Topics to cover

Introduction: about what we are going to do this. What prior knowledge required for this.

Revising what we have learned and on.

About the assignment.

Why chosen this assignment.

About tensorflow,

Which model we are going to create.

How to write the model in tensorflow.

Which algorithms we are using. And what about that.

And how we are combining them.

Hw obtimization is gogin to happen.

How to conversion is happening to .pb and retrained graph.txtx labels.

About tfcomreml

About inspect.

Hpow to install them.

How yo use them to convert a tensorflow model to coreml

How to create app in xcode.

Explanation of code.

Introduction

[Creating Tensor flow Image recognition model: 2](#_Toc517258341)

Objective:

To create an image recognition IOS app using a model created using Tensorflow created model.

To accomplish this, these are the series of tasks we have to perform.

* Create Tensor flow Image recognition model
* Converting it into a .ml model file
* And create an IOS / SWIFT app to use that model.

# Creating Tensor flow Image recognition model:

Usually to create a tensorflow model it takes a long series of steps and coding as this chapter is not focusing on this. Here we are using a pre build **inceptionv3 model** which you can download from.

<https://github.com/azamsharp/TensorFlowToCoreML/raw/master/inception_v1_2016_08_28_frozen.pb>

And also it’s labels from <https://raw.githubusercontent.com/azamsharp/TensorFlowToCoreML/master/imagenet_slim_labels.txt>

Once we get these two file. We can say that we are having tensor flow model for image recognition.

If you are interested to create you custom image recognition model through tenso flow you can go through this tutorial.

<https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/#0>

Converting it into an .mlmodel file:

In order to use this tensorlow model in to an IOS app we have to convert it in to an IOS understandable format which is coreml format.

Installing tfcoreml

There are multiple ways of installing tfcoreml tool. The quickest way is to use the pip tool to install it.

pip install -U tfcoreml

At the time of this writing I would recommend against the above method. The reason is that there were some fixes to the source of the tfcoreml tool in master branch which are only available if you build the tool from source

Converting TensorFlow Model

Before performing the actual conversion let’s get a hold of the TensorFlow model. You can find several TensorFlow compatible models included at the end of the documentation. We are going to the “Inception v1 (Slim) model for our demo. Download the model and you will notice that it contains two files.

inception\_v1\_2016\_08\_28\_frozen.pb

imagenet\_slim\_labels.txt

The file “inception\_v1\_2016\_08\_28\_frozen.pb” is the actual model and the “imagenet\_slim\_labels.txt” is the class labels. You can think of class labels as the label/title that will be attached to each prediction.

In the documentation you will see the following conversion code:

import tfcoreml as tf\_converter

tf\_converter.convert(tf\_model\_path = 'my\_model.pb',

mlmodel\_path = 'my\_model.mlmodel',

output\_feature\_names = ['softmax:0'])

I created a file called “convertor.py” and placed all the above code in that file. As you can see I have substituted the variables with the correct file names

WE have missed some points. Read the <https://hackernoon.com/integrating-tensorflow-model-in-an-ios-app-cecf30b9068d> and update the project code with custom model.