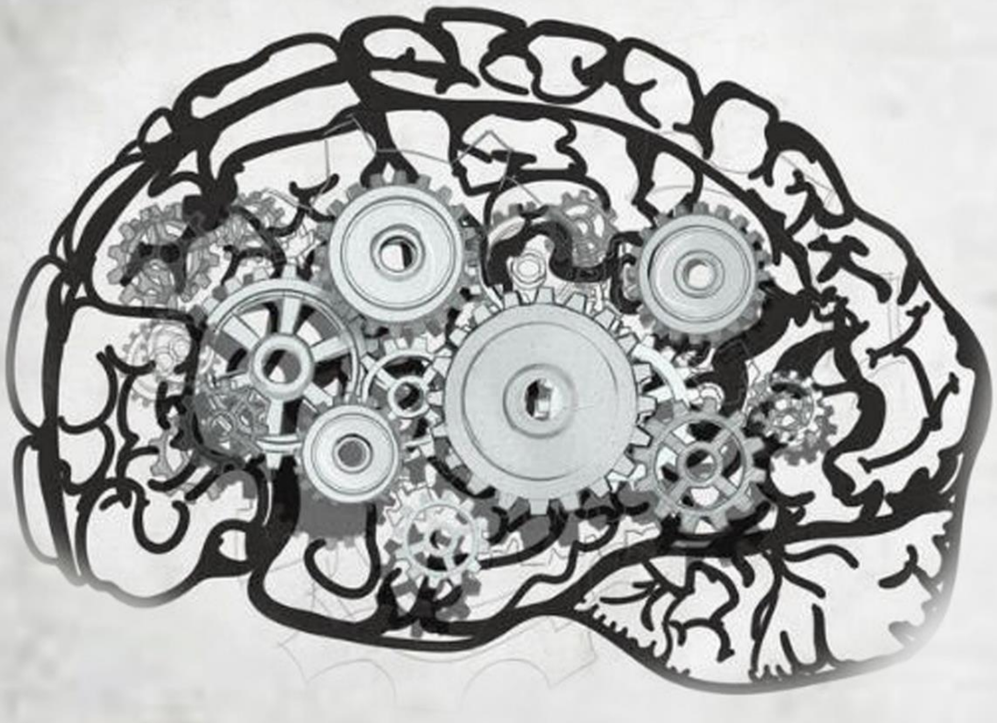
SET IT UP YOURSELF

DEEP LEARNING QUICK START GUIDE



THIMIRA AMARATUNGA

Deep Learning Quick Start Guide

How to setup all the tools you need for Deep Learning and Computer Vision

Thimira Amaratunga

www.codesofinterest.com

CODES OF INTEREST

Deep Learning - Quick Start Guide

Version 2.0, November 2017

Thank you for downloading the Quick Start Guide for Deep Learning and Machine Learning from **Codes of Interest**. This guide will help you to setup the right set of tools to get you jump started with Machine Learning and Computer Vision.

Head over to the Codes of Interest Blog (http://www.codesofinterest.com/) to get the latest tips and tricks, knowledge, and news on variety of topics on Artificial Intelligence, Machine Learning, Deep Learning, Computer Vision, Image Processing, and more.

Feel free to contact me - on any questions you have on any topics covered in this guide, or the blog - at thimira@codesofinterest.com.

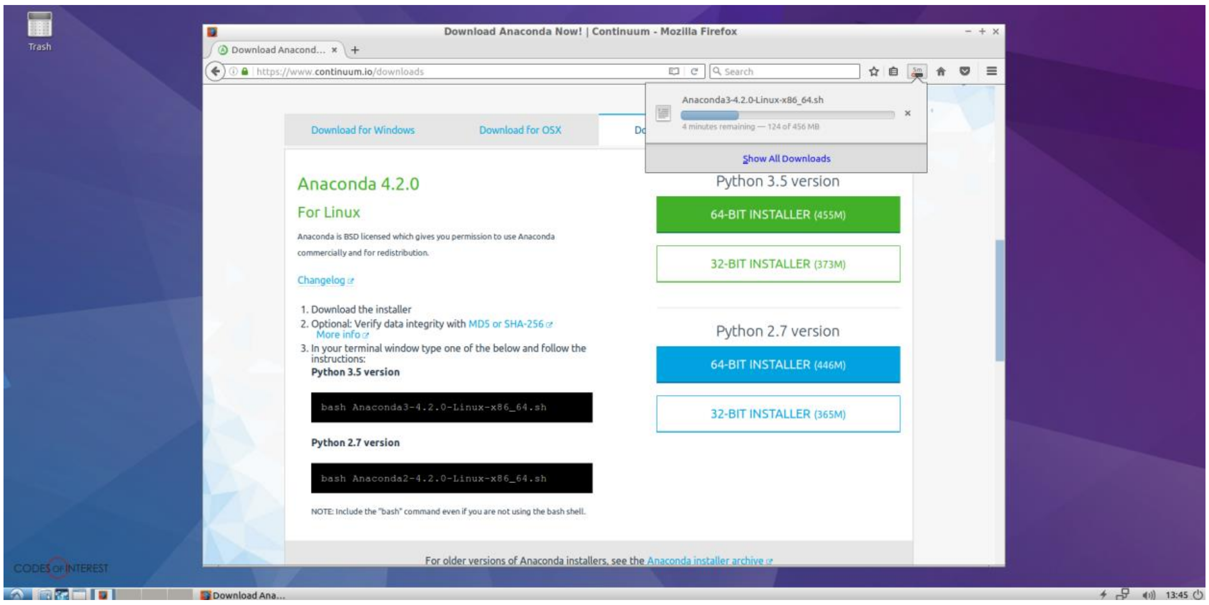
- Thimira Amaratunga

Table of Contents

Anaconda Python

Anaconda is an Open Source platform meant for data science and scientific computing. It contains optimized versions of Python for many platforms and architectures, and hundreds of pre-built and tested python packages for those platforms. Its conda package manager simplifies the task of creating isolated python environments and manage packages in them.

Installing Anaconda is as simple as downloading the installer package for the platform you want (Windows, Linux, and Mac OS) from the Anaconda Downloads Page, and following the instructions to install it.



Anaconda 4.2.0

For Linux

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

Changelog

1. Download the installer
2. Optional: Verify data integrity with MD5 or SHA-256
   More info
3. In your terminal window type one of the below and follow the instructions:

Python 3.5 version

```
bash Anaconda3-4.2.0-Linux-x86_64.sh
```

Python 2.7 version

```
bash Anaconda2-4.2.0-Linux-x86_64.sh
```

NOTE: Include the "bash" command even if you are not using the bash shell.

For older versions of Anaconda installers, see the Anaconda installer archive

I recommend using the **Python 3.5 64-Bit** version for your Machine Learning experiments.

Python Libraries

I recommend installing following Python packages as a base. These set will help you get started with most of the common tasks in Machine Learning.

- Numpy
- Scipy
- Scikit-learn
- Scikit-image
- Pillow
- H5py
- Matplotlib

Installing these would be quite simple with Anaconda Python - just create a new Anaconda environment with the packages installed using the following command,

```
conda create --name deep-learning python=3.5 numpy scipy
scikit-learn scikit-image pillow h5py matplotlib
```

Here, we are naming our environment '**deep-learning**'. But, you can name it anything you like. The latest Anaconda will try to install Python 3.6 by default. So make sure you tell it explicitly to install Python 3.5 by using the '**python=3.5**' flag.

Once you create the environment, you can activate it using,

```
activate deep-learning
```

On Linux or Mac OS, it should be,

```
source activate deep-learning
```

Note: On Windows, you may need to install 'mingw' and 'libpython' packages as well. i.e.

```
conda create --name deep-learning python=3.5 numpy scipy
scikit-learn scikit-image pillow h5py matplotlib mingw
libpython
```

OpenCV

OpenCV is the de-facto standard library when it comes to Computer Vision. It's primarily written for C++, but also contains a comprehensive Python interface. There's currently 2 major versions of OpenCV - v2.x and v3.x. I recommend using OpenCV v3 (v3.2 being latest at the time of this writing), as it is more optimized.

OpenCV has pre-built binaries for Windows. But you will have issues getting them to work with 64-Bit Python 3.5 (issues explained here).

The easiest way to get OpenCV 3.2 working on Windows with 64-Bit Python 3.5 is to use the Anaconda package.

```
conda install -c conda-forge opencv=3.2.0
```

For Linux, installing OpenCV is bit more complicated - you need to build OpenCV from source on your platform. I have made a comprehensive guide on how to compile OpenCV for Anaconda Python on Ubuntu, which you can find at Installing OpenCV from source on Anaconda Python on Ubuntu 16.10.

Keras

Keras is a Machine Learning Framework, written in Python and capable of running on top of either TensorFlow, CNTK, or Theano, and simplifying creating complicated neural network structures. It was developed with a focus on enabling fast experimentation.

Installing Keras is as simple as,

```
pip install keras
```

As mentioned, Keras doesn't operate alone. It needs a 'backend' – which is either TensorFlow, Theano, or CNTK. It defaults to TensorFlow. Which backend to use in Keras is defined in the **keras.json** file, which is located at ~/.keras/keras.json in Linux and Mac OS, and at %USERPROFILE%\.keras\keras.json on Windows.

You can use my guide "What is the image data format parameter in Keras, and why is it important" to understand the important parameters in the keras.json file and how to switch the backend properly.

Theano

Theano is a numerical computation and deep learning library, and one of the backends supported by Keras. Theano can utilize libraries like OpenBLAS, and Nvidia CUDA to run efficiently on CPU and GPU respectively.

Theano is available through pip (with pip install theano). But it's better to always install the latest development version by running,

```
pip install --upgrade --no-deps
git+git://github.com/Theano/Theano.git
```

Also, always install Theano before Keras to avoid any dependency issues.

TensorFlow

TensorFlow is the 2nd generation Machine Learning library by the Google Brain Team, and has gained a huge popularity due to its deep learning capabilities, and the new approach it took on tensor representation.

Installing TensorFlow used to be complicated (and only worked on Linux in the beginning). But since TensorFlow v0.12, it can be installed from pip, and works on Windows with both CPU and GPU support.

The latest version of TensorFlow, at the time of this writing is **TensorFlow v1.4.0**. To install it on Windows, run the following 2 commands.

```
pip install --upgrade --no-deps tensorflow
```

```
pip install tensorflow
```

Note: You need to run both these commands to properly install it.

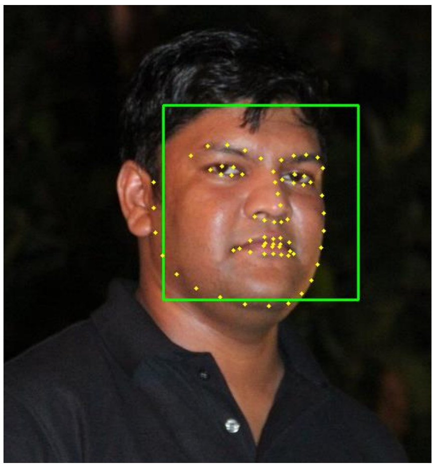
This installs the CPU version of TensorFlow. Check out my guide on [Setting up TensorFlow on Windows]() for more details.

To setup the TensorFlow GPU version on Windows, follow my guide on Setting up TensorFlow with CUDA on Windows. (You need to complete the CUDA installation as part of getting TensorFlow GPU version working)

For other platforms (Linux, Mac OS), check out the official Install Guide. Those are also almost similar to the Windows installation steps.

Dlib

Dlib is a library and a toolkit for machine learning, linear algebra, data structures, image processing, computer vision, data mining, XML and text parsing, numerical optimization, and many other tasks. It also has some handy functions for computer vision.


Dlib Face Landmark Detection in action

Dlib is available through pip. However, you need to have CMake installed with a C++ compiler, and Boost Python installed. The easiest way to get Dlib working is to use the Anaconda package.

```
conda install -c conda-forge dlib=19.4
```

Check out more details at [Installing Dlib on Anaconda Python on Windows](). You can use the same commands to install on Linux as well.

OpenBLAS

OpenBLAS is an open source implementation of the BLAS (Basic Linear Algebra Subprograms), containing optimizations for many specific processor types. Machine Learning libraries such as Theano is able to speed up certain routines by utilizing BLAS libraries.

For Windows, OpenBLAS has pre-built binaries. My guide Getting Theano working with OpenBLAS on Windows covers how to set it up on Windows. For Linux, the installation can be done with the following command,

```
sudo apt-get install libopenblas-dev
```

Installing OpenBLAS is only needed if you're running Theano on CPU. TensorFlow has its own internal CPU optimizers, and thus doesn't need (or use) OpenBLAS. But with Theano, it's recommended to have OpenBLAS setup, as it sometimes doubles the speed of which deep learning models train on it.

CUDA and cuDNN

CUDA is a parallel computing platform and programming model invented by Nvidia. It enables dramatic increases in computing performance by harnessing the power of the GPU. cuDNN - or, CUDA Deep Neural Network library - is a GPU-accelerated library of primitives for deep neural networks. cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers. Using CUDA and cuDNN along with either Theano or TensorFlow could extremely speed up your neural networks (networks which took hours to train might take just minutes). The only requirement is that you need to have a CUDA supported Nvidia GPU in your system. A list of supported GPUs can be found here.

In order to install CUDA, head over to the CUDA Downloads Page, and follow the instructions. However, before installing, you need to have a couple of dependencies installed. On Windows, you need to have Visual Studio 2015 installed. We'll be installing **CUDA Toolkit 8.0** with **cuDNN 5.1 for CUDA 8.0**. These only work with Visual Studio 2015, not with the latest 2017 version. You may be able to get Visual Studio 2015 Community Edition (Free) from this page.

Also, make sure you select to install the C++ compiler when you install. On Linux, make sure you have GCC installed (installing the build-essential package should be enough).

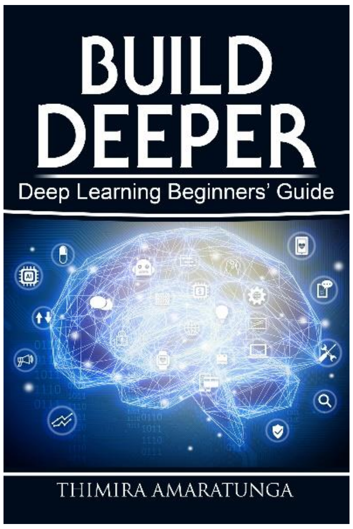
In order to get cuDNN, head over to Nvidia cuDNN Page and download the zip file. You will need to register for an Nvidia Developer Account (it's free) in order to download. When you download and unzip the package, you will get 3 directories - 'bin', 'include', and 'lib'. Copy those to your CUDA installation directory, which also has 3 directories named ‘bin’, ‘include’, and ‘lib’. i.e. the contents from cuDNN ‘bin’ should go in to CUDA ‘bin’ directory, ‘lib’ to ‘lib’, and ‘include’ to ‘include’.

Follow my guide on Working Theano configs in order to setup Theano with CUDA. For TensorFlow, you need to point the LD_LIBRARY_PATH and CUDA_HOME environment variables to the appropriate path in your CUDA installation path. Instructions for TensorFlow can be found here.

Go Further, Build Deeper

Ready to go further in Deep Learning?

My book, **Build Deeper: Deep Learning Beginners' Guide** will help you take the next step.



The book is meant to be a guide for the enthusiasts of Deep Learning. It takes a deeper look into the libraries we looked at in this Quick Start Guide, as well as concepts of Deep Learning.

It looks at the history of Deep Learning, and what makes it 'deep'.

It guides you through to building your first Deep Learning model, and sets your thoughts on more advanced topics.

Build Deeper: Deep Learning Beginners' Guide is now available through Amazon, in both Paperback and Kindle formats.

[Get your copy now!](https://www.amazon.com/dp/1549681060)

https://www.amazon.com/dp/1549681060