Project Report

Course Name: Artificial Intelligence (101)

Project: Brand Classifier

Group Members:

Student ID	Name
20198020	Nowroz Islam
20188433	WanXuqi
20152785	Mohammad

Course Tutor: Prof. Ho Jiacang

Spring 2019

Table of Contents

Section 1 Introduction

1.1 Purpose

Brand Classifier is an application based on Convolutional Neural Network designed to classify the brand of a product from an image. It provides a simple graphical user interface enabling the user to browse and select an image file and run the application to identify the brand of the product. The application can classify only three brands: Apple, Lenovo, and Samsung.

Section 2 Technical Architecture

2.1 System Architecture

Software, tools and language requirements:

- Language: Python 3
- Library: TensorFlow
- GUI: tkinter Package
- OS: Linux/ Windows/ macOS

System Configuration during build:

- CPU: Intel Core i3
- OS: Ubuntu 18.04.2 LTS
- RAM: 4GB
- Python 3.6.7
- TensorFlow 1.13.1 (Built from source CPU-only)
- tkinter GUI Package

Section 3 Detailed Design

3.1 User Interface Design Overview

The graphical user interface is kept as simple as possible. The graphical user interface consists of two buttons (‘Browse Image’ and ‘Run’) and two labels (Path’ and ‘Output’). The user must press the button, ‘Browse Image’ to select an image file before pressing the ‘Run’ button. Selecting an image will update the label, ‘Path:’ and display the path of the selected image. After that, user may press the ‘Run’ button to start the application. The application will display

the classified brand by updating the ‘Output:’ label and below that it will display the accuracy. Pressing the ‘Run’ button without selecting an image will display “MUST SELECT AN IMAGE FIRST” on the ‘Output’ label.

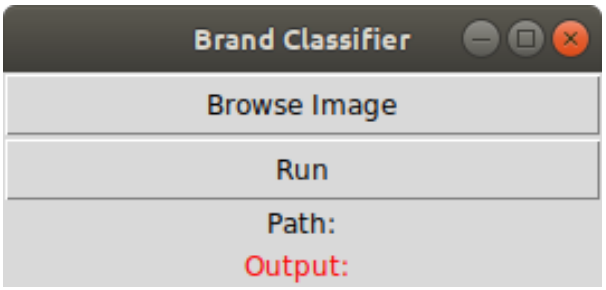

Illustration 1: Brand Classifier

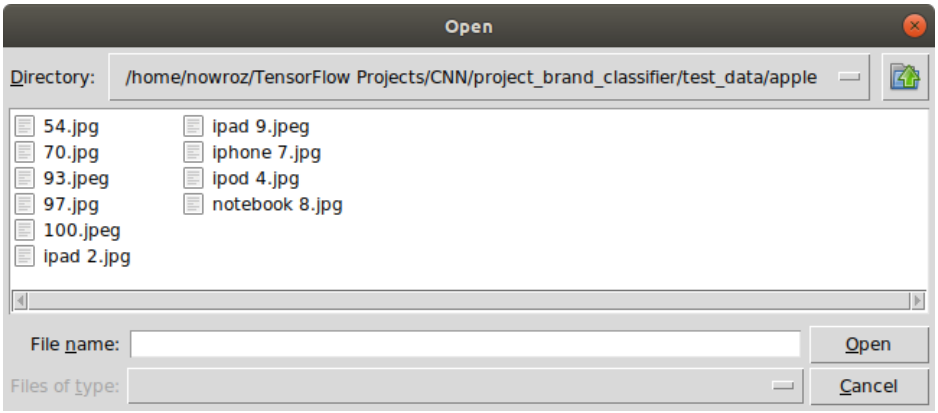

Illustration 2: Browse and select image

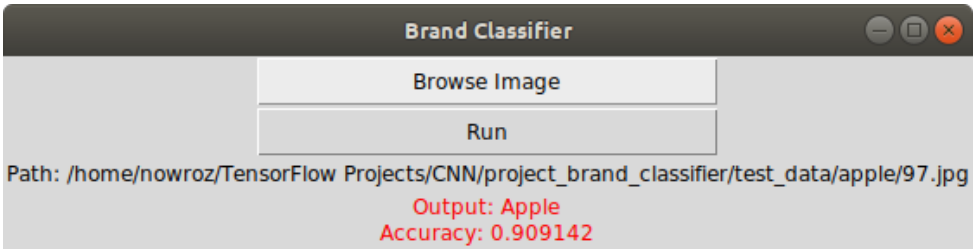

Illustration 3: Displaying output with accuracy

Illustration 4: Asking the user to select an image first

3.2 Convolutional Neural Network Design

The training dataset contains a set of 270 RGB images. Each brand (Apple, Lenovo, or Samsung) has a set of 90 images. The convolutional neural network is built with two convolutional modules followed by two dense layers. Each convolutional module consists of a convolutional layer followed by a pooling layer (Max Pool). The first convolutional layer has 32, 5x5 filters followed by a pooling layer that performs a max pooling with a 2x2 filter and

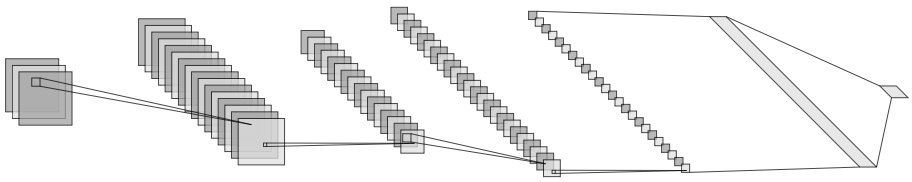

Illustration 5: CNN

stride of 2. The second convolutional layer has 64, 3x3 filters followed by a max pooling with a 2x2 filter and stride of 2. The output from the second convolutional module is flattened and connected to the first dense layer containing 1024 nodes with the dropout regularization rate of 0.25. Finally, the final dense layer is added with 3 nodes each representing one of the brands.

Section 4 Project Folder Contents

4.1 Contents

Folder: train_data

Contains the training data i.e., the 270 training images of Apple, Lenovo, and Samsung Devices.

Folder: test_data

Contains 30 test images of Apple, Lenovo, and Samsung Devices.

Folder: Report

Contains the project report both in odt and pdf format

Compressed File: train_data.tar.xz

Required to create the training dataset.

Python Script: brand_classifier_cnn.py

The script file to train and save the model.

Python Script: Application.py

The script file to run the Brand Classifier Application.

N.B. User must run the 'brand_classifier_cnn.py' file first if there is no file in the directory named 'trained_model.data-00000-of-00001' before running the 'Application.py' file.

Section 5 Training, Testing and Evaluation

5.1 Training

The application was trained with a batch size of 10. The training session was conducted for 14,999 steps. During training, the loss was shown at each step for the purpose of observation. At step number 14,999 the loss was approximately 0.000000. After that, the training session was ended and the trained model was saved in the project directory.

5.2 Testing

The testing data was retrieved from the 'test_data' directory. The directory contains three sets of images (10 images per set) each belonging to the brands Apple, Lenovo and Samsung respectively. During the testing, the application was able to classify brands as follows:

Apple: 6 out of 10 images were classified correctly.

Lenovo: 5 out of 10 images were classified correctly.

Samsung: 5 out of 10 images were classified correctly.

5.3 Evaluation

It is apparent that the model is not so accurate in classifying the images. It is because the data was insufficient and, the images were not pre-processed i.e, cropped so, it was expected that the model would not be so accurate. It was done intentionally because the purpose of this project was not to make a highly accurate application rather, it was to learn how to collect, prepare and feed the data to a convolutional neural network.

6 References

- https://www.tensorflow.org/tutorials/estimators/cnn
- https://www.tensorflow.org/api_docs/python/tf/data/Dataset
- https://medium.com/ymedialabs-innovation/how-to-use-dataset-and-iterators-in-tensorflow-with-code-samples-3bb98b6b74ab
- https://cv-tricks.com/tensorflow-tutorial/save-restore-tensorflow-models-quick-complete-tutorial/