/floydhub/dl-setup> (<https://github.com/floydhub/dl-setup>)

http://www.born2data.com/2017/deeplearning_install-part1.html

(http://www.born2data.com/2017/deeplearning_install-part1.html)

(<http://www.pyimagesearch.com/2016/10/24/ubuntu-16-04-how-to-install-opencv>)



Join the discussion...

LOG IN WITH



OR SIGN UP WITH DISQUS ?

Name



Nimrod • 9 days ago

Thanks for the article! After installing tensorflow, I opened the terminal and typed "
python", afterwards I typed "import tensorflow as tf" and got the following error:

Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named tensorflow.
Same with keras.

Any idea what's the problem and how to solve it?

On tensorflow's site they say to install:
\$ sudo apt-get install libcupti-dev
(I haven't found it in this article. Is it a must for using gpu?)

^ | v • Reply • Share >



Haran Rajkumar • 18 days ago

Thanks for the article! Very well made.

I have a doubt. When I'm installing the packages in the virtual environments, it's redownloading them. Is it normal? Does that mean, it stores a different copy of the package for every virtual environment?

^ | v • Reply • Share >



Vikas Gupta → Haran Rajkumar • 16 days ago

Yes, thats how it is done. Otherwise what's the point of having different environments?

^ | v • Reply • Share >



Hash • 3 months ago

Thanks alot man! I have been trying to install keras for three days now but this guide atleast made it possible.
one thing though, in this command

sudo apt-get install -y cuda

here you should change 'cuda' with 'cuda-8.0', otherwise it just installs cuda 9.0 (latest version) which i think is not flexible with TF.

Best regards.

^ | v • Reply • Share >

COPYRIGHT © 2018 · BIG VISION LLC