IMAGE SCRAPING AND CLASSIFICATION PROJECT

Submitted by:

Arpan Pattanayak

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my SME Mr. Shwetank Mishra from Flip Robo Technologies for letting me work on this project and providing me with all necessary information and dataset for the project. I would also like to thank my mentor of Data Trained academy for providing me sufficient knowledge so that I can complete the project.

I have utilized a few external resources that helped me to complete the project. I ensured that I learn from the samples and modify things according to my project requirement. All the external resources that were used in creating this project are listed below:

- https://www.google.com/
- https://www.youtube.com/
- https://scikit-learn.org/stable/user_guide.html
- https://github.com/
- https://www.analyticsvidhya.com/

INTRODUCTION

BUSINESS PROBLEM FRAMING

Classification is a systematic arrangement in groups and categories based on its features. Image classification came into existence for decreasing the gap between the computer vision and human vision by training the computer with the data. The image classification is achieved by differentiating the image into the prescribed category based on the content of the vision. We use Deep Learning to accomplish the task of image Classification.

In this project we have to classify whether is image is of a jeans , a trouser or a saree. We could we such model either for automatic segmentation of items using IOT and techniques for Industry 4.0

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

Deep learning (DL) is a sub field to the machine learning, capable of learning through its own method of computing. A deep learning model is introduced to persistently break down information with a homogeneous structure like how a human would make determinations. To accomplish this, deep learning utilizes a layered structure of several algorithms expressed as an artificial neural system (ANN). The architecture of an ANN is simulated with the help of the biological neural network of the human brain. This makes the deep learning most capable than the standard machine learning models.

In deep neural networks every node decides its basic inputs by itself and sends it to the next tier on behalf of the previous tier. We train the data in the networks by giving an input image and conveying the network about its output. Neural networks are expressed in terms of number of layers involved for producing the inputs and outputs and the depth of the neural network.

REVIEW OF LITERATURE

Recently, 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 (Faux & Luthon, 2012).

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 colour, 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 project we will be using a transfer learning state of the art model for getting the best results that is VGG 16. Which was a winner in the ImageNet Competition?

MOTIVATION FOR THE PROBLEM UNDERTAKEN

The problem was undertaken in order to classify images efficiently using best deep learning algorithms and data augmentation techniques.

ANALYTICAL PROBLEM FRAMING

MATHEMATICAL/ ANALYTICAL MODELING OF THE PROBLEM

We have scraped images of all 3 classes that are men's jeans, men's trouser and sarees for women from the internet and we have built our model by training it on this data. We have used transfer learning to get state of the art results for our model.

DATA SOURCES AND THEIR FORMATS

DATA COLLECTION PHASE:

Data has been scraped from **amazon.in** using a python script which with selenium. All the data is in the **.jpg image format**.

We have over 368 images per class.

MODEL BUILDING PHASE:

After the data collection and preparation is done, you need to build an image classification model that will classify between these 3 categories mentioned above. You can play around with optimizers and learning rates for improving your model's performance.

DATA PREPROCESSING DONE

- We have first labelled the image data.
- We have manually removed irrelevant images that were downloading via the script.
- We have removed the duplicate images using a script.
- We have also performed multiple data augmentation techniques in order to train the model better or multiple angles and orientation of the same image.

DATA INPUTS- LOGIC- OUTPUT RELATIONSHIPS

We have given raw yet cleaned images to the machine learning model and the output produced is a label of the image on which the model is predicted

HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED

HARDWARE:

1. Processor — core i5 and above
2. RAM — 8 GB or above
3. SSD — 250GB or above

SOFTWARE:

i. Programming language: Python
ii. Distribution: Anaconda Navigator
iii. Browser based language shell: Jupyter Notebook

LIBRARIES:

- Numpy
- Pandas
- Tensorflow
- MAtplotlib
- Keras.savemodel

MODEL/S DEVELOPMENT AND EVALUATION

IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS)

- We will be first scraping the data from amazon and then cleaning the data to be further used for training the model.
- Then splitting the data into train and test set to model evaluation.
- We would perform multiple data augmentation techniques on the images for better generalization of our model.
- We could check and identity an optimal batch size and number of epoch to Train the model using trial and error method also keeping the epoch loss for both training set and validation in mind.

TESTING OF IDENTIFIED APPROACHES (ALGORITHMS)

```
In [3]: # Let's try to print some of the scrapped images from each category
        import matplotlib.image as mpimg
        import matplotlib.pyplot as plt

        train_jeans=r'Clothes/Train/Jeans_Images'
        train_saree=r'Clothes/Train/Sarees_Images'
        train_trouser=r'Clothes/Train/Trousers_Images'
        sni

        Cloth_train=[train_jeans, train_saree, train_trouser]
        for dirs in Cloth_train:
            k=listdir(dirs)
            for i in k[:3]:
                img=mpimg.imread('{}/{}'.format(dirs,i))
                plt.imshow(img)
                plt.axis('off')
                plt.show()
```

RUN AND EVALUATE SELECTED MODELS

After collecting the data we next do training of the data. For that firstly, we created a main folder called “Clothes” in current working directory inside which I further created two folders called “Train” and “Test”.

In Train folder we have kept 250 images from each clothing category and remaining 118 images we have kept in Test folder for each category. Hence, we got 750 images for training and 354 for testing.

In [1]:
```
import os
from os import listdir
```

In [2]:
```
train_data=r'Clothes/Train'
test_data=r'Clothes/Test'
```

In [3]:
```
# Let's try to print some of the scrapped images from each category
import matplotlib.image as mpimg
import matplotlib.pyplot as plt

train_jeans=r'Clothes/Train/Jeans_Images'
train_saree=r'Clothes/Train/Sarees_Images'
train_trouser=r'Clothes/Train/Trousers_Images'
sni

Cloth_train=[train_jeans, train_saree, train_trouser]
for dirs in Cloth_train:
    k=listdir(dirs)
    for i in k[:3]:
        img=mpimg.imread('{}/{}'.format(dirs,i))
        plt.imshow(img)
        plt.axis('off')
        plt.show()
```


en=9ad891197a3ce8...




In [6]:
```
print("Count of Training Images")
print("No.of Images of Sarees in train dataset -> ",len(os.listdir(r'
print("No.of Images of Jeans in train dataset -> ",len(os.listdir(r'C
print("No.of Images of Trousers in train dataset ->",len(os.listdir(r
"\n"

print("Count of Test Images")
print("No.of Images of Sarees in test dataset-> ",len(os.listdir(r'Cl
print("No.of Images of Jeans in test dataset ->",len(os.listdir(r'Clo
print("No.of Images of Trousers in test dataset-> ",len(os.listdir(r'
```

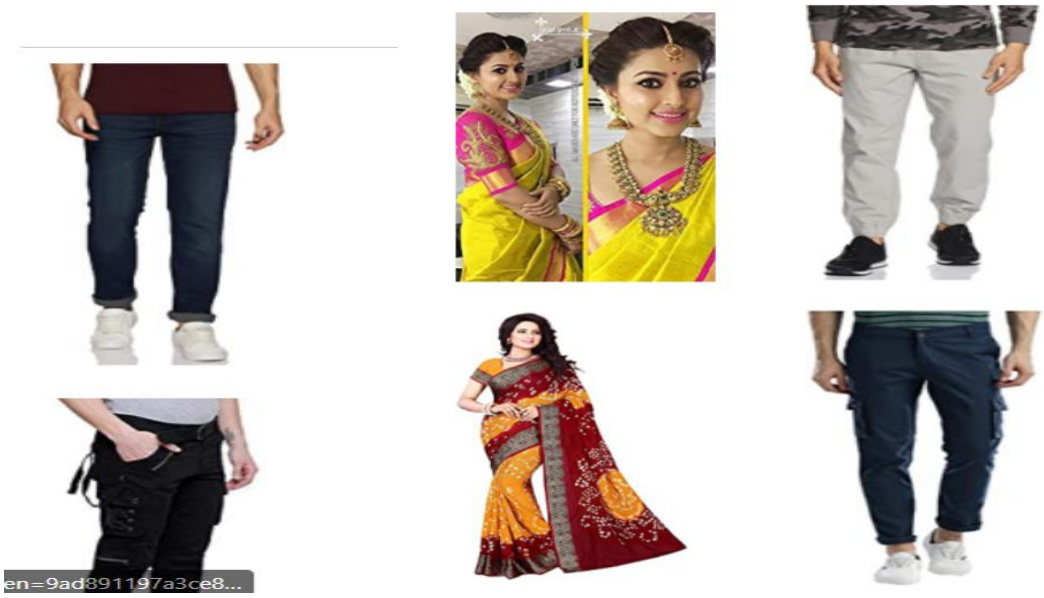
```
Count of Training Images
No.of Images of Sarees in train dataset ->  250
No.of Images of Jeans in train dataset ->  250
No.of Images of Trousers in train dataset -> 250
Count of Test Images
No.of Images of Sarees in test dataset->  118
No.of Images of Jeans in test dataset -> 118
No.of Images of Trousers in test dataset->  118
```

Next we defined dimensions of images and other parameters also. Then, for data augmentation we defined training and testing set.

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDE CONSIDERATION

We have considered the model accuracy and loss for both the training and validation data.

VISUALIZATION



FINAL MODEL

In [9]:
```
# Data Augmentation on Training Images

Train_datagen=ImageDataGenerator(rescale=1./255,
                                 zoom_range=0.2,
                                 rotation_range=30,
                                 horizontal_flip=True)
Training_set=Train_datagen.flow_from_directory(train_data,
                                               target_
                                               batch_s
                                               class_m

# Test Data Generator
Test_datagen=ImageDataGenerator(rescale=1./255)
Test_set=Test_datagen.flow_from_directory(test_data,
                                          target_size=(img_wi
                                          batch_size=batch_si
                                          class_mode='categor
```

```
Found 750 images belonging to 3 classes.
Found 354 images belonging to 3 classes.
```

In [10]:
```
# Creating the model
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='adam',metric
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 126, 126, 32)      896
_________________________________________________________________
activation (Activation)      (None, 126, 126, 32)      0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 63, 63, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 63, 63, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 61, 61, 32)        9248
_________________________________________________________________
activation_1 (Activation)    (None, 61, 61, 32)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 30, 30, 32)        0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 12, 12, 64)        36928
_________________________________________________________________
activation_3 (Activation)    (None, 12, 12, 64)        0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
dropout_3 (Dropout)          (None, 6, 6, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 2304)              0
_________________________________________________________________
dense (Dense)                (None, 128)               295040
_________________________________________________________________
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: 360,995
Trainable params: 360,995
Non-trainable params: 0
_________________________________________________________________
None
```

In [12]:
```
# Fitting the Training Data
history = model.fit(
    Training_set,
    epochs=epoch,
    validation_data=Test_set,
    validation_steps=test_samples//batch_size,
    steps_per_epoch=train_samples//batch_size,
    callbacks=[ES,MC])
```

```
Epoch 1/100
25/25 [==============================] - 18s 677ms/step - loss: 1.10
78 - accuracy: 0.4200 - val_loss: 1.0333 - val_accuracy: 0.6667

Epoch 00001: val_accuracy improved from -inf to 0.66667, saving mode
l to best.h5
Epoch 2/100
25/25 [==============================] - 9s 348ms/step - loss: 0.862
5 - accuracy: 0.5680 - val_loss: 0.8543 - val_accuracy: 0.6111

Epoch 00002: val_accuracy did not improve from 0.66667
Epoch 3/100
25/25 [==============================] - 8s 324ms/step - loss: 0.702
4 - accuracy: 0.6293 - val_loss: 0.6255 - val_accuracy: 0.6944

Epoch 00003: val_accuracy improved from 0.66667 to 0.69444, saving m
odel to best.h5
Epoch 4/100
```

In [14]:
```
losses = pd.DataFrame(model.history.history)
losses
```

Out[14]:

	loss	accuracy	val_loss	val_accuracy
0	1.054061	0.479167	1.010016	0.453704
1	0.897520	0.504167	0.831043	0.675926
2	0.754331	0.604167	0.766594	0.638889
3	0.609279	0.658333	0.541394	0.759259
4	0.589435	0.679167	0.611278	0.638889
...	...	...	...	...
57	0.311112	0.858333	0.455440	0.824074
58	0.340405	0.870833	0.430164	0.787037
59	0.308678	0.862500	0.407513	0.814815
60	0.340133	0.862500	0.398927	0.814815
61	0.280360	0.866667	0.369472	0.842593

62 rows × 4 columns

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

- When it comes to the evaluation of a data science model's performance, sometimes accuracy may not be the best indicator.
- Some problems that we are solving in real life might have a very imbalanced class and using accuracy might not give us enough confidence to understand the algorithm's performance.
- In the Rating Prediction problem that we are trying to solve, the data is balanced. So accuracy score nearly tells the right predictions. So the problem of overfitting in this problem is nearly not to occur. So here, we are using an accuracy score to find a better model.

CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

Our Model was able to classify the three clothing items distinctly. Since in all three categories there were some extra/unnecessary items other than the main items hence, it could have been removed and we could have got better result. Moreover, training data could have been increased.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

The larger the data the better the model could predict. Multi class predictions are somewhat relatively harder to train in comparison to a Binary class prediction. Data Augmentation is necessary where we have small datasets of images.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

One could always provide more data for training the model in order to get better results. We could use to model for apparel segmentation at Supermarkets and shopping stores.