IMAGE SCRAPING AND CLASSIFICATION

Submitted by:

Vivek Rai

ACKNOWLEDGMENT

I would like to express my special thanks of grattitude to the sources Medium, TowardsDataScience, StackOverflow, Krish Naik's youtube channel which helped me to accomplish this 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 Model Building.

- ## Conceptual Background of the Domain Problem

  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 project, 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.

- ## Review of Literature

  Computational models of neural networks have been around for a long time, first model proposed was by McCulloch and Pitts.

Neural networks are made up of a number of layers with each layer connected to the other layers forming the network. A feed-forward neural network or FFNN can be thought of in terms of neural activation and the strength of the connections between each pair of neurons.

In FFNN, the neurons are connected in a directed way having clear start and stop place i.e., the input layer and the output layer. The layer between these two layers, are called as the hidden layers. Learning occurs through adjustment of weights and the aim is to try and minimize error between the output obtained from the output layer and the input that goes into the input layer. The weights are adjusted by process of back propagation (in which the partial derivative of the error with respect to last layer of weights is calculated). The process of weight adjustment is repeated in a recursive manner until weight layer connected to input layer is updated.

- Motivation for the Problem Undertaken

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.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  Firstly we label the images in our dataset according to their categories as 0 for jeans, 1 for sarees and 2 for trousers.

  The images in our dataset are of shape 320, 137, 3 we resize it to the shape 180, 180, 3 as shown in figure below.

```
img.shape
```

```
(320, 137, 3)
```

```
cv2.resize(img,(180,180)).shape
```

```
(180, 180, 3)
```

```
X, y = [], []

for clothe_name, images in clothes_images_dict.items():
    for image in images:
        img = cv2.imread(str(image))
        resized_img = cv2.resize(img,(180,180))
        X.append(resized_img)
        y.append(clothes_labels_dict[clothe_name])
```

  Then we scale images by dividing them with 255 before feeding it to our neaural network.

- Data Sources and their formats

  There are three categories of images present in our dataset they are jeans, saree and trouser labeled as 0, 1 and 2 respectively.

- Data Preprocessing Done

  In the preprocessing part we resize the images present in our dataset from 320, 137, 3 to 180, 180, 3.

  Then we scale the images of both the training and test set before feeding it to the neaural network.

- Data Inputs- Logic- Output Relationships

  There are only two variables present in our image classification dataset, images and their labels.

  The images are labeled into their categories as 0, 1 and 2 accordingly.

- State the set of assumptions (if any) related to the problem under consideration

  After knowing that it was image classification problem we assumed that we would be feeding the dataset to convolutunal neural network for better accuracy.

- Hardware and Software Requirements and Tools Used

  This project was done on laptop with i5 processor with quad cores and eight threads with 8gb of ram and latest GeForce GTX 1650 GPU on Anaconda, jupyter notebook.

  The tools, libraries and packages we used for accomplishing this project are numpy, matplotlib, seaborn, pathlib, cv2, sklearn, keras.

With the help of pathlib we specified path of our dataset images.

With the help of numpy we worked with arrays.

With the help of matplotlib and seaborn we did plot various graphs and figures and done data visualization.

With sklearn's train test split we split the dataset into training set and testing set.

Through cv2 we read our dataset and resized the shape of it.

Through keras we preprocessed our dataset.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  We first label the images in our dataset according to their categories as 0 for jeans, 1 for sarees and 2 for trousers.

  Then we did some visualization of our dataset.

  The images in our dataset are of shape 320, 137, 3 we resize it to the shape 180, 180, 3.

  Then we scale images by dividing them with 255 before feeding it to our neaural network.

  We noticed that our prior model was overfitting so with the help of data augmentation techniques we increased the number of images in our dataset before feeding it to the convolutional neural network.

- Testing of Identified Approaches (Algorithms)

  The algorithms we used for the training and testing is convolutional neural network.

  We played with some parameters increased the number of epochs from 10 to 30 used the activation function relu to get the desired accuracy.

  We found out that our model was overfitting so with the help of data augmentation techniques we increased the size of our dataset and fed it to the convolutional neural network and the desired accuracy was achieved in our test dataset.

  We added maxpooling to our layers and increased the number of neurons from 16 to 64 and also added a dropout layer to our neural network.

  For our model development we used the optimizer adam and for loss we used SparseCtaegoricalCrossentropy.

- Run and Evaluate selected models

  The algorithms we used are shown below,

```
num_classes = 3

model = Sequential([
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

model.fit(X_train_scaled, y_train, epochs=30)
```

  Then after applying some data augmentation techniques we added dropuout layer in our convolutional neural network.

```
num_classes = 3

model = Sequential([
  data_augmentation,
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Dropout(0.2),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

model.fit(X_train_scaled, y_train, epochs=30)
```

- Key Metrics for success in solving problem under consideration

  On the basis of accuracy and F1 score we choose convolutional neural network after data augmentation with dropout layer being applied.

- Visualizations

  Visualizing the image of our first category which is jeans with help of PIL,

```
PIL.Image.open(str(jeans[1]))
```

Visualizing the image of our second category which is saree with help of PIL,

```
saree = list(data_dir.glob('saree/*'))
PIL.Image.open(str(saree[0]))
```

Visualizing the image of our third category which is trouser with help of PIL,

```
trouser = list(data_dir.glob('trouser/*'))
PIL.Image.open(str(trouser[0]))
```

- Interpretation of the Results

  The results observed over different evaluation metrics are shown below,

```
Epoch 30/30
20/20 [==============================] - 10s 496ms/step - loss: 0.2680 - accuracy: 0.9024

<tensorflow.python.keras.callbacks.History at 0x21cd88b3970>
```

```
model.evaluate(X_test_scaled,y_test)
```

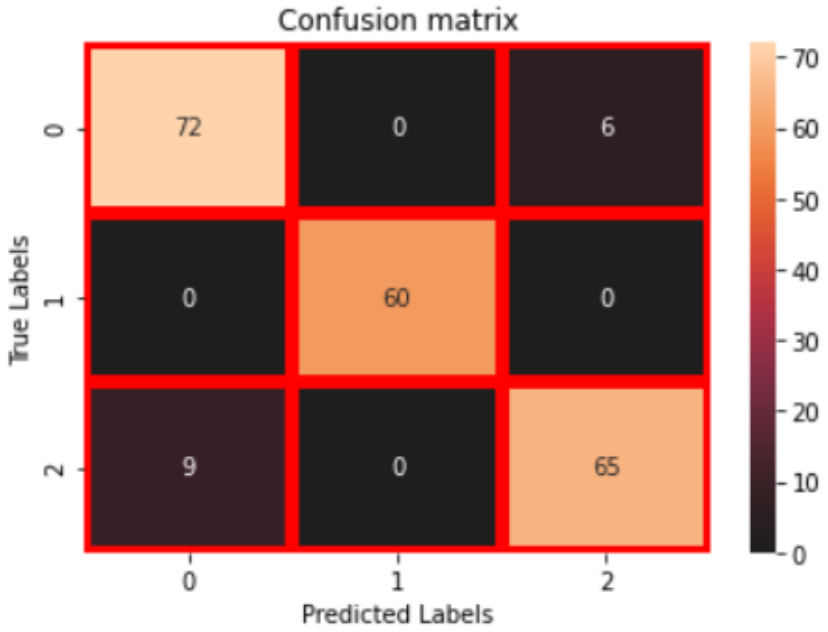
```
7/7 [==============================] - 1s 74ms/step - loss: 0.1801 - accuracy: 0.9292

[0.18007293343544006, 0.9292452931404114]
```

```
classification_report
              precision    recall  f1-score   support

           0       0.89      0.92      0.91        78
           1       1.00      1.00      1.00        60
           2       0.92      0.88      0.90        74

    accuracy                           0.93       212
   macro avg       0.93      0.93      0.93       212
weighted avg       0.93      0.93      0.93       212
```



After performing data augmentatioin techniques and adding a dropuout layer to our dataset we observed that our accuracy of testing dataset improved from 88% to 92.92%.

CONCLUSION

- Key Findings and Conclusions of the Study

  In conclusion, this research is about image classification by using deep learning via framework TensorFlow. The roles of epochs in

DNN was able to control accuracy and also prevent any problems such as overfitting. Thus we were able to classify the three categories of images with high testing accuracy of 92.92%.

- ## Learning Outcomes of the Study in respect of Data Science

  It has three (3) 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. 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.

- ## Limitations of this work and Scope for Future Work

  While we couldn't reach out goal of 100% accuracy in image classification, we did end up creating a system that can with enough time and data get very close to that goal. As with any project there is room for improvement here. This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. This provides a great degree of modularity and vesatility to the project.