Image Scraping and Classification

Submitted by:

Praveen Pandey

ACKNOWLEDGMENT

I would like to express my gratitude to my guide Shwetank Mishra (SME, Flip Robo) for his constant guidance, continuous encouragement and unconditional help towards the development of the project. It was he who helped me whenever I got stuck somewhere in between. The project would have not been completed without his support and confidence he showed towards me.

Lastly, I would l like to thank all those who helped me directly or indirectly toward the successful completion of the project.

INTRODUCTION

- Business Problem Framing

Images are one of the major sources of data in the field of data science and AI. This field is making appropriate use of information that can be gathered through images by examining its features and details. We are trying to give you an exposure of how an end-to-end project is developed in this field.

The idea behind this project is to build a deep learning-based Image Classification model on images that will be scraped from e-commerce portal. This is done to make the model more and more robust.

This task is divided into two phases: Data Collection and Mode Building.

Data Collection Phase: In this section, you need to scrape images from e-commerce portal, Amazon.com. The clothing categories used for scraping will be:

- Sarees (women)
- Trousers (men)
- Jeans (men)

You need to scrape images of these 3 categories and build your data from it. That data will be provided as an input to your deep learning problem. You need to scrape minimum 200 images of each category. There is no maximum limit to the data collection. You are free to apply image augmentation techniques to increase the size of your data but make sure the quality of data is not compromised.

Remember, in case of deep learning models, the data needs to be big for building a good performing model. More the data, better the results.

- ## Review of Literature

  From the dataset I get to know that it is a classification problem. And there are many features which help to find it.

- ## Motivation for the Problem Undertaken

  The project was the first provided to me by Flip Robo Technologies as a part of the internship programme. The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary motivation.

  Data needed for this project is require to scrap from E-commerce platform and data cleaning operation over it. Features derived from textual reviews are used to predict its corresponding star ratings. To accomplish it, the prediction problem is transformed into a multi-class classification task to classify reviews to one of the five classes corresponding to its star rating. Getting an overall sense of a textual review could in turn improve consumer experience. However, the motivation for taking this project was that it is relatively a new field of research.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

In order to apply image classification, the unstructured format of image has to be converted into a structured format for the simple reason that it is much easier for computer to deal with numbers of image. This is mainly achieved by projecting the image contents into Vector Space Model, where text data is converted into vectors of numbers.

In the field of image classification are commonly treated like image, meaning that each word is image from the others that are same in the other images. In such a model, the term frequency (occurrence of each image) is used as a feature in order to train the classifier. However, using the term frequency implies that all terms are considered equally important. As its name suggests, the term frequency simply weights each term based on their occurrence frequency and does not take the discriminatory power of terms into account. To address this problem and penalize images that are too frequent.

- Data Sources and their formats

Data is collected from Amazon.in and flipkart using selenium. Around 2000 images are collected for this project.

Loading data

```
# re-size all the images to this
IMAGE_SIZE = [224, 224]

Train = 'Images/Train'
Test = 'Images/Test'
```

This is multi-classification problem and Images is our target feature class to be predicated in this project. There are Three different images in feature target i.e., The image are trousers, sarees and Jean.

- ## Data Inputs- Logic- Output Relationships

The dataset consists of 3 features with a label. The features are independent and label is dependent as our label varies the values (image) of our independent variable's changes. Using images, we can see most occurring image for different categories.

- ## Hardware and Software Requirements and Tools Used

Hardware Used -

1. Processor — Intel i5 processor with 2.4GHZ

2. RAM — 4 GB

3. GPU — 2GB AMD Radeon Graphics card

Software utilised -

1.Anaconda – Jupyter Notebook

2. Selenium – Web scraping

Libraries used for web scraping data from e-commerce website are:

```
import selenium
import requests
import bs4
import time
import pandas as pd
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.keys import Keys
import warnings
warnings.filterwarnings('ignore')
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException
from matplotlib import pyplot as plt
import matplotlib.image as mpimg
import numpy as np
import urllib
import os
```

Libraries: -

Loading libraries

```
import os
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import cv2
import pandas as pd


from keras.layers import Activation, Dropout, Flatten, Dense
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Convolution2D, MaxPooling2D, ZeroPadding2D
from keras import optimizers
from keras.callbacks import TensorBoard, ModelCheckpoint,EarlyStopping


from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.vgg19 import VGG19
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras.applications.inception_resnet_v2 import InceptionResNetV2
from tensorflow.keras.applications.mobilenet import MobileNet

from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
from glob import glob
#import matplotlib.pyplot

from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession

config = ConfigProto()
config.gpu_options.per_process_gpu_memory_fraction = 0.5
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  First part of problem solving is to scrap data from amazon.in and flipkart.in website which we already done. Second is performing model building through transfer learning. Third part of problem building machine learning model to predict image. This problem can be solve using classification-based machine learning algorithm. Further performed to build more accurate model out of best model.

- Run and evaluate selected models

Building Model

1.loading the models with no last layer

```
 #Import the  library and add preprocessing layer to the front of transfer models
# Here we will be using imagenet weights

vgg16 = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
vgg19 = VGG19(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
resnet = ResNet50(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

```
# don't train existing layers- weights will change otherwise
for layer in vgg16.layers:
    layer.trainable = False

for layer in vgg19.layers:
    layer.trainable = False

for layer in resnet.layers:
    layer.trainable = False
```

```
x2 = Flatten()(vgg19.output)

x3 = Flatten()(resnet.output)
```

3.Checking our folder

```
# useful for getting number of output classes
folders = glob('Images/Train/*')
folders
```

```
['Images/Train\\Jeans', 'Images/Train\\sarees', 'Images/Train\\Trousers']
```

4.Appending folders as Dense Layer with activation function softmax

The folders will get appended as last layer with the flattened x

```
prediction1 = Dense(len(folders), activation='softmax')(x1)

prediction2 = Dense(len(folders), activation='softmax')(x2)

prediction3 = Dense(len(folders), activation='softmax')(x3)
```

```
print(prediction1)
print(prediction2)
print(prediction3)
```

```
KerasTensor(type_spec=TensorSpec(shape=(None, 3), dtype=tf.float32, name=None), name='dense/Softmax:0', description="created by
layer 'dense'")
KerasTensor(type_spec=TensorSpec(shape=(None, 3), dtype=tf.float32, name=None), name='dense_1/Softmax:0', description="created
by layer 'dense_1'")
KerasTensor(type_spec=TensorSpec(shape=(None, 3), dtype=tf.float32, name=None), name='dense_2/Softmax:0', description="created
by layer 'dense_2'")
```

5.Creating final model

```
# create a model object
model1 = Model(inputs=vgg16.input, outputs=prediction1)

model2 = Model(inputs=vgg19.input, outputs=prediction2)

model3 = Model(inputs=resnet.input, outputs=prediction3)
```

```
# view the structure of the model
print("--------model vgg16----------------------------------------------------------------")
print(model1.summary())

print("--------model vgg19----------------------------------------------------------------")
print(model2.summary())

print("--------model resnet---------------------------------------------------------------")
print(model3.summary())
```

Data Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

#data augmentation
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

# Applying data augmentation
#making sure inputs are of same size

training_set = train_datagen.flow_from_directory('Images/Train',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 1064 images belonging to 3 classes.
```

```
#data augmentation
test_datagen = ImageDataGenerator(rescale = 1./255)

test_set = test_datagen.flow_from_directory('Images/Test',
                                            target_size = (224, 224),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 160 images belonging to 3 classes.
```

```
# view the structure of the model
print("--------model vgg16-------------------------------------------------------------")
print(model1.summary())

print("--------model vgg19-------------------------------------------------------------")
print(model2.summary())

print("--------model resnet------------------------------------------------------------")
print(model3.summary())
```

```
# tell the model what cost and optimization method to use
model1.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)

model2.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)

model3.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)
```

Applying model on data

```
# fit the model
# Run the cell. It will take some time to execute
vgg16_model_fit = model1.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=25,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

```
vgg19_model_fit = model2.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=25,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

```
resnet_model_fit = model3.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=25,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

Choosing the best model

```
losses_vgg16=pd.DataFrame(model1.history.history)
print("VGG16----------------------------------------\n",losses_vgg16)
```

```
VGG16----------------------------------------
         loss  accuracy   val_loss  val_accuracy
0   1.035097  0.683913  10.311376       0.05000
1   0.368187  0.833490  10.437211       0.20000
2   0.229957  0.896519   9.211484       0.22500
3   0.170792  0.933208   8.458446       0.15625
4   0.148221  0.940734   8.448769       0.13125
5   0.167862  0.927563   8.763288       0.19375
6   0.132227  0.945437   8.603072       0.21875
7   0.114929  0.953904   9.431997       0.11250
8   0.123988  0.949200   9.648898       0.13125
9   0.108337  0.953904   9.399262       0.18125
10  0.097158  0.959548   9.927741       0.18125
11  0.080905  0.968956  10.017992       0.20000
12  0.074023  0.975541  10.210706       0.20000
13  0.068157  0.981185  10.283397       0.21250
14  0.061565  0.978363  10.261515       0.19375
15  0.069095  0.980245  10.268614       0.20625
16  0.048950  0.991533  10.698068       0.20625
17  0.059384  0.985889  10.321543       0.23125
18  0.088687  0.967074  10.215014       0.20000
19  0.050628  0.985889  10.416662       0.21875
20  0.087196  0.970837  10.132393       0.20625
21  0.045952  0.986830  10.860739       0.20000
22  0.052118  0.984948  10.773313       0.20000
23  0.050420  0.984008  10.664034       0.17500
24  0.048556  0.984008  11.245936       0.21250
```

- Visualizations

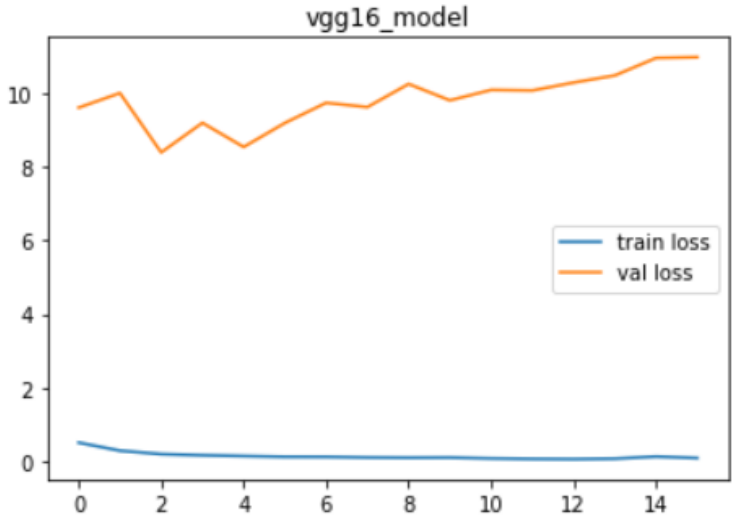
Accuracy of the model

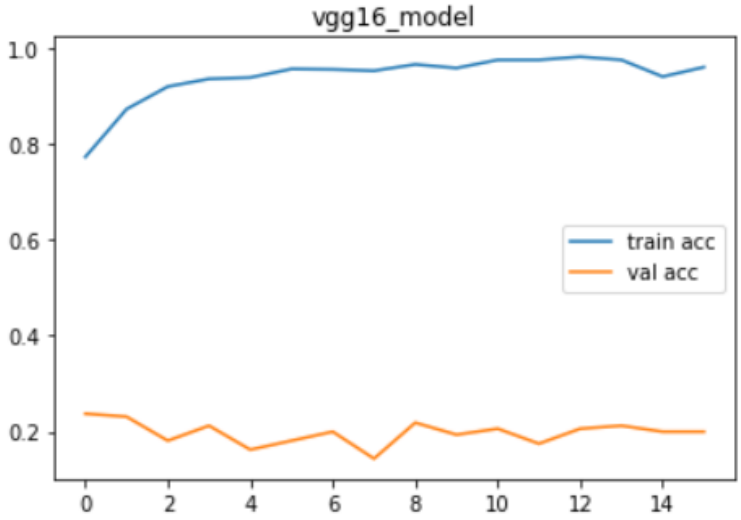
```
import matplotlib.pyplot as plt
```

```
# plot the loss

plt.plot(final_vgg16_model_fit.history['loss'], label='train loss')
plt.plot(final_vgg16_model_fit.history['val_loss'], label='val loss')
plt.legend()
plt.title("vgg16_model")
plt.show()
plt.savefig('LossVal_loss')

# plot the accuracy
plt.plot(final_vgg16_model_fit.history['accuracy'], label='train acc')
plt.plot(final_vgg16_model_fit.history['val_accuracy'], label='val acc')
plt.legend()
plt.title("vgg16_model")
plt.show()
plt.savefig('AccVal_acc')
```





CONCLUSION

In conclusion, this research is about image classification by using deep learning via framework TensorFlow and Keras. It has three objectives that have achieved throughout this research. The objectives are linked directly with conclusions because it can determine whether all objectives are successfully achieved or not. It can be concluded that all results that have been obtained, showed quite impressive outcomes. The deep neural network (DNN) becomes the main agenda for this research, especially in image classification technology. DNN technique was studied in more details

starting from assembling, training model and to classify images into categories. The roles of epochs in DNN were able to control accuracy and also prevent any problems such as overfitting. Implementation of deep learning by using framework TensorFlow also gave good results as it is able to simulate, train and classified with up to 90% percent of accuracy towards five different types of flowers that have become a trained model. Lastly, Python have been used as the programming language throughout this research since it comes together with framework TensorFlow which leads to designing of the system involved Python from start until ends