Face Recognition in TensorFlow Tutorial

Advanced Sensing Computation and Control (ASCC) Lab
Oklahoma State University
September, 2016

Contents

I. Install TensorFlow (Ubuntu 14.04)

TensorFlow are able to run in many environments. This setup guide will only focus on how to install in Ubuntu 14.04, but it will includes setup guide in raspberry pi, android in futures. It might will works similar on Ubuntu 15.04-16.04 but maybe few of the library are old.

Basically, the idea is to use the inception-v3 models [1] from Imagenet challenge winner for image recognition. Then, we retrained the final layer by using lab member facial images. We also trained the new face model from scratch, but because of the less data, the accuracy was decrease.

There are many ways to install TensorFlow Ubuntu such as Pip install, Virtualenv install, Anaconda install, Docker install and install from sources but install by Virtualenv is most convenience because it will create a virtual storage for TensorFlow and does not conflict the other dependencies. (Refer to TensorFlow website https://www.tensorflow.org/versions/r0.10/get_started/index.html)

1. Install pip and Virtualenv:

Virtualenv is a tool to keep the dependencies required by different Python projects in separate places. The Virtualenv installation of TensorFlow will not override pre-existing version of the Python packages needed by TensorFlow. (**Whenever you want to use code need to import tensorfow, you have to active this environment**)

```
# Ubuntu/Linux 64-bit
$ sudo apt-get install python-pip python-dev python-virtualenv zip
```

Create a Virtualenv environment in the directory ~/tensorflow:

```
$ virtualenv --system-site-packages ~/tensorflow
```

Activate the environment:

```
$ source ~/tensorflow/bin/activate  # If using bash
$ source ~/tensorflow/bin/activate.csh  # If using csh
```

When you want to exit the environment

```
$ deactivate
```

2. Finally install TensorFlow:

You also need to check with **the newest** version on TensorFlow website and with specific computer with CPU or GPU as well. We use CPU version.

```
(tensorflow)$ export
TF_BINARY_URL=https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-0.10.0-cp27-none-linux_x86_64.whl
```

Python 2

```
(tensorflow)$ pip install --upgrade $TF_BINARY_URL
```

Python 3

```
(tensorflow)$ pip3 install --upgrade $TF_BINARY_URL
```

From here you're able to use some TensorFlow examples.

```
 $ python
...
>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!')
>>> sess = tf.Session()
>>> print(sess.run(hello))
Hello, TensorFlow!
>>> a = tf.constant(10)
>>> b = tf.constant(32)
>>> print(sess.run(a + b))
42
>>>
```

3. **Clone the TensorFlow repository**

```
$ cd tensorflow
$ git clone --recurse-submodules https://github.com/tensorflow/tensorflow
```

4. **Configure the installation**

Run the configure script at the root of the tree of tensorflow (~/tensorflow/tensorflow). If you have a GPU card, you can enable to make the training time faster.

```
$ ./configure
Please specify the location of python. [Default is /usr/bin/python]:
Do you wish to build TensorFlow with Google Cloud Platform support? [y/N] N
No Google Cloud Platform support will be enabled for TensorFlow
Do you wish to build TensorFlow with GPU support? [y/N] N
GPU support will be unenabled for TensorFlow

Configuration finished
```

However, on purpose of retraining the image recognition model we have to install the bazel as well to compile C++.

5. **Install Bazel**

Check to install with **the newest version** and also the installation manual.
https://github.com/bazelbuild/bazel/releases
https://www.bazel.io/versions/master/docs/install.html

Install JDK 8
Ubuntu Trusty (14.04 LTS). OpenJDK 8 is not available on Trusty. To install Oracle JDK 8:

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
```

This is example of install bazel-0.1.4-installer-linux-x86_64.sh (0.1.4 version):

```
$ wget https://github.com/bazelbuild/bazel/releases/download/0.1.4/bazel-0.1.4-installer-linux-x86_64.sh
```

Run the installer

```
$ chmod +x bazel-0.1.4-installer-linux-x86_64.sh
$ ./ bazel-0.1.4-installer-linux-x86_64.sh --user
```

Set up your environment

```
$ export PATH="$PATH:$HOME/bin"
```

Install other dependencies

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

6. **Retrain the last layer of model by using bazel.**

So, Bazel is a build tool which help build the model.
Build the retrainer from root of TensorFlow. Refer to document:
https://www.tensorflow.org/versions/r0.10/how_tos/image_retraining/index.html

```
$ bazel build tensorflow/examples/image_retraining:retrain
```

Then use can build the final layer with your own categories

```
$ bazel-bin/tensorflow/examples/image_retraining/retrain --image_dir
/path/to/categories
```

After training section, you **will see a model, label file, bottleneck folder in /tmp**, copy them to **your directory** since everything in /tmp will be deleted after restart.

- You also can try to retrain a model by using python code which store in ./tensorflow/example/image_retraining/retrain.py –image_dir /path/to/categories (faster). However, the different branch of tensorflow may conflict with that python file. You may need to install some individually libraries. (Check TensorFlow github)

- The categories should be the face dataset people you want to recognize. Each people should collect at least 200 images of different angles and expression to make the best result. The size of image should be in square dimension for example 800x800 and .jpg (Check the section II to know how to aligned the image before using to train)

- It maybe will occur an error here with low efficient memory. Swap the storage with a memory follow that document:

  https://digitizor.com/create-swap-file-ubuntu-linux/

- Maybe it comes from the bazel version. Check with newest version.

- Always able to check the issues on TensorFlow github:

  https://github.com/tensorflow/tensorflow

7. **Using retrained model.**

There are two ways to use the retrained model.
First, we can use Bazel but it will take longer time.

```
bazel build tensorflow/examples/label_image:label_image && \
bazel-bin/tensorflow/examples/label_image/label_image \
--graph=/tmp/output_graph.pb --labels=/tmp/output_labels.txt \
--output_layer=final_result \
--image=/your/image/
```

The second way is to use the **classify_image.py** from image recognition example (stored in /tensorflow/models/image/imagenet/) but you need to modify the python file to match with the location of new model and label.

Check the file **classify.py** (zip file) with already modify with new trained model. Copy the **classify.py** (zip file) to the folder with your model, label was trained above. Try to test with few images (check inside python file to know how to change the name of the images)

II. Install Dlib and OpenCV

The idea is to use Dlib and OpenCV to landmark the face, align the face and detect the face. Before going in the training section, the image need to be aligned to pre-process the input data which make a better result, and also to detect and recognize the face. (However, you still can use the raw data to test first but it will maybe affect by the background of the images, so if you take pictures and test picture. Try to use the same background)

1. Install OpenCV

(Refer to the document: http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html)

Required Packages:

```
$ sudo apt-get install build-essential
$ sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev
$ sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
$ Cd ~/openface
$ Wget https://github.com/Itseez/opencv/archive/2.4.11.zip
$ Unzip 2.4.11.zip
$ Cd opencv-2.4.11
$ mkdir release
$ cd release
$ cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local ..
$ make
$ sudo make install
```

2. Install Dlib

(Refer to the document: https://pypi.python.org/pypi/dlib)

Require packages

```
$ apt-get install -y build-essential cmake curl gfortran git graphicsmagick libatlas-dev libavcodec-dev libavformat-dev libboost-all-dev libgtk2.0-dev libjpeg-dev liblapack-dev libswscale-dev pkg-config python-dev python-numpy python-protobuf zip
$ sudo apt-get install libpython-dev
```

```
$ sudo apt-get install libopenblas-dev liblapack-dev
$ sudo apt-get install cmake libblkid-dev e2fslibs-dev libboost-all-dev libaudit-dev
```

```
$ mkdir -p ~/src
$ cd ~/src
$ wget https://github.com/davisking/dlib/releases/download/v18.16/dlib-18.16.tar.bz2
$ tar xf dlib-18.16.tar.bz2
$ cd dlib-18.16/python_examples
$ mkdir build
$ cd build
$ cmake ../../tools/python
$ cmake --build . --config Release
$ sudo cp dlib.so /usr/local/lib/python2.7/dist-packages
```

Install dependencies: (zip file)

```
$ apt-get update
$ apt-get install -y curl git graphicsmagick python-dev python-pip python-numpy python-nose python-scipy python-pandas python-protobuf wget zip
$ apt-get clean
$ rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
$ pip2 install -r requirements.txt
$ pip2 install -r training/requirements.txt
```

3. Aligned the images: (zip file)

The best images size is 96x96 and because the different images may with different angle, align image will make the eyes in the same place and fixed the size to 96x96.

Prepare

./models.sh (zip file) to download the landmark of face dlib.

3.1 Create raw image directory.

Create a directory for your raw images so that images from different people are in different subdirectories. The names of the labels or images do not matter, and each person can have a different amount of images. The images should be formatted as jpg and have a lowercase extension.

```
$ tree /mydataset/
person-1
├── image-1.jpg
├── image-2.jpg
...
└── image-p.jpg

...

person-m
├── image-1.jpg
├── image-2.jpg
...
└── image-q.jpg
```

3.2 Preprocess the raw images

Change 8 to however many separate processes you want to run:for N in {1..8}; do

```
$ ./align-dlib.py <path-to-raw-data> align outerEyesAndNose <path-to-aligned-data> --size 96 & done
```

- I created the **classify_align.py** file which using the aligned dataset with better result.
- However, that file only using for testing with define for the image. To get the real-time result on Ubuntu. I refer two file one I am using on Raspberry Pi which take the image from Pi camera and do the recognition called **classify_rasp.py**. The other file called **classify_camera.py** which you will found the function to enable the webcam and take, recognize the face.
- You can check the folder 'Test'. It is my trained model and label with lab members. You can make a same folder to test it.