

Install Keras in 35 minutes

Md. Zahiduzzaman

Senthilkumar Sockalingam Kathiresan

Arka Mallick

Step 1. If the XMG machine does not have any OS or has arch linux then install own ubuntu. We installed 16.04.3 LTS (Xenial Xerus) (<http://releases.ubuntu.com/16.04/>) using a bootable usb. We used Rufus(<https://rufus.akeo.ie/>) to create the bootable usb from the ubuntu disk image. Further details on how to install ubuntu is outside the scope of this tutorial. But very *important* is when you boot the machine from usb for installing ubuntu, you might have to *force* boot from usb . For that you have to press **F7** or **F12** when the first screen comes up. Then go to the option of boot from USB.

Step 2. Now that the ubuntu is installed, we will go straight to installing anaconda from following the website (<https://conda.io/docs/user-guide/install/linux.html>)

Download the “Anaconda installer for Linux” then use the following command after doing cd into the downloaded folder

```
bash Anaconda-latest-Linux-x86_64.sh
```

Do not have to install visual studio if you do not need it.

Step 3: *lspci | grep -i nvidia*

to check which NVIDIA chipset which the machine has.

Step 4: Install the required linux-headers

```
sudo apt-get install linux-headers-$(uname -r)
```

Step 5: Download and install CUDA(9.0)

https://developer.nvidia.com/cuda-90-download-archive?target_os=Linux&target_arch=x86_64&target_distro=Ubuntu&target_version=1604

CUDA Toolkit 9.0 Downloads

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	17.04	16.04				
Installer Type ⓘ	runfile [local]	deb [local]	deb [network]	cluster [local]		

Then select and download the *deb(local)*

```
sudo dpkg -i cuda-repo-ubuntu1604-9-0-local_9.0.176-1_amd64.deb
sudo apt-key add /var/cuda-repo-9-0-local/7fa2af80.pub (Follow the CUDA suggestion on your screen instead of this one)
sudo apt-get update
sudo apt-get install cuda
```

Step 6: Download the related patch for the CUDA driver and lets install it

```
sudo dpkg -i cuda-repo-ubuntu1604-9-0-local-cublas-performance-update_1.0-1_amd64.deb
```

Step 7: /usr/local/cuda-9.0/extras/demo_suite/deviceQuery

For detecting available cuda enabled device is present or not. If not available then try after restart.

Step 8: Adding the environment variables.

```
export PATH=/usr/local/cuda-9.0/bin${PATH:+:${PATH}}
echo $PATH
export LD_LIBRARY_PATH=/usr/local/cuda-9.0/lib64
echo $LD_LIBRARY_PATH
```

Step 9: restart laptop

Step 10: cd ~/Downloads/

Step 11: Install CUDNN: You may need to register to be able to download the required file. (<https://developer.nvidia.com/rdp/cudnn-download>)

Download cuDNN v7.0.5 (Dec 5, 2017), for CUDA 9.0

Select the compatible option for the CUDA version. Then download runtime , developer library and code samples all three

[cuDNN v7.0.5 Runtime Library for Ubuntu16.04 \(Deb\)](#)

[cuDNN v7.0.5 Developer Library for Ubuntu16.04 \(Deb\)](#)

[cuDNN v7.0.5 Code Samples and User Guide for Ubuntu16.04 \(Deb\)](#)

```
sudo dpkg -i libcudnn7_7.0.5.15-1+cuda9.0_amd64.deb
sudo dpkg -i libcudnn7-dev_7.0.5.15-1+cuda9.0_amd64.deb
sudo dpkg -i libcudnn7-doc_7.0.5.15-1+cuda9.0_amd64.deb
```

```
sudo apt-get install cuda-command-line-tools
sudo apt-get install cuda-command-line-tools-9-0
```

Set the path variable:

```
export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/extras/CUPTI/lib64
```

Step 12: Creating a Anaconda virtual environment so that all the settings for Keras and Tensorflow remain independent of the rest of the python setup.

```
conda create -n tensorflow pip python=3.6 jupyter
```

```
source activate tensorflow
```

Step 13: Install **Tensorflow**: it installs Tensorflow inside the virtual environment if it is still active.

```
pip install --ignore-installed --upgrade
https://storage.googleapis.com/tensorflow/linux/gpu/tensorflow_gpu-1.5.0-
cp36-cp36m-linux_x86_64.whl
```

```
sudo apt-get update
```

Step 14: Install **Keras** inside the virtual environment

```
sudo pip install keras
```

Step 15: Restart the system

Step 16: Time to test out the Keras. Follow the instructions from the website <https://keras.io/>

After restart need to activate the environment every time before using the Keras. (I.e.)

source activate tensorflow

Hope things work out and may be able to save some time. Cheers!