How do THEY do it?

(Tensorflow seminar)

Speakers: Mr. Akshay Sharma akshay.sharma2402@gmail.com
Mr. Mrunal Sawant mrunalsawant7@gmail.com
Mr. Vipul Modak vipzshady@gmail.com

Affiliation: Speakers are ***third year*** students of ***M.Sc.*** in Industrial Mathematics with Computer Applications at ***Department of Mathematics***, Savitribai Phule ***Pune University***.
They have completed in ***B.Sc.*** in ***Mathematics*** from the Savitribai Phule ***Pune University*** and now share great interest in computer programming.

Project Coordinator: Mr. Shashank Date stdate@gmail.com

Affiliation: Shashank Date got introduced to Vipul, Mrunal and Akshay when he was invited to present on "The Age of IoT - programming for small devices", at Dept. of Mathematics, Pune University in ***Feb 2017***. He has been guiding these three students since then and completed two small projects with them. One of the these projects ***introduced the students to the TensorFlow*** software library for Machine Intelligence developed by Google. The students completed a project to do ***image recognition*** using TensorFlow.

Shashank Date has over thirty years of experience in software development and project management. He is an ***alumni of Pune University*** and began his career at the Mathematics Department working as a ***research assistant on a DRDO project.*** He immigrated to the United States many years ago and has worked at various international companies like ***GE, Sprint, Google***. He visits his alma mater often and is very interested in guiding students to improve their chance of success.

About seminar

- In this seminar speakers will share their experiences in learning how to use Tensorflow (TF) and then applying it to solve the image recognition problem.
- Participants will learn how to install TF, how to develop using TF API, how to achieve image recognition and lastly, an idea to make an android application using Tensorflow.

About Tensorflow

- TensorFlowTM is an open source software library for numerical computation using data flow graphs. It is a framework for Artificial Intelligence.
- Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them.
- The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API.

- TensorFlow was originally developed by researchers and engineers working on the Google
- Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

Please visit

Please visit the below mentioned URL(GitHub repo) for the source code, and helpful links to understand in detail.
Site contains the information about our project. Also useful to understand our below write up.

https://github.com/Akshay2402/How-do-THEY-do-it-

Image Net

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently it have an average of over five hundred images per node. ImageNet is a useful resource for researchers, educators, students and all of you who share our passion for pictures. visit here **http://www.image-net.org/**

Inception v3 Model

Inception was developed at Google to provide state of the art performance on the ImageNet Large-Scale Visual Recognition Challenge and to be more computationally efficient than its competitor architectures. However, what makes Inception exciting is that its architecture can be applied to a whole host of other learning problems in computer vision.

Training your own Image Classifier

1. Gather your training dataset:

- Firstly you have to gather your own dataset, How to do it?
  if you dont have a data set and want one you are at the right place. Here is the trick.
  Can you not make a dataset from search result of google images?
  Exactly, you can. Here's how you will do it.
  Visit Here https://32hertz.blogspot.in/2015/03/download-all-images-from-google-search.html
- Now cleaning your dataset seems to be a easy job.
- Place the Dataset in a folder with different classes as different folders in the dataset.
  (if you have flowers as a dataset then the classes will contain
  /Daisy
  /roses
  /sunflowers
  different flowers with desired images)

2. Run the Training:

As noted in the introduction, Imagenet models are networks with millions of parameters that can differentiate a large number of classes. We're only training the final layer of that network, so training will end in a reasonable amount of time.

Start your retraining with one big command (note the --summaries_dir option, sending training progress reports to the directory that tensorboard is monitoring) :

```
python -m scripts.retrain \
  --bottleneck_dir=tf_files/bottlenecks \
  --how_many_training_steps=500 \
  --model_dir=tf_files/models/ \
  --summaries_dir=tf_files/training_summaries/"${ARCHITECTURE}" \
  --output_graph=tf_files/retrained_graph.pb \
  --output_labels=tf_files/retrained_labels.txt \
  --architecture="${ARCHITECTURE}" \
  --image_dir=tf_files/YOUR_DATASET_NAME
```

This script downloads the pre-trained model, adds a new final layer, and trains that layer on the dataset photos you've downloaded.

The above retraining command iterates only 500 times. You can very likely get improved results (i.e. higher accuracy) by training for longer. To get this improvement, remove the parameter --how_many_training_steps to use the default 4,000 iterations.

3. Classifying an Image:

The repo also contains a copy of tensorflow's label_image.py example, which you can use to test your network. Take a minute to read the help for this script:

```
python -m  scripts.label_image -h
```

Now, let's run the script on a test image :

```
python -m scripts.label_image \
    --graph=tf_files/retrained_graph.pb  \
    --image=tf_files/YOUR_DATASET/FOLDER1/IMAGE_NAME
```

Each execution will print a list of labels, in most cases with the correct on top (though each retrained model may be slightly different).

You might get results like this:

```
CLASS_OF_IMAGE (score = 0.99071)
OTHER_CLASS (score = 0.00595)
OTHER_CLASS (score = 0.00252)
OTHER_CLASS (score = 0.00049)
OTHER_CLASS (score = 0.00032)
```

This indicates a high confidence (~99%) that the image is a correctly classified image, and low confidence for any other label.

You can use label_image.py to classify any image file you choose, either from your downloaded collection, or new ones. You just have to change the --image file name argument to the script.