MPI-CBG Puzzle

Prathvik G S

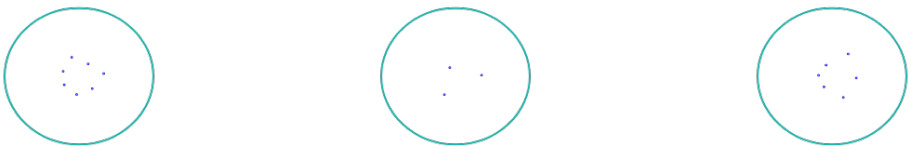
December 2022

1 Introduction

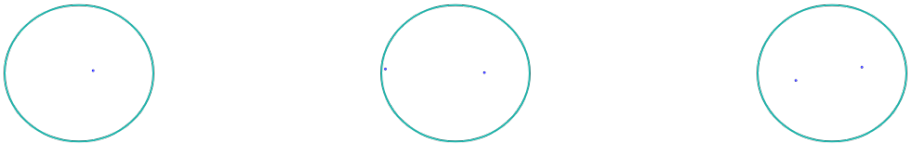
We have a dataset of images with two classes - Type 1 and Type 2. Using these images, we have to build a ML model for classifying new unseen images containing points into either Type 1 or Type 2.

Description of the given dataset
The given dataset has images containing points of two types- Type 1 and Type 2 with a count of 1000 each. The test set contains 20 images that are to be classified. Below are six randomly selected images from type 1 and type 2.

Sample images of type 1



Sample images of type 2



Intuitive deduction by observation of the images manually

We can observe that the number of points in an image of Type 1 is much greater than 2, while in Type 2, the no. of points are typically < 3, but this is just a manual observation and could be wrong. Most of the images in the test folder contain greater than 2 points, so if this is true, then most of the images in the test folder should belong to Type 1.

2 Neural Network model for this Image classification

We will try to build a Neural network model capable of classifying new images into Type 1 and Type 2. We will make use of Convolutional Neural Nets for this purpose.

Image preprocessing

The first step is to convert the image into numbers and make them ready to be trained data. An image is stored as a three-dimensional array. Note that we can grayscale the image (reducing RBG) to reduce the size of data by threefold, only if the accuracy of the output isn't affected, or in other words, we can grayscale the images if we don't lose any data from that. After this, we can go on building our model. First, we will use 60% of the data to train and the remaining to test. Once we get a good accuracy, we can divide it into 80:20 and train a new model. Finally, we can use the whole data to train the model and then use the model to classify new images whose class is unknown.

Model 1

Training the model using 60% of the data and using the remaining 40% of the data to test the accuracy and behavior of the model.
Training set- 60% which corresponds to 1200 images
The test set- 40% which corresponds to 800 images
Accuracy on the Training set-99.6%
Accuracy on the Test set-99.5%

```
Confusion Matrix
[[379   1]
 [  3 417]]

Accuracy on the test set
99.5%
```

So the model is neither underfitted nor overfitted as training and test set accuracies are 99.67 and 99.5%, respectively. We can now go ahead and build a model with 80% training data.

Model 2

Building a model using 80% of the data to train and using the remaining 20% of the data to test the accuracy and behavior of our model.
Training set- 80%, which corresponds to 1600 images
The test set- 20%, which corresponds to 400 images
Accuracy on the Training set-99.9%
Accuracy on the Test set-99.5%

```
Confusion Matrix
[[201   0]
 [  2 197]]

Accuracy on the test set
99.5%
```

So the model is neither underfitted nor is overfitted as the training and test accuracies are more than 99.9 and 99.5% respectively. We can now go ahead and build a model with all the training data and use it to classify the unknowns.

Final Neural Net model

Training set- 100%, which corresponds to 2000 images
Accuracy on the Training set-99.6%
Accuracy on the Test set-99.5%

```
Confusion Matrix
[[379   1]
 [  3 417]]
Accuracy on test set
99.5%
```

The summary of the Neural Network trained

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 398, 398, 32)      896

 max_pooling2d_6 (MaxPooling  (None, 199, 199, 32)     0
 2D)

 conv2d_7 (Conv2D)           (None, 197, 197, 32)      9248

 max_pooling2d_7 (MaxPooling  (None, 98, 98, 32)       0
 2D)

 flatten_3 (Flatten)         (None, 307328)            0

 dense_6 (Dense)             (None, 128)               39338112

 dense_7 (Dense)             (None, 1)                 129

=================================================================
Total params: 39,348,385
Trainable params: 39,348,385
Non-trainable params: 0
_________________________________________________________________
no. of layers
7
```

Using the model built to classify images in the test folder

When I used the model to classify the images in the test folder, I got the following:
Number of Type 1 images-17
Number of Type 2 images-3

```
0 represents Type 1
1 represents Type 2

Type1-18, Type2-2
[[0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]]
```

Surprisingly this is inline with the initial observation and prediction at the beginning.
So we trained a Neural Network model capable of classifying a given image into either Type 1 or Type 2 with high accuracy. The Neural Network has 7 layers. The final model, which was trained using all the data had an accuracy of 99.9% on the training data. We used this model to classify new images. For details and code click here.