IMAGE CLASSIFICATION PROJECT

Submitted by-:
NITISH KUMAR SHARMA

ACKNOWLEDGEMENT

It is great pleasure for me to undertake this project. I feel overwhelmed doing this project entitled – “Image Classification Project“.

Some of the resources that helped me to complete this project are as follows:

- Youtube videos
- Blogs
- Stack overflow
- Analytics Vidhya
- Articles published in Medium.com

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.

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

Image classification is growing and becoming a trend among technology developers especially with the growth of data in different parts of industry such as e-commerce, automotive, healthcare, and gaming. The most obvious example of this technology is applied to Facebook. Facebook now can detect up to 98% accuracy in order to identify your face with only a few tagged images and classified it into your Facebook's album. The technology itself almost beats the ability of human in image classification or recognition.

One of the dominant approaches for this technology is deep learning. Deep learning falls under the category of Artificial Intelligence where it can act or think like a human. Normally, the system itself will be set with hundreds or maybe thousands of input data in order to make the 'training' session to be more efficient and fast. It starts by giving some sort of 'training' with all the input data. Machine learning is also the frequent systems that has been applied towards image classification. However, there are still parts that can be improved within machine learning. Therefore, image classification is going to be occupied with deep learning system.

REVIEW OF LITERATURE

Image classification has become a major challenge in machine vision and has a long history with it. The challenge includes a broad intra-class range of images caused by color, size, environmental conditions and shape. It is required big data of labelled training images and to prepare this big data, it consumes a lot of time and cost as for the training purpose only

In this paper, deep neural network, based on TensorFlow is used with Python as the programming language for image classification. Thousands of images are used as the input data in this project. The accuracy of each percentage of 'train' session will be studied and compared.

MOTIVATION FOR THE PROBLEM UNDERTAKEN

As directed that we have to scrape images from e-commerce website on our own and then build a deep learning model for Image Classification, which would be a whole new experience for me.

This type of project helps you to get hands on deep learning type of projects. It's a very interesting and at the same time also problematic to work on this kind of project. Mostly most of the information is conveyed through images, videos etc. Therefore I would be able to explore technologies like computer vision which is a part of AI (Artificial intelligence) which helps in classifying digital images.

ANALYTICAL PROBLEM FRAMING

MATHEMATICAL / ANALYTICAL MODELING OF THE PROBLEM

The directory created in the jupyter notebook contains two folders : One is Garments which contain subfolders of webscrapped images of Sarees, Trousers and Jeans for training data and other folder of Garment which contains same sub folders for test data.

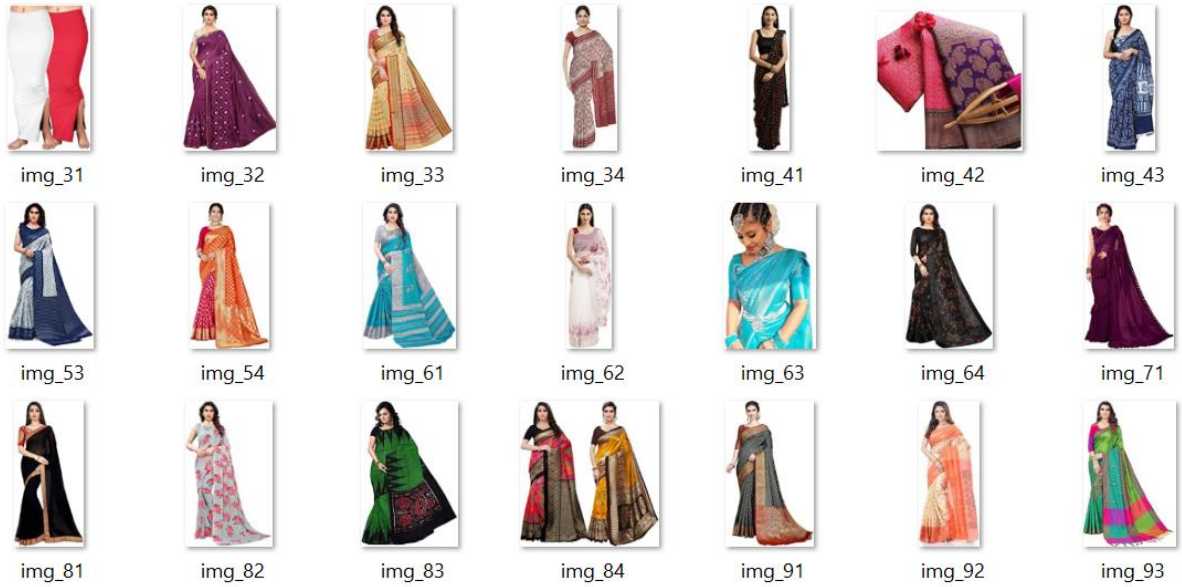
For training data there are total of 718 images and for each category of garments there is minimum 200 images and for test data there are 157 images.

DATA SOURCES AND THEIR FORMATS

The data (images) scrapped was saved in the directories created in the jupyter notebook.

All the images are in JPEG format.

- The above are the Images of Jeans which I have scraped and stored in a folder.



- The above are the Images of Sarees which I have scraped and stored in a folder.

- The above are the Images of Trousers which I have scraped and stored in a folder.

Displaying scrapped sample images of Jeans, Trousers and Trousers from ecommerce website (Amazon):

DATA PREPROCESSING DONE

In the pre-processing stage, we'll prepare the data to be fed to the Keras model. The first step is clearing the dataset of null values. Neural Network with numerical data, not categorical.

```
# Defining Dimensions for the image to be input and then loading the images
input_shape=(576,576,3)
img_width=576
img_height=576
nb_train_samples=192
nb_validation_samples=48
batch_size=8
epoch=150
```

- Setting the parameters for the images and feeding the values for model building

```
# Training Data Generator (Data Augmentation on Training Images)

Train_generator_augmented=ImageDataGenerator(rescale=1./255,
                                             zoom_range=0.2,
                                             rotation_range=30,
                                             horizontal_flip=True)
Train_generator=Train_generator_augmented.flow_from_directory(Train_data_dir,
                                                              target_size=(img_width,img_height),
                                                              batch_size=batch_size,
                                                              class_mode='categorical')

# Validation Data Generator
Data_gen=ImageDataGenerator(rescale=1./255)
validation_generator=Data_gen.flow_from_directory(validation_data_dir,
                                                  target_size=(img_width,img_height),
                                                  batch_size=batch_size,
                                                  class_mode='categorical')
```

- Image Data Generator used to generate data according to some pattern for images

```
# Defining Early stopping and Model check point
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint

ES = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=30)
MC = ModelCheckpoint('Image_Classification.h5', monitor='val_accuracy',
                     mode='max', verbose=1, save_best_only=True)
```

- Defined Early stopping by setting mode to min to stop training when the quantity monitored has stopped decreasing and Model check point mode to max to achieve maximum validation accuracy

```
# Fitting the Training Data
history = model.fit_generator(
    Train_generator,
    epochs=epoch,
    validation_data=validation_generator,
    validation_steps=nb_validation_samples//batch_size,
    steps_per_epoch=nb_train_samples//batch_size,
    callbacks=[ES,MC])
```

- Fitting data into memory

```
Epoch 75/150
24/24 [==============================] - ETA: 0s - loss: 0.3743 - accuracy: 0.8438
Epoch 00075: val_accuracy did not improve from 0.87500
24/24 [==============================] - 44s 2s/step - loss: 0.3743 - accuracy: 0.8438 - val_loss: 0.6383 - val_accuracy: 0.812
5
Epoch 00075: early stopping
```

- On 75 Epoch itself the val_accuracy is: 87.50% and val_loss:37.43%, thus saving the best model

HARDWARE / SOFTWARE REQUIREMENTS AND TOOLS USED

- Anaconda Navigator 1.10.0
- Language/Interpreter : Python 3
- Editor : Jupyter Notebook 6.1.4
- OS : Windows 10 x64

Libraries

```
# Importing Useful Libraries
import pandas as pd
import numpy as np
import matplotlib.cm as cm
import matplotlib.pyplot as plt
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Dropout, Activation, Conv2D, MaxPooling2D,BatchNormalization
from tensorflow.keras import optimizers
from keras.models import load_model
from tensorflow.keras.preprocessing import image
import os
from os import listdir
import random
import scipy
import pylab as pl
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import warnings
warnings.filterwarnings("ignore")
```

Libraries and Packages used:

- Matplotlib for visualization.
- Numpy – To process the image matrices
- os – To access the file system to read the image from the train and test directory from our machines
- Tensorflow – to create large-scale neural networks with many layers
- Random – To shuffle the data to overcome the biasing

MODEL/s DEVELOPMENT AND EVALUATION

IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS)

```
Train_data_dir=r'Garments/Train'
validation_data_dir=r'Garment/Test'
```

```
# Seeing the number of classes in the training folder
file = os.listdir(r"Garments/Train")
file
```

```
['Jeans (men)', 'Sarees (women)', 'Trousers (men)']
```

- From the above we can observe that it is a classification problem and there are three categories which are Jeans, Sarees and Trousers.
- So it's a deep learning problem where we have to build a Convolutional Neural Network (CNN) model for images classification.

RUN AND EVALUATE SELECTED MODELS

Build a CNN Model for Image classification

```
# Developing a convolutional neural network to classify images for correct labels
model=Sequential()

# First convolution layer
model.add(Conv2D(32,(3,3),input_shape=input_shape))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

# Second convolution layer
model.add(Conv2D(32,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

# Third convolution layer
model.add(Conv2D(64,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
```

```
# Fourth convolution layer
model.add(Conv2D(64,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))


model.add(Flatten())
model.add(Dense(128))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(3))
model.add(Activation('softmax'))

print(model.summary())

model.compile(loss='categorical_crossentropy',optimizer='rmsprop',metrics=['accuracy'])
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 574, 574, 32)      896

 activation (Activation)     (None, 574, 574, 32)      0

 max_pooling2d (MaxPooling2D  (None, 287, 287, 32)     0
 )

 dropout (Dropout)           (None, 287, 287, 32)      0

 conv2d_1 (Conv2D)           (None, 285, 285, 32)      9248

 activation_1 (Activation)   (None, 285, 285, 32)      0

 max_pooling2d_1 (MaxPooling  (None, 142, 142, 32)     0
 2D)

 dropout_1 (Dropout)         (None, 142, 142, 32)      0

 conv2d_2 (Conv2D)           (None, 140, 140, 64)      18496

 activation_2 (Activation)   (None, 140, 140, 64)      0

 max_pooling2d_2 (MaxPooling  (None, 70, 70, 64)       0
 2D)

 dropout_2 (Dropout)         (None, 70, 70, 64)        0


 conv2d_3 (Conv2D)           (None, 68, 68, 64)        36928

 activation_3 (Activation)   (None, 68, 68, 64)        0

 max_pooling2d_3 (MaxPooling  (None, 34, 34, 64)       0
 2D)

 dropout_3 (Dropout)         (None, 34, 34, 64)        0

 flatten (Flatten)           (None, 73984)             0

 dense (Dense)               (None, 128)               9470080

 activation_4 (Activation)   (None, 128)               0

 dropout_4 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 3)                 387

 activation_5 (Activation)   (None, 3)                 0

=================================================================
Total params: 9,536,035
Trainable params: 9,536,035
Non-trainable params: 0
```

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

Selenium is an open-source web-based automation tool. Python language is used with Selenium for testing. It has far less verbose and easy to use than any other programming language. The Python APIs empower you to connect with the browser through Selenium.

 Web driver:- We all know that we need to have browser drivers, .exe files like chromedriver.exe in case of windows environment or binary files like chrome driver in distributions, in order to run our selenium web driver automation scripts on chrome browsers.

And also we need to set the path of these files in our script like below or need to add location to the class path.

 Import shutil -Python shutil module enables us to operate with file objects easily and without diving into file objects a lot. It takes care of low-level semantics like creating file objects, closing the files once they are copied and allows us to focus on the business logic of our program

 Import requests:- Requests will allow you to send HTTP/1.1 requests using Python. With it, you can add content like headers, form data, multipart files, and parameters via simple Python libraries. It also allows you to access the response data of Python in the same way.

 Import Tensorflow:- Tensorflow is an open-source library developed by Google primarily for deep learning applications. It also supports traditional machine learning. TensorFlow was originally developed for large numerical computations without keeping deep learning in mind.

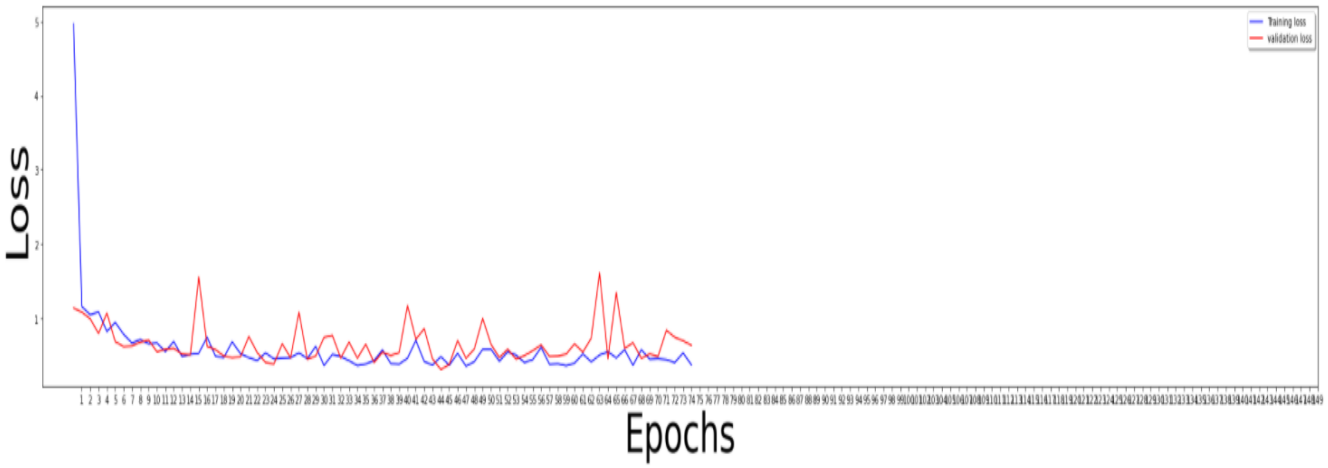
 Import os :- The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality. Path* modules include many functions to interact with the file system.

 Import pickle:-Pickle is used for serializing and de-serializing Python object structures, also called marshalling or flattening. Serialization refers to the process of converting an object in memory to a byte stream that can be stored on disk or sent over a network

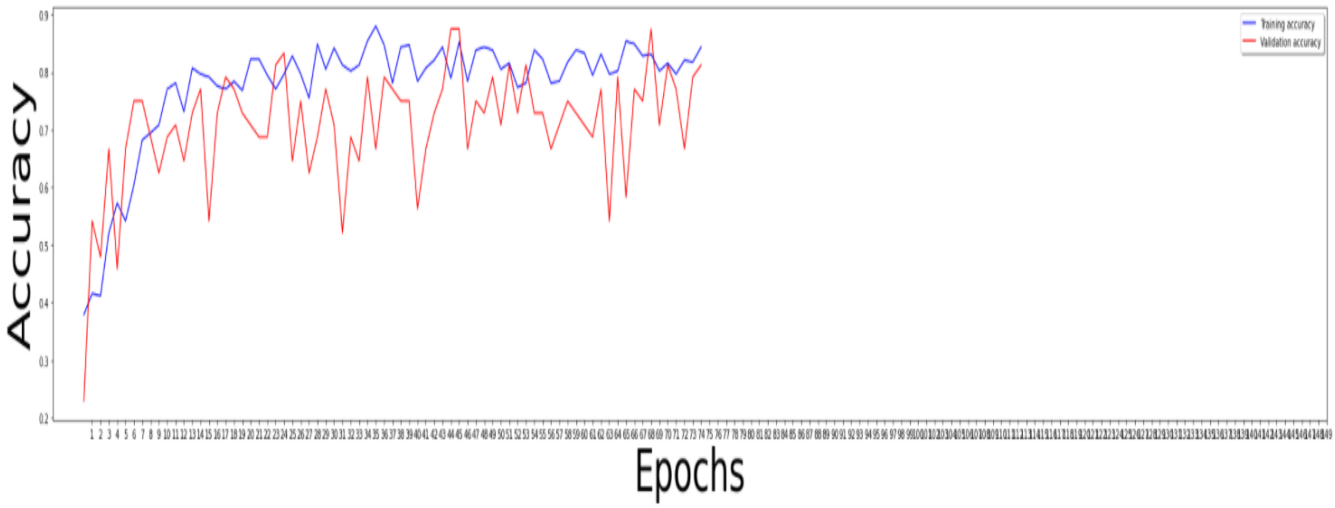
VISUALIZATION

```
# Visualizing  Training
import matplotlib.pyplot as plt
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(30, 12))
ax1.plot(history.history['loss'], color='b', label="Training loss")
ax1.plot(history.history['val_loss'], color='r', label="validation loss")
ax1.set_xticks(np.arange(1, epoch, 1),)
ax1.set_xlabel('Epochs' ,fontsize=50)
ax1.set_ylabel('Loss' ,fontsize=50)
ax1.legend(loc='best', shadow=True)

ax2.plot(history.history['accuracy'], color='b', label="Training accuracy")
ax2.plot(history.history['val_accuracy'], color='r',label="Validation accuracy")
ax2.set_xlabel('Epochs' ,fontsize=50)
ax2.set_ylabel('Accuracy' ,fontsize=50)
ax2.set_xticks(np.arange(1, epoch, 1))

ax2.legend(loc='best', shadow=True)
plt.tight_layout()
plt.show()
```

Blue Lines - Training Accuracy Red Lines - Validation Accuracy



- Losses are decreasing as number of epochs are increasing.



- Accuracy is increasing as number of epochs are increasing.

Input Image is: img_111.jpeg

Predicted Label is: sarees

Input Image is: img_131.jpeg

Predicted Label is: sarees

- Here we can see the few loaded images of Sarees which we scrapped from e-commerce website (Amazon) and our classification model predicted the label exactly the same.

Input Image is: img_551.jpeg

Predicted Label is: jeans

Input Image is: img_381.jpeg

Predicted Label is: jeans

- Here we can see the few loaded images of Jeans which we scrapped from e-commerce website (Amazon) and our classification model predicted the label exactly the same.

Input Image is: img_221.jpeg

Predicted Label is: trouser

Input Image is: img_301.jpeg

Predicted Label is: trouser

- Here we can see the few loaded images of Trousers which we scrapped from e-commerce website (Amazon) and our classification model predicted the label exactly the same.

INTERPRETATION OF RESULTS

- On epoch 75, we get the val_accuracy of 87.50% and val_loss of 37.43%.
- With increase in number of epochs, accuracy increases and losses decreases.
- The output we got when prepared data for training was that it found 718 images of train and 157 images of test data belonging to 3 classes.
- Most of the images were predicted correctly when all images were displayed as output in the jupyter notebook.

CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

From the whole project evaluation these are the inferences that I could draw from the visualization of data.

- The Convolutional Neural Network (CNN), a machine learning algorithm is being used for the image classification.
- The roles of epochs in DNN was able to control accuracy and also prevent any problems such as overfitting.
- Images used in the training purpose are small.
- The Deep neural network (DNN) becomes the main agenda for this research, especially in image classification technology.
- Data format plays a very important role in the visualization.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

Object detection is a computer vision task that involves both localizing one or more objects within an image and classifying each object in the image.It is a challenging computer vision task that requires both successful object localization in order to locate and draw a bounding box around each object in an image, and object classification to predict the correct class of object that was localized.
Using deep learning convolutional neural networks. While the dataset is effectively solved, it can be used as the basis for learning and practicing how to develop, evaluate, and use convolutional deep learning neural networks for image classification from scratch.
This includes how to develop a robust test harness for estimating the performance of the model, how to explore improvements to the model, and how to save the model and later load it to make predictions on new data.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

Stacking the model with more layers and training the network with more image data using clusters of GPUs will provide more accurate results of classification of images.

We can also play around by changing different parameters and discovering how we would get the best accuracy and score. Try changing the batch size, the number of epochs or even adding/removing layers in the CNN model.

Especially, the multi-dimensional input vector image WDIN can effectively avoid the complexity of data reconstruction in the process of feature extraction and image classification