CLOTH-CLASSIFICATION

Submitted by:

SANDES P

ACKNOWLEDGMENT

I used informative tutorials to follow some steps in the task from websites like geeksforgeeks, stackoverflow, etc.

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 categories. 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.

· Conceptual Background of the Domain Problem

Our goal is to determine the image is of a saree, jeans, or a pair of trousers.

· Review of Literature

Nowadays, every domain requires machine learning to simplify its jobs. Here our objective is to find out the category of the image which is whether saree, jeans, or trousers. This can be used to match(validate) the product image and title while uploading to the e-commerce website by the seller.

· Motivation for the Problem Undertaken

To make a model that predicts whether the cloth is a saree, jeans, or trousers, we should use a convolution neural network. First, we have to fetch some product images from the e-commerce website Amazon. We scraped images of saree, jeans, and trousers from “amazon.in” using the python Beautifulsoup library. We scraped and saved it in different folders. We fed these images to the CNN to train the model.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  Here we tried to use Sequential modeling to create the classification model.

- Data Sources and their formats

  The data required to build this model is scraped from "www.amazon.in". The data is scraped and saved in PNG format. The data consist of 400 images in each category.

- Data Preprocessing Done

  - Read the images using cv2 library
  - Labeled each image category through a for loop
  - One hot encoded the output values

- Hardware and Software Requirements and Tools Used

  - 8GB RAM
  - i5 7th gen processor

  Softer requirements

  - Python
  - Jupyter notebook

  Libraries

  - Pandas
  - Numpy
  - Matplotlib
  - Seaborn
  - Re
  - Sklearn

- Joblib
- Tensorflow
- CV2

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
    - One hot Encoding
- Testing of Identified Approaches (Algorithms)
    - Sequential

· Run and Evaluate selected models

```
[ ]
# creating a sequential model
model= Sequential()
model.add(Conv2D(kernel_size=(3,3), filters=32, activation='tanh', input_shape=(200,200,3,)))
model.add(Conv2D(filters=30,kernel_size = (3,3),activation='tanh'))
model.add(MaxPool2D(2,2))
model.add(Conv2D(filters=30,kernel_size = (3,3),activation='tanh'))
model.add(MaxPool2D(2,2))
model.add(Conv2D(filters=30,kernel_size = (3,3),activation='tanh'))

model.add(Flatten())

model.add(Dense(20,activation='relu'))
model.add(Dense(15,activation='relu'))
model.add(Dense(3,activation = 'softmax'))

model.compile(
              loss='categorical_crossentropy',
              metrics=['acc'],
              optimizer='adam'
             )
```

```
[ ]
# printing model summary
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 198, 198, 32)      896
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 196, 196, 30)      8670
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 98, 98, 30)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 96, 96, 30)        8130
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 48, 48, 30)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 46, 46, 30)        8130
_________________________________________________________________
flatten_1 (Flatten)          (None, 63480)             0
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1269620
_________________________________________________________________
dense_4 (Dense)              (None, 15)                315
_________________________________________________________________
dense_5 (Dense)              (None, 3)                 48
=================================================================
Total params: 1,295,809
Trainable params: 1,295,809
Non-trainable params: 0
```

```
1 # training the model
2 history = model.fit(x_train,y_train,epochs=50,batch_size=50,validation_data=(x_val,y_val))

Epoch 1/50
18/18 [==============================] - 112s 6s/step - loss: 2.7394 - acc: 0.5911 - val_loss: 0.6109 - val_acc: 0.8100
Epoch 2/50
18/18 [==============================] - 111s 6s/step - loss: 0.5428 - acc: 0.8089 - val_loss: 0.3695 - val_acc: 0.8400
Epoch 3/50
18/18 [==============================] - 111s 6s/step - loss: 0.4921 - acc: 0.8222 - val_loss: 0.4348 - val_acc: 0.8300
Epoch 4/50
18/18 [==============================] - 110s 6s/step - loss: 0.3468 - acc: 0.8611 - val_loss: 0.4500 - val_acc: 0.8067
Epoch 5/50
18/18 [==============================] - 112s 6s/step - loss: 0.2988 - acc: 0.8833 - val_loss: 0.3697 - val_acc: 0.8433
Epoch 6/50
18/18 [==============================] - 112s 6s/step - loss: 0.2107 - acc: 0.9178 - val_loss: 0.3466 - val_acc: 0.8767
Epoch 7/50
18/18 [==============================] - 111s 6s/step - loss: 0.1674 - acc: 0.9322 - val_loss: 0.3493 - val_acc: 0.8900
Epoch 8/50
18/18 [==============================] - 111s 6s/step - loss: 0.1691 - acc: 0.9344 - val_loss: 0.4498 - val_acc: 0.8500
Epoch 9/50
18/18 [==============================] - 111s 6s/step - loss: 0.1816 - acc: 0.9256 - val_loss: 0.5399 - val_acc: 0.8633
Epoch 10/50
18/18 [==============================] - 112s 6s/step - loss: 0.1415 - acc: 0.9400 - val_loss: 0.3229 - val_acc: 0.8833
Epoch 11/50
18/18 [==============================] - 112s 6s/step - loss: 0.1166 - acc: 0.9500 - val_loss: 0.3269 - val_acc: 0.8867
Epoch 12/50
18/18 [==============================] - 112s 6s/step - loss: 0.0723 - acc: 0.9689 - val_loss: 0.3444 - val_acc: 0.8933
Epoch 13/50
18/18 [==============================] - 115s 6s/step - loss: 0.0872 - acc: 0.9722 - val_loss: 0.3696 - val_acc: 0.8900
Epoch 14/50
18/18 [==============================] - 113s 6s/step - loss: 0.0951 - acc: 0.9656 - val_loss: 0.3923 - val_acc: 0.9000
Epoch 15/50
18/18 [==============================] - 113s 6s/step - loss: 0.0524 - acc: 0.9833 - val_loss: 0.3707 - val_acc: 0.8967
Epoch 16/50
18/18 [==============================] - 113s 6s/step - loss: 0.0340 - acc: 0.9889 - val_loss: 0.3668 - val_acc: 0.9167
Epoch 17/50
18/18 [==============================] - 113s 6s/step - loss: 0.0259 - acc: 0.9922 - val_loss: 0.3711 - val_acc: 0.9100
Epoch 18/50
18/18 [==============================] - 114s 6s/step - loss: 0.0218 - acc: 0.9944 - val_loss: 0.3662 - val_acc: 0.9233
Epoch 19/50
18/18 [==============================] - 113s 6s/step - loss: 0.0135 - acc: 0.9967 - val_loss: 0.3136 - val_acc: 0.9267
Epoch 20/50
18/18 [==============================] - 113s 6s/step - loss: 0.0153 - acc: 0.9956 - val_loss: 0.3605 - val_acc: 0.9233
Epoch 21/50
18/18 [==============================] - 113s 6s/step - loss: 0.0135 - acc: 0.9967 - val_loss: 0.3289 - val_acc: 0.9300
Epoch 22/50
```

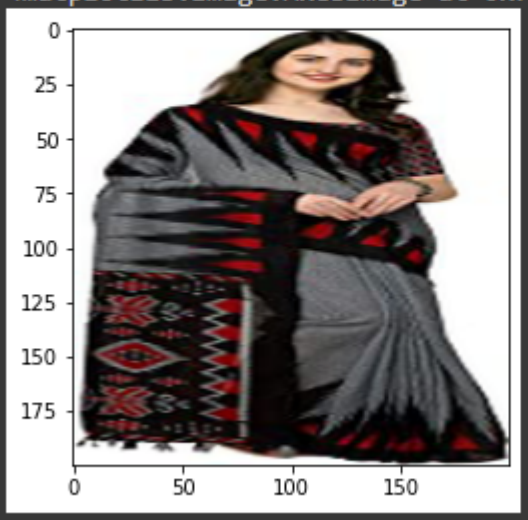
- Key Metrics for success in solving problem under consideration

Here we used categorical cross-entropy loss to find out the accuracy of the model.

· Visualizations

```
[ ]  1 # visualizing training data
     2 print(train_labels[0])
     3 plt.imshow(train_images[0])
```

```
[0 1 0]
<matplotlib.image.AxesImage at 0x7f02eaefd390>
```




```
[ ]  1 # visualizing training data
     2 print(train_labels[1100])
     3 plt.imshow(train_images[1100])
```

```
[0 0 1]
<matplotlib.image.AxesImage at 0x7f02eaed9a90>
```

CONCLUSION

- Key Findings and Conclusions of the Study

  Trained the images through a convolution network to identify the object is whether saree, jeans or trousers. We were able to achieve an accuracy of 93%.

- Learning Outcomes of the Study in respect of Data Science

  Got the opportunity to work with CNN and sequential model