Image Scraping and Classification project

By

Allen Paul

Introduction

- Classification between the objects is easy task for humans but it has proved to be a complex problem for machines. The raise of high capacity computers, the availability of high quality and low priced video cameras and the increasing need for automatic video analysis has generated an interest in object classification algorithms.
- Image Classification System consists of database that contains predefined patterns that compares with detected object to classify in to proper category.

Problem Statement

- 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

The convolutional neural network (CNN) is a class of deep learning neural networks. CNNs represent a huge breakthrough in image recognition. They’re most commonly used to analyse visual imagery and are frequently working behind the scenes in image classification. They can be found at the core of everything from Facebook’s photo tagging to self-driving cars. They’re working hard behind the scenes in everything from healthcare to security. They’re fast and they’re efficient.

Motivation for the Problem Undertaken

Since this task of recognizing a visual concept (e.g. cat) is relatively trivial for a human to perform, it is worth considering the challenges involved from the perspective of a Computer Vision algorithm. As we present (an inexhaustive) list of challenges below, keep in mind the raw representation of images as a 3-D array of brightness values:

- Viewpoint variation. A single instance of an object can be oriented in many ways with respect to the camera.
- Scale variation. Visual classes often exhibit variation in their size (size in the real world, not only in terms of their extent in the image).
- Deformation. Many objects of interest are not rigid bodies and can be deformed in extreme ways.
- Occlusion. The objects of interest can be occluded. Sometimes only a small portion of an object (as little as few pixels) could be visible.
- Illumination conditions. The effects of illumination are drastic on the pixel level.
- Background clutter. The objects of interest may *blend* into their environment, making them hard to identify.
- Intra-class variation. The classes of interest can often be relatively broad, such as *chair*. There are many different types of these objects, each with their own appearance.

A good image classification model must be invariant to the cross product of all these variations, while simultaneously retaining sensitivity to the inter-class variations.

Dataset

- In this Project there are two phases: Data Collection and Mode Building.
- **Data Collection Phase:** In this section, we have scraped images from Amazon.com. The clothing categories used for scraping are:

- Sarees (women)
- Trousers (men)
- Jeans (men)

- **Model Building Phase:** Using the above data we need to build an image classification model that will classify between these 3 categories mentioned above

Saree Images sample

249saree.jpg

250saree.jpg

251saree.jpg

252saree.jpg

253saree.jpg

254saree.jpg

255saree.jpg

256saree.jpg

Jeans Images sample

249jeans.jpg

250jeans.jpg

251jeans.jpg

252jeans.jpg

253jeans.jpg

254jeans.jpg

255jeans.jpg

256jeans.jpg

Trousers Images sample

Data Pre-processing Done

CNN Architecture:

Images and photos are structured differently from your normal data stored in CSVs and contain even more information if they are coloured. This means that we will have to first transform photos into arrays of numbers that can be understood by the network

A CNN is made up of hidden layers that are attached to a fully-connected layer which handles the classification decisions based on the pixel information flowing through the previous layers

```python
# Import the Sequential model and layers
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense

model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(300, 300, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2), padding = 'same'))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

# the model so far outputs 3D feature maps (height, width, features)
```

```python
model.add(Flatten())  # this converts our 3D feature maps to 1D feature vectors
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(3))
model.add(Activation('sigmoid'))

model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optimizer='adam')
model.summary()

batch_size = 5
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 298, 298, 32)      896
_________________________________________________________________
activation (Activation)      (None, 298, 298, 32)      0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 149, 149, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 147, 147, 64)      18496
_________________________________________________________________
activation_1 (Activation)    (None, 147, 147, 64)      0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 73, 73, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 71, 71, 128)       73856
_________________________________________________________________
activation_2 (Activation)    (None, 71, 71, 128)       0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 35, 35, 128)       0
_________________________________________________________________
flatten (Flatten)            (None, 156800)            0
_________________________________________________________________
dense (Dense)                (None, 64)                10035264
_________________________________________________________________
activation_3 (Activation)    (None, 64)                0
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 195
_________________________________________________________________
activation_4 (Activation)    (None, 3)                 0
=================================================================
Total params: 10,128,707
Trainable params: 10,128,707
Non-trainable params: 0
_________________________________________________________________
```

- ## Data Inputs- Logic- Output Relationships

  In this problem the images of jeans, saree and trousers scraped from the amazon.in are the inputs and after training the model using deep learning we use this model to predict the probability of an image to be jeans, saree and trousers is the output.

- ## Hardware and Software Requirements and Tools Used

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense
from keras.preprocessing.image import ImageDataGenerator
import pandas as pd
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  CNN is one of the most useful techniques in solving the image classification problem. We must see to it that all the images have proper resolution, scaling and all the images should be of same size in order to make a model that predict accurately. Data pre-processing on the input/image data must be done properly.

- Testing of Identified Approaches (Algorithms)

  In this problem we have used ImageDataGenerator model in order to train the model.

Augmentation:

```
# Training Augmentation configuration
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

# Testing Augmentation - Only Rescaling
test_datagen = ImageDataGenerator(rescale = 1./255)

# Generates batches of Augmented Image data
#this is a generator that will read pictures found in C:\\Users\\allen\\train and indefinitely generate
#batches of augmented image data
train_generator = train_datagen.flow_from_directory('C:\\Users\\allen\\train',
                                                    target_size = (300, 300), # all images will be resized to 300x300
                                                    batch_size = batch_size,
                                                    class_mode = 'categorical')

# Generator for validation data
validation_generator = test_datagen.flow_from_directory('C:\\Users\\allen\\test',
                                                        target_size = (300, 300),
                                                        batch_size = batch_size,
                                                        class_mode = 'categorical')
```

```
Found 723 images belonging to 3 classes.
Found 60 images belonging to 3 classes.
```

We can see from above that there are 723 images in training set and 60 images in testing set and they belong to 3 classes. Namely saree, jeans and trouser. Having 20 images each in testing set and 241 images each in training.

Results

```
# Fit the model on Training data
model.fit_generator(train_generator,
                    epochs = 20,
                    validation_data = validation_generator,
                    verbose = 1)

# Evaluating model performance on Testing data
loss, accuracy = model.evaluate(validation_generator)

print("\nModel's Evaluation Metrics: ")
print("---------------------------")
print("Accuracy: {} \nLoss: {}".format(accuracy, loss))
```

```
Model's Evaluation Metrics:
---------------------------
Accuracy: 0.8333333134651184
Loss: 0.3408423066139221
```

Predicting data using the trained model:

```
# predicting the class of images using the trained model
predict_generator = ImageDataGenerator(rescale = 1./255)

predict_generator = predict_generator.flow_from_directory('C:\\Users\\allen\\predict',
                                                          target_size = (300, 300),
                                                          batch_size = batch_size,
                                                          class_mode = 'categorical')
```

```
Found 90 images belonging to 1 classes.
```

```
predictions = model.predict(predict_generator)
```

```
import pandas as pd
```

```
Predictions = pd.DataFrame(predictions, columns=['Jeans', 'Saree','Trouser'])
```

```
Predictions # predicted data
```

	Jeans	Saree	Trouser
0	9.094689e-01	0.048458	0.999477
1	9.999417e-01	0.001092	0.999627
2	7.693771e-01	0.003275	0.999968
3	9.988963e-01	0.017502	0.999119
4	9.998775e-01	0.000278	0.997930

CONCLUSION

- ## Key Findings and Conclusions of the Study

  We can see that by using CNN, Image augmentation and multiple layers we can develop a good image classification model

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

  Here the quality of the data plays a very major role in order to make a good model. We must see to it that all the images have proper resolution, scaling and all the images should be of same size in order to make a model that predict accurately. Data pre-processing on the input/image data must be done properly.

Limitations

Viewpoint Variation: In a real world, the entities within the image are aligned in different directions and when such images are fed to the system, the system predicts inaccurate values. In short, the system fails to understand that changing the alignment of the image (left, right, bottom, top) will not make it different and that is why it creates challenges in image recognition.

Scale Variation: Variations in size affect the classification of the object. The closer you view the object the bigger it looks in size and vice-versa

Limitations

Deformation: Objects do not change even if they are deformed. The system learns from the perfect image and forms a perception that a particular object can be in specific shape only. We know that in the real world, shape changes and as a result, there are inaccuracies when the system encounters a deformed image of an object.

Inter-class Variation: Certain object varies within the class. They can be of different shape, size, but still represents the same class. For example, buttons, chairs, bottles, bags come in different sizes and appearances.

Thank You