FLIP ROBO

Image Scraping and Classification Project

Using

Deep Learning

Submitted By :

Shivani Angadi

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my trainers at Data Trained “Dr. Deepika Sharma” and my trainer for Deep Learning for being such a fantastic coach as well as my SME at Flip Robo Technologies “Mr. Shubham Yadav " who gave me the golden opportunity to do this wonderful project on the topic “Image Scraping and Classification Project”, which helped me in doing a lot of research and exploring so many new things. Also, I would like to show appreciation to “Mr. Sajid Choudhary” for guiding me throughout and helping me solve issues regarding this project.

I am really thankful to Flip Robo Technologies for giving me this prime opportunity of interning in their organization and enhancing my skills.

Secondly, I want to thank Data Trained Academy for giving me the best possible study notes, online platforms such as www.medium.com, www.stackoverflow.com, www.scikit-learn.org and www.github.com for providing me with the resources when I stumbled and fell along the way.
It helped me increase my knowledge and skills.

THANKS AGAIN TO ALL WHO SUPPORTED.

Introduction

Business Problem

In the discipline of data science and AI, images are one of the most crucial sources of data. Relevant use of information can be assembled through images by investigating its features and details. Deep Learning has helped many businesses to solve big issues in their industry and earn great profits.

Background Domain Knowledge

Classification of Images is one of the foremost domains of Artificial Intelligence. Neural Networks play a vital role in Image Classification. It gives an output by giving probability to an input image saying it belongs to a particular class. It is a procedure in which a set of pixels are arranged and tagged or grouping of vectors on the basis of rules defined. The two classification methods used are “pattern recognition” and “clustering of images”.

Literature Review

Computer Vision/ Deep Learning is an area of Artificial Intelligence that educates the system to learn and illuminate the picture of the real visual world. It extracts useful information from Images, audios and videos and provides us with the content of the knowledge learnt from any of the above files. Computer Vision applications include Image Detection, Classification, Segmentation, Verification, Recognition, Identification, etc.

Project Motivation

The motivation of the project is to build a deep learning model with the help of Convolutional Neural Networks which inturn is useful in large areas of domains to solve problems.

Analytical Problem Framing

Analytical Modeling of Problem

“Image Scraping and Classification Project” comprises two phases i.e. collection of cata using web scraping and model building using deep learning mechanisms. Number of data scrapped for each category is 300 images wherein there are 3 different categories which gives a total data of 900 images. Using image augmentation and pre-processing, a model is built to classify images. Problem is approached in the following way :

1. **Project Domain Research -** Since this task was to build knowledge in the field of Computer Vision and deep Learning, domain research was not a part of this project.
2. **Collecting Data -** Data is scrapped using tools such as Selenium and BeautifulSoup. There are 3 categories of clothing that were scrapped.
3. **Image Preprocessing -** Image transformations and augmentations were performed as a part of data pre-processing. Each data was was resized to a size of 224*224.
4. **Model Building -** Transfer Learning technique is used to train the training data set where the weights and biases are pre-trained & learned on the “ImageNet” data set. This helps in fast learning of Images.
5. **Interpreting Solutions -** Once the images are prepared for processing, they are passed through the model in batches and classification accuracy is obtained.

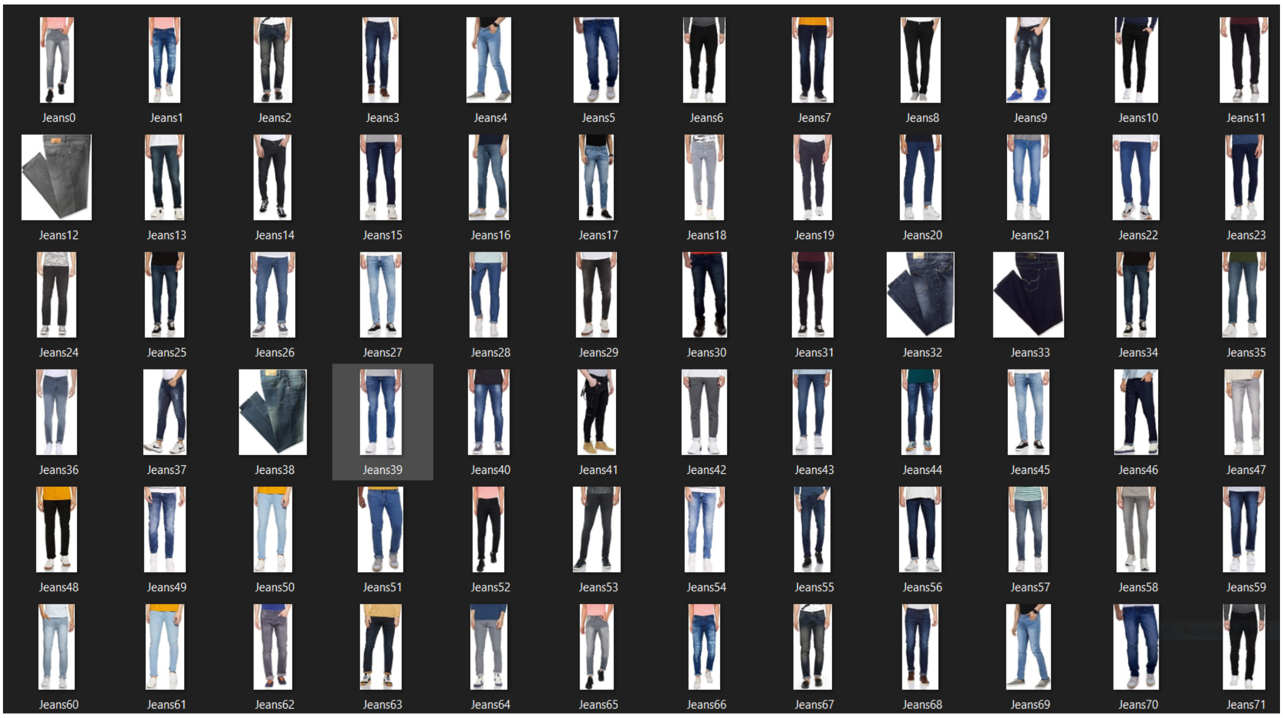
Data Sources

Data scraping is done from an e-commerce website, www.amazon.com. Total data of 900 images have been retrieved. It is said that more the amount of data, better the performance of the model, therefore better the results. Tools used for Image Scraping are Selenium and BeautifulSoup. Three categories of clothing data were scrapped from the website, those are:

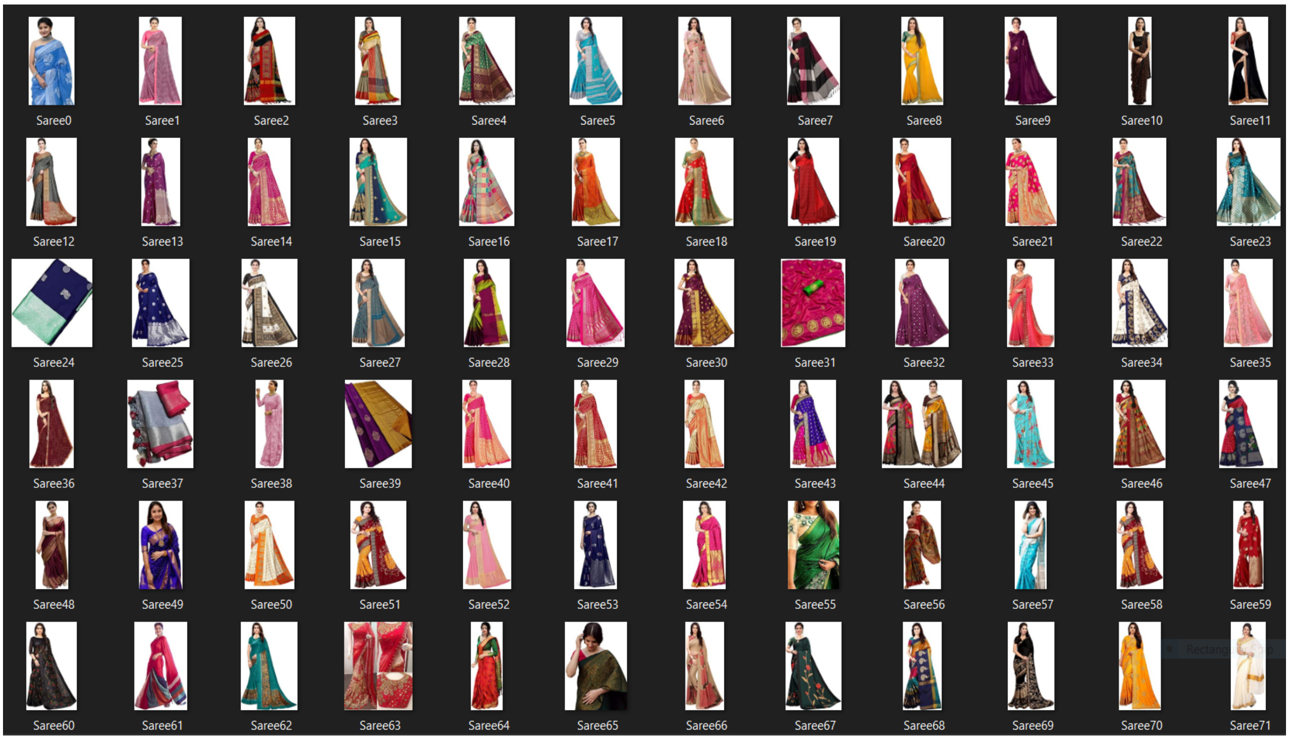
1. Sarees (Women)
2. Trousers (Men)
3. Jeans (Men)

This is how the data directory looks like:

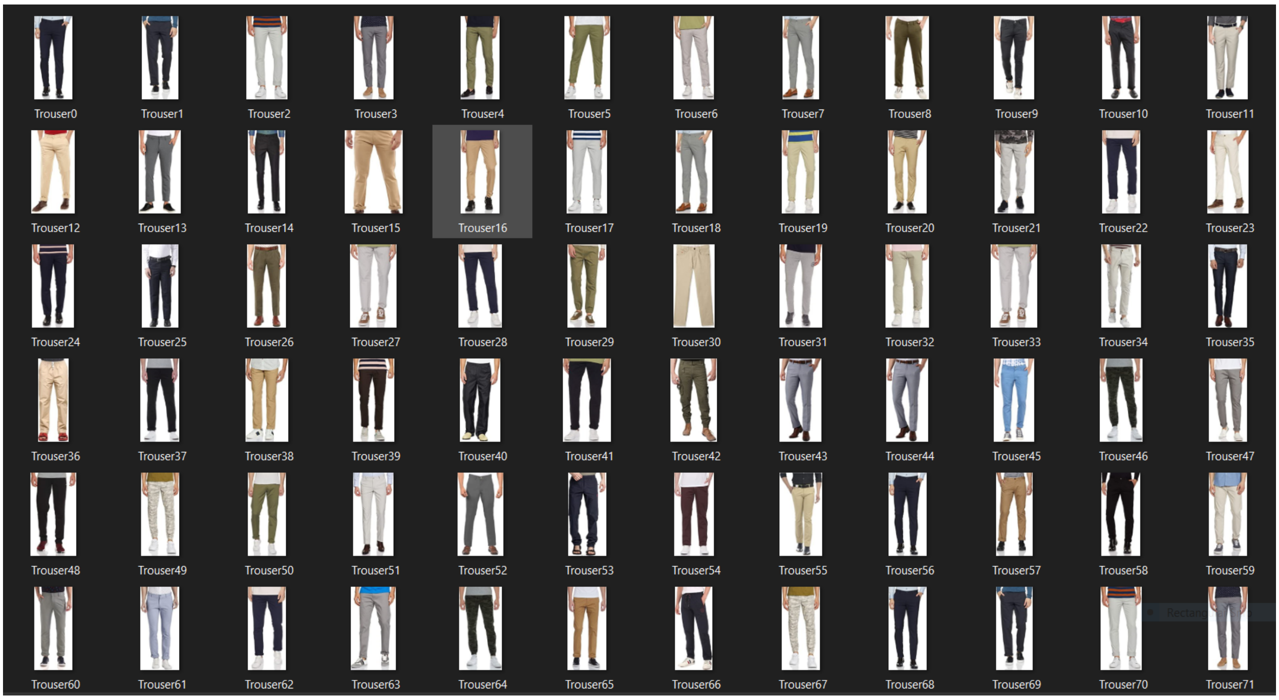
1. Jeans (Men)


Jeans0 Jeans1 Jeans2 Jeans3 Jeans4 Jeans5 Jeans6 Jeans7 Jeans8 Jeans9 Jeans10 Jeans11
Jeans12 Jeans13 Jeans14 Jeans15 Jeans16 Jeans17 Jeans18 Jeans19 Jeans20 Jeans21 Jeans22 Jeans23
Jeans24 Jeans25 Jeans26 Jeans27 Jeans28 Jeans29 Jeans30 Jeans31 Jeans32 Jeans33 Jeans34 Jeans35
Jeans36 Jeans37 Jeans38 Jeans39 Jeans40 Jeans41 Jeans42 Jeans43 Jeans44 Jeans45 Jeans46 Jeans47
Jeans48 Jeans49 Jeans50 Jeans51 Jeans52 Jeans53 Jeans54 Jeans55 Jeans56 Jeans57 Jeans58 Jeans59
Jeans60 Jeans61 Jeans62 Jeans63 Jeans64 Jeans65 Jeans66 Jeans67 Jeans68 Jeans69 Jeans70 Jeans71

2. Sarees (Women)

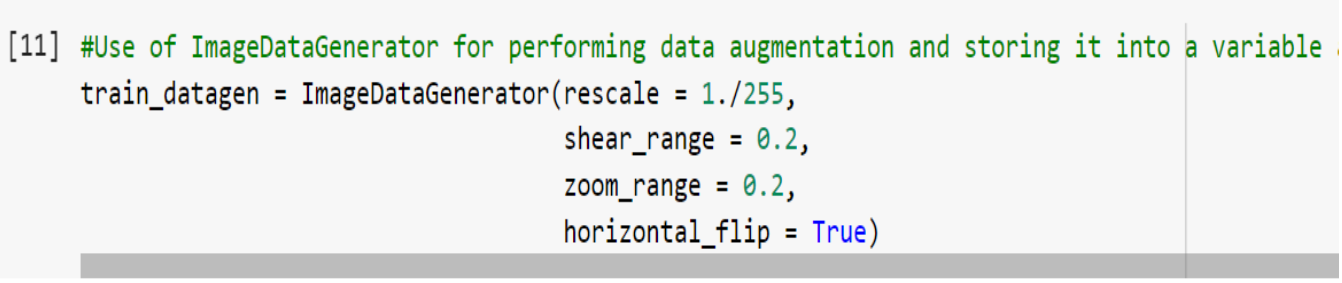

Saree0 Saree1 Saree2 Saree3 Saree4 Saree5 Saree6 Saree7 Saree8 Saree9 Saree10 Saree11
Saree12 Saree13 Saree14 Saree15 Saree16 Saree17 Saree18 Saree19 Saree20 Saree21 Saree22 Saree23
Saree24 Saree25 Saree26 Saree27 Saree28 Saree29 Saree30 Saree31 Saree32 Saree33 Saree34 Saree35
Saree36 Saree37 Saree38 Saree39 Saree40 Saree41 Saree42 Saree43 Saree44 Saree45 Saree46 Saree47
Saree48 Saree49 Saree50 Saree51 Saree52 Saree53 Saree54 Saree55 Saree56 Saree57 Saree58 Saree59
Saree60 Saree61 Saree62 Saree63 Saree64 Saree65 Saree66 Saree67 Saree68 Saree69 Saree70 Saree71

3. Trousers (Men)

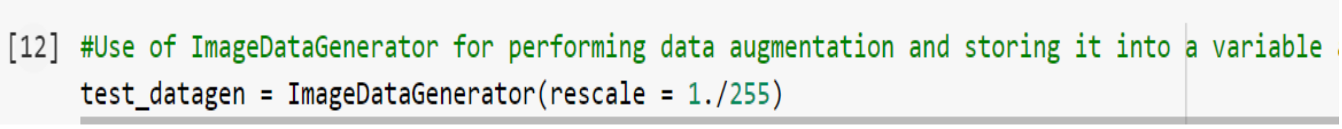


Data Pre-processing

Image pre-processing is performed using ImageDataGenerator in which images are transformed by resizing them and image augmentation is performed where images are flipped, zoomed and sheared. Code representation of image pre-processing is shown in the below image



```
[11] #Use of ImageDataGenerator for performing data augmentation and storing it into a variable
     train_datagen = ImageDataGenerator(rescale = 1./255,
                                        shear_range = 0.2,
                                        zoom_range = 0.2,
                                        horizontal_flip = True)
```



```
[12] #Use of ImageDataGenerator for performing data augmentation and storing it into a variable
     test_datagen = ImageDataGenerator(rescale = 1./255)
```

Hardware and Software Requirements

Hardware

1. 16 GB RAM - used for data storage and training models.

2. CPU - used for executing algorithms for Images.

Software

1. Pandas - A fast, powerful and flexible open source tool used for data manipulation and data analysis.

2. Numpy - Numpy is an open source library used for mathematical and computational analysis of data and works best with multidimensional arrays and matrices.

3. Matplotlib - It works like MATLAB and helps pyplot function in making some changes to a figure such as creating a figure, creating plotting area, etc.

4. Google Colab - It is a product researched by google which provides good memory and RAM and free open source GPU to process image data sets and contains all libraries just as a Jupyter Notebook.

Assumptions

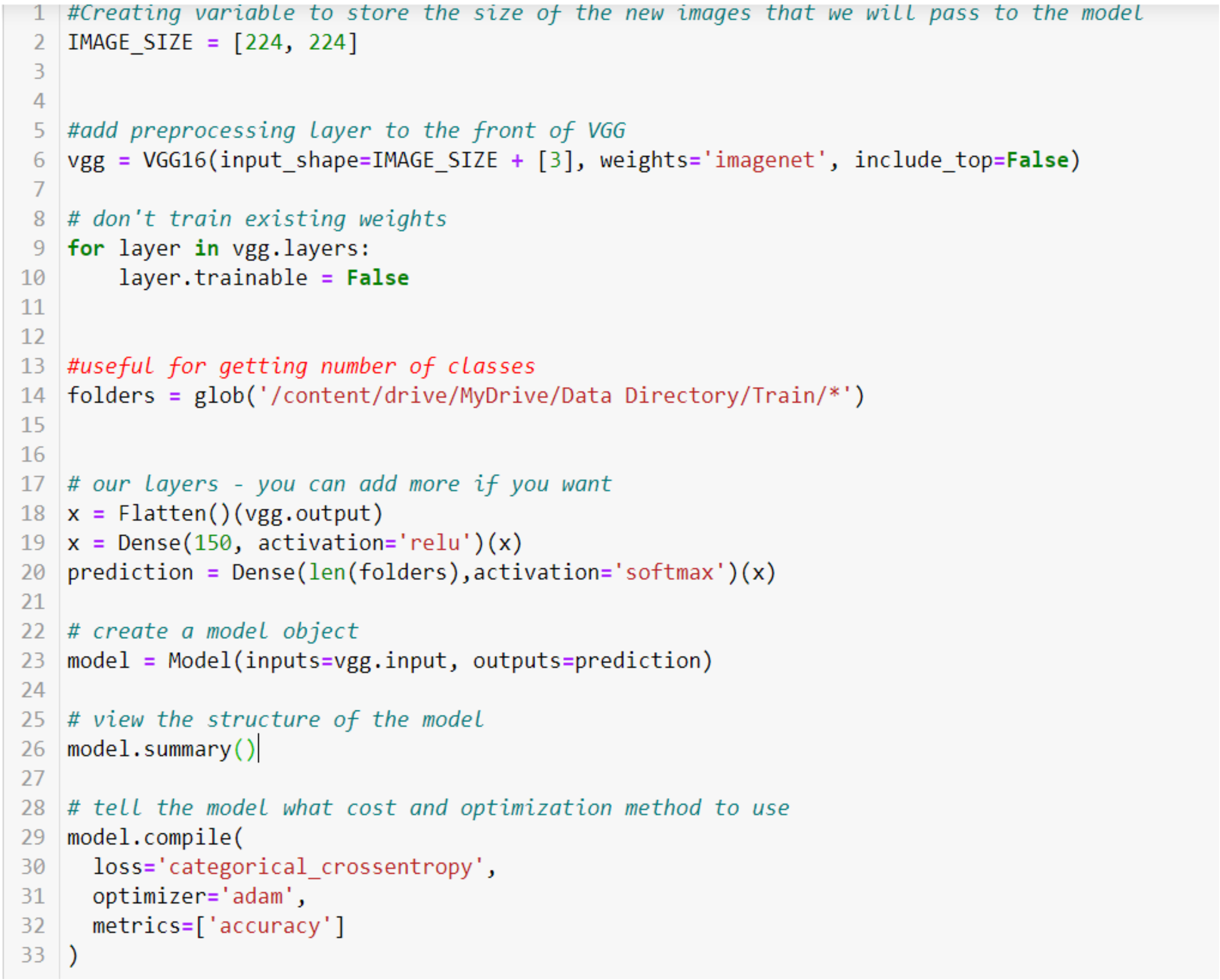
- An assumption was made that the images will take a long execution time since images are bulky data which takes time to process.

- Good accuracy was not expected since it takes a huge amount of data to attain the best accuracy of models.

Model Development & Evaluation

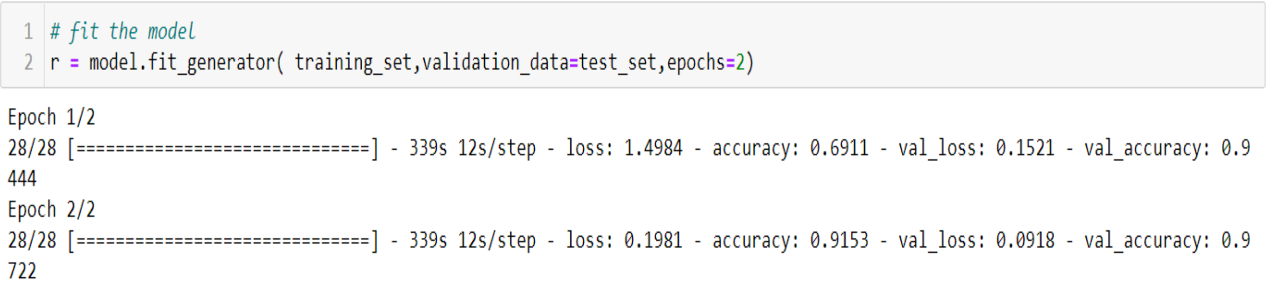
Problem Solving Approach

- The first stage of Model Development was to provide image size to the input parameter.
- Use of Transfer Learning is made in order to train images since the features are pre-learnt and the model already has the knowledge of the features that our data set images might contain.
- This data was previously trained on the “ImageNet” data set.
- On the top of the “ImageNet” model, application of output layers in accordance with the required data set using “Relu” activation function for hidden layers and “Softmax” for classifying images as outputs.
- The algorithm used for building model is VGG16. Code for model building is shown below:



```
#Creating variable to store the size of the new images that we will pass to the model
IMAGE_SIZE = [224, 224]


#add preprocessing layer to the front of VGG
vgg = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

# don't train existing weights
for layer in vgg.layers:
    layer.trainable = False


#useful for getting number of classes
folders = glob('/content/drive/MyDrive/Data Directory/Train/*')


# our layers - you can add more if you want
x = Flatten()(vgg.output)
x = Dense(150, activation='relu')(x)
prediction = Dense(len(folders),activation='softmax')(x)

# create a model object
model = Model(inputs=vgg.input, outputs=prediction)

# view the structure of the model
model.summary()

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

Results -



```
# fit the model
r = model.fit_generator( training_set,validation_data=test_set,epochs=2)
```

```
Epoch 1/2
28/28 [==============================] - 339s 12s/step - loss: 1.4984 - accuracy: 0.6911 - val_loss: 0.1521 - val_accuracy: 0.9
444
Epoch 2/2
28/28 [==============================] - 339s 12s/step - loss: 0.1981 - accuracy: 0.9153 - val_loss: 0.0918 - val_accuracy: 0.9
722
```

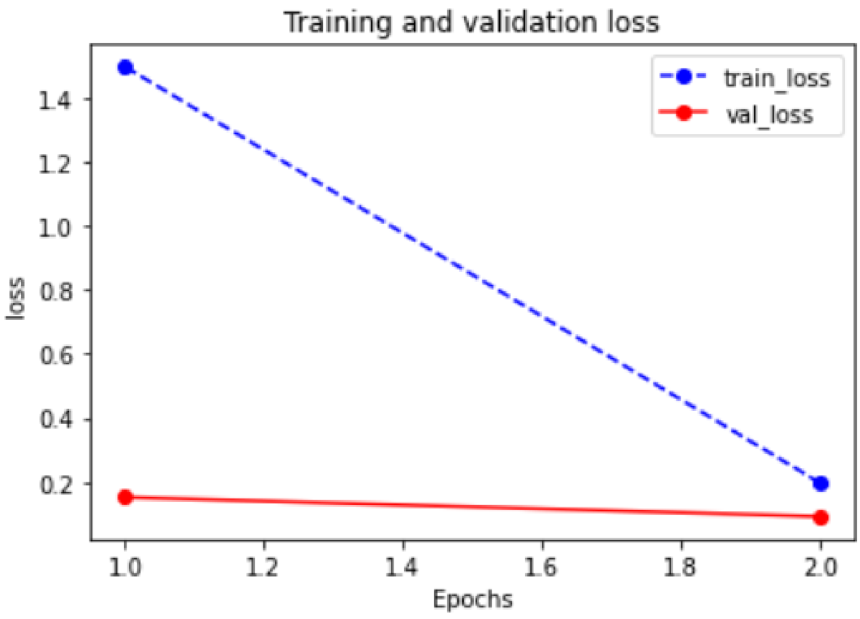
Conclusion -

The accuracy obtained for Image Classification after applying different learning rates and epochs is 97% validation accuracy and 91% training accuracy. In this case, both testing and validation accuracy are the same. Data is trained on just 2 epochs and still a good accuracy was retained. After the 1st epoch, training accuracy is 69% and validation accuracy is 94%. Therefore, we could notice a high jump in accuracies of the 1st epoch and the second epoch.

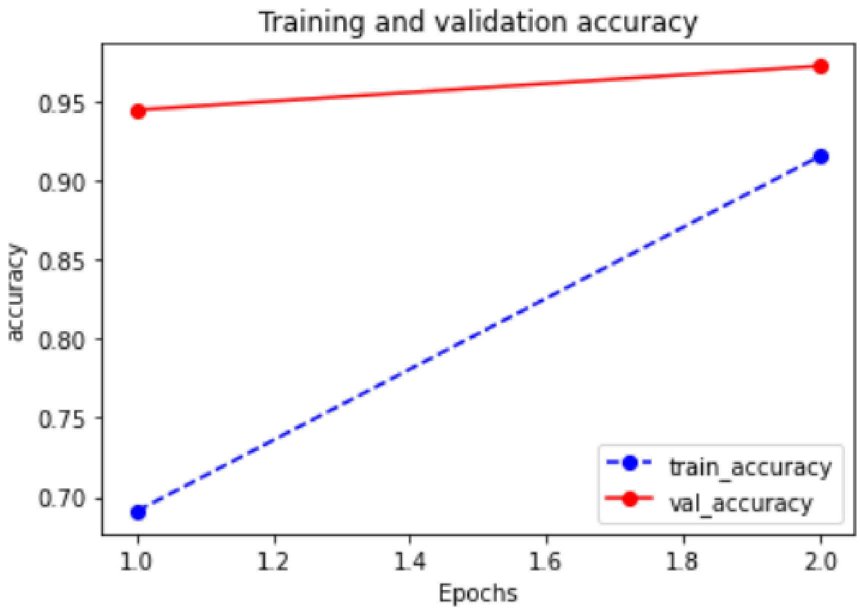
Visualizations

- **Line Charts -**

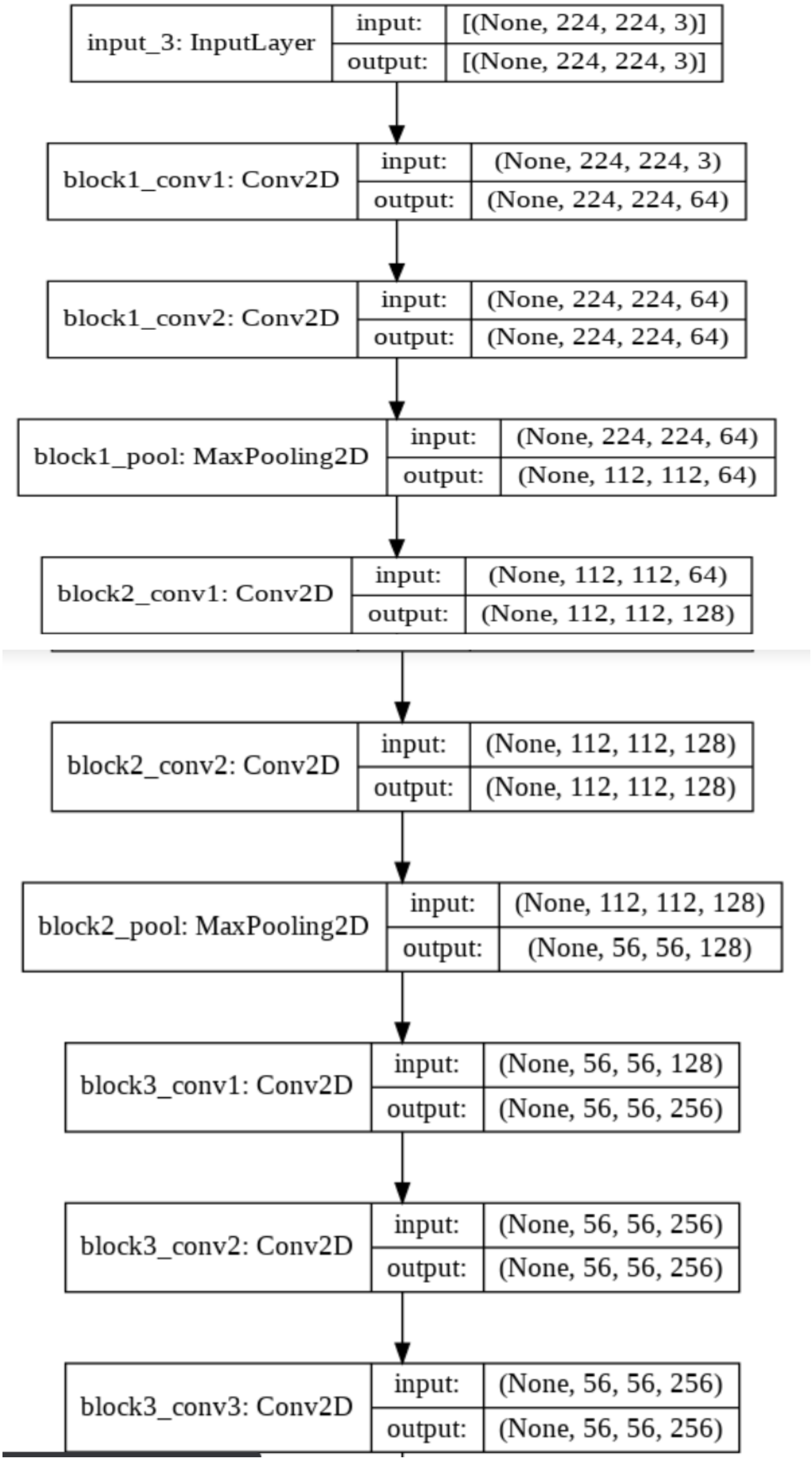
Plotting of train loss and validation loss per epoch has been depicted in the below image.

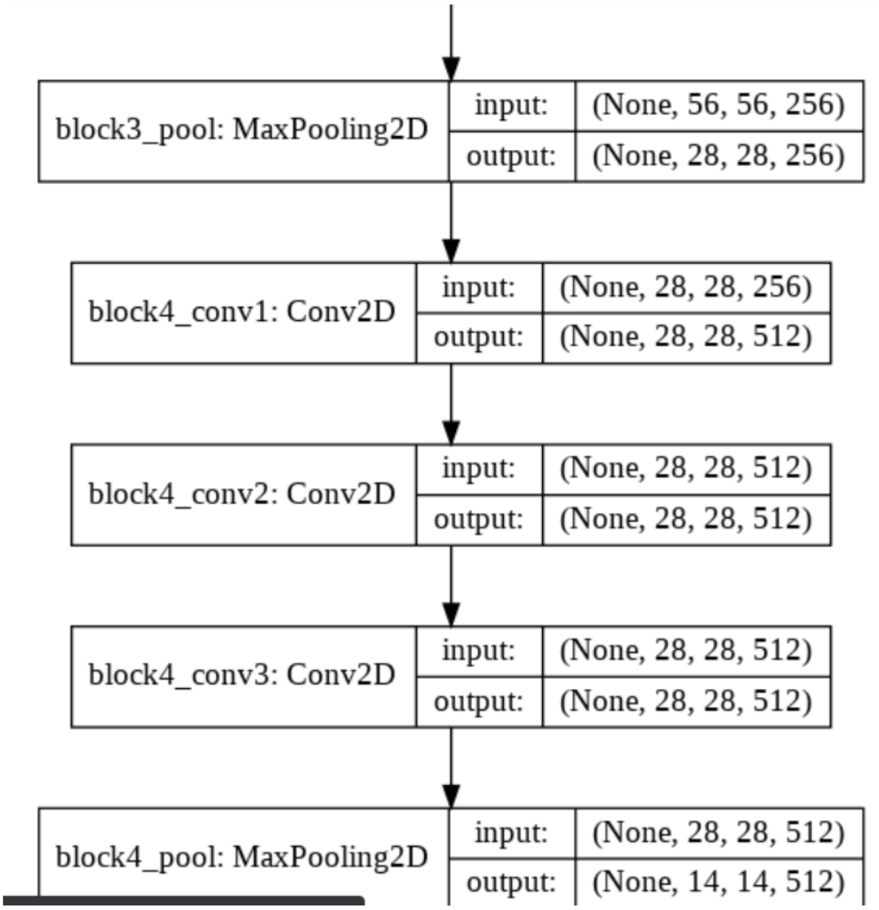


Representation of training accuracy and validation accuracy per epoch in the image below



- **Flowchart -**


input_3: InputLayer
input:
[(None, 224, 224, 3)]
output:
[(None, 224, 224, 3)]
block1_conv1: Conv2D
input:
(None, 224, 224, 3)
output:
(None, 224, 224, 64)
block1_conv2: Conv2D
input:
(None, 224, 224, 64)
output:
(None, 224, 224, 64)
block1_pool: MaxPooling2D
input:
(None, 224, 224, 64)
output:
(None, 112, 112, 64)
block2_conv1: Conv2D
input:
(None, 112, 112, 64)
output:
(None, 112, 112, 128)
block2_conv2: Conv2D
input:
(None, 112, 112, 128)
output:
(None, 112, 112, 128)
block2_pool: MaxPooling2D
input:
(None, 112, 112, 128)
output:
(None, 56, 56, 128)
block3_conv1: Conv2D
input:
(None, 56, 56, 128)
output:
(None, 56, 56, 256)
block3_conv2: Conv2D
input:
(None, 56, 56, 256)
output:
(None, 56, 56, 256)
block3_conv3: Conv2D
input:
(None, 56, 56, 256)
output:
(None, 56, 56, 256)



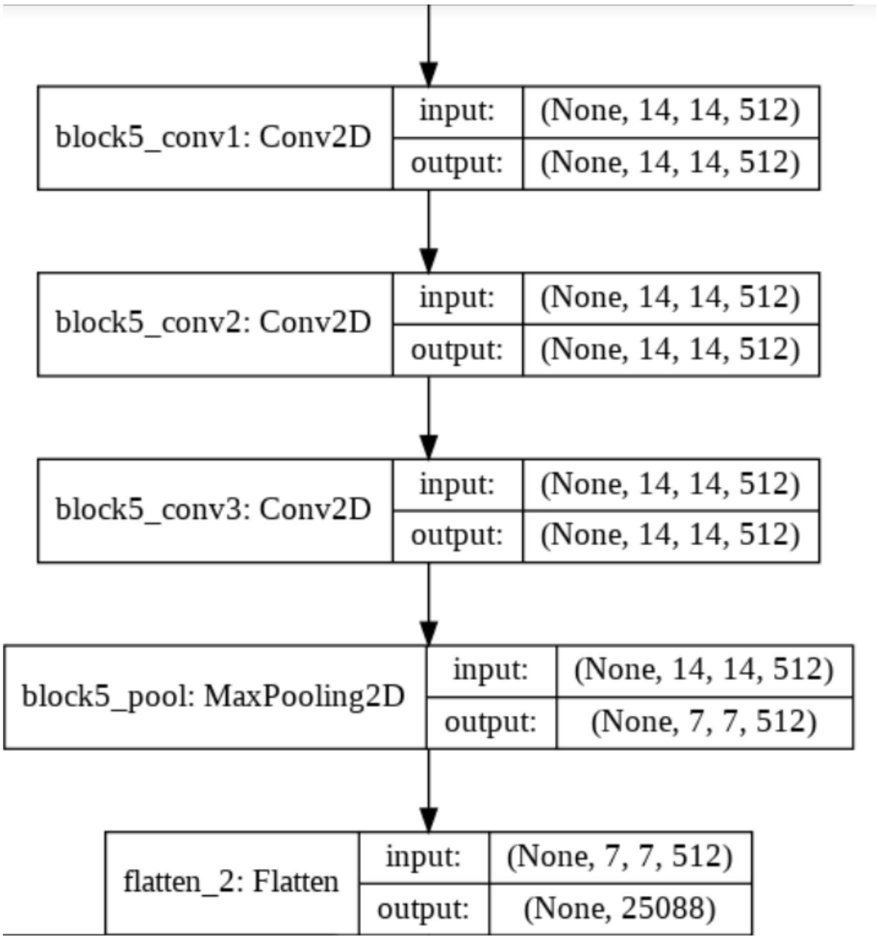
block3_pool: MaxPooling2D | input: (None, 56, 56, 256) | output: (None, 28, 28, 256)

block4_conv1: Conv2D | input: (None, 28, 28, 256) | output: (None, 28, 28, 512)

block4_conv2: Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)

block4_conv3: Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)

block4_pool: MaxPooling2D | input: (None, 28, 28, 512) | output: (None, 14, 14, 512)



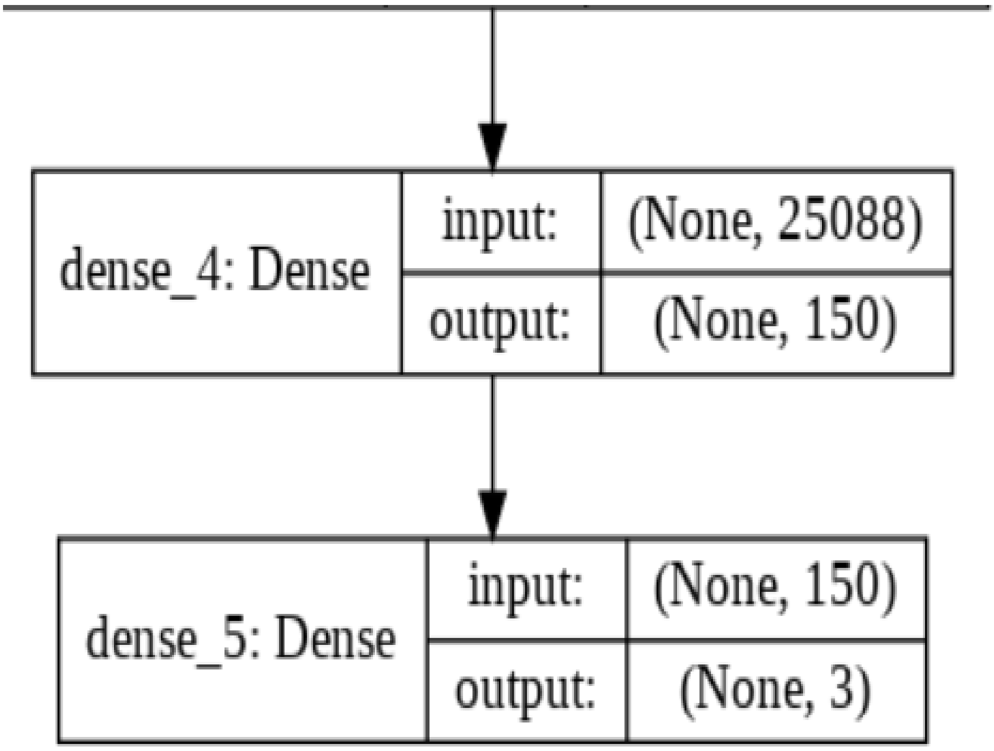
block5_conv1: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)

block5_conv2: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)

block5_conv3: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)

block5_pool: MaxPooling2D | input: (None, 14, 14, 512) | output: (None, 7, 7, 512)

flatten_2: Flatten | input: (None, 7, 7, 512) | output: (None, 25088)



The above model has been put into use for training of images which helped in attaining a good accuracy.

Conclusion

- Updation of learning rates and epochs for training of images was one of the biggest challenges faced while model building since training consumes a lot of time.

- Also few issues came up regarding downloading new libraries, functions and resolving issues. Trying out different optimizers and different algorithms such as resnet and googlenet was done to check which gave the best accuracy, but VGG16 proved to give the best accuracy with 97% validation accuracy on the testing data set. And as proven, “adam” optimizer was the best in optimizing the gradients of the data and extracting the right features from the images.

- The limitation of this project is that the model cannot be standardized since it has been built considering only three categories of class labels and that too with the help of transfer learning.