Classification of Street View House Numbers Images Using CNN

Tushar Joshi
D17129761
DT228A, Full Time
Data Analytics
d17129761@mydit.ie

***Abstract*—This report describes an experiment carried out to perform image classification using Convolutional Neural Network on images in Street View House Numbers dataset. Street View House Number dataset is a dataset that contains real-world images obtained from Google Street View images. The trained model is optimised using AdaDelta optimiser and evaluated using average f1 scores for each class in test data.**

Keywords—image classification, Convolutional Neural Network, AdaDelta

I. Model design

A convolutional neural network model was built to classify images in the Street View House Numbers dataset and identify the house number in the given image. A convolutional neural network has one input layer followed by one or more convolutional layers, pooling layers and fully connected layers. In this experiment, there are two convolutional layers along with two pooling layers and one fully connected layer. Input image is converted into a matrix representation and the features are normalised to get all values in same range. These values are then fed as inputs to the first convolutional layer.

The class labels in the training and test data sets are converted to categorical variables using one hot encoding technique. This converts all class labels into a two dimensional array. Furthermore, class 0 has label 10, which is converted to label 0. Input image dimensions are 32x32, and they are fed to the network in the form of matrices. The values in these matrices are normalised to keep all input values between 0 and 1.

The convolutional layer scans each image in the training set and each unit in a convolutional layer learns small local features in the given input image. In this case, each unit in first convolutional layer scans a local feature of size 3x3 and outputs a feature map. Activation function used in this layer is Rectified Linear Units (ReLU) function. ReLU simply outputs linear combination of inputs and weights if the outcome is positive, otherwise it outputs zero for all negative values. Using this function makes the algorithm easier to train, as it acts like a linear function and is less likely to saturate when processing inputs. The feature maps generated by the convolutional layer are passed through the first pooling layer. The pooling layer focuses on important features and hence, reduces the number of connections in the network. In this implementation, max pooling is used, which takes the maximum value across a group of units as the activation value for the new aggregate unit. After this layer, the size of feature maps is reduced. This layer is followed by another convolutional layer and a pooling layer to further reduce the size of feature maps and emphasise on learning only the key features in the input image for classification.

The model is trained with a batch size of 128 inputs. This means the model parameters are updated every time when a batch of 128 input images is fed to the network based on the error value calculated using categorical cross entropy function. The training dataset is passed through the network five times, to reduce error value and help the model learn the underlying patterns and characteristics within the data.

The final layer in the network is a fully connected layer which is a standard non-convolutional neural network layer. This layer combines all the local features detected by convolutional layers and identifies global features in the given image. The activation function used in this layer is Softmax activation function, which gives a vector of probability distributions for all possible classes for a given input image. In this case, number of possible outcomes or class labels is ten, with numbers from zero to nine each representing a possible class label. All probabilities add up to one, and the class with maximum probability is the class label for the input image. Figure 1 shows architecture of the proposed network.

```
Train on 543949 samples, validate on 60439 samples
Epoch 1/5
543949/543949 [==============================] - 1602s 3ms/step - loss: 0.3929 - acc: 0.8907 - val_loss: 0.2122 - val
_acc: 0.9442
Epoch 2/5
543949/543949 [==============================] - 1526s 3ms/step - loss: 0.2278 - acc: 0.9390 - val_loss: 0.1752 - val
_acc: 0.9518
Epoch 3/5
543949/543949 [==============================] - 1599s 3ms/step - loss: 0.1927 - acc: 0.9481 - val_loss: 0.1511 - val
_acc: 0.9590
Epoch 4/5
543949/543949 [==============================] - 1657s 3ms/step - loss: 0.1725 - acc: 0.9534 - val_loss: 0.1442 - val
_acc: 0.9611
Epoch 5/5
543949/543949 [==============================] - 1637s 3ms/step - loss: 0.1590 - acc: 0.9572 - val_loss: 0.1362 - val
_acc: 0.9629
Test score: 0.3189770015782019
Test accuracy: 0.9166026429010449
26032/26032 [==============================] - 17s 666us/step
Average F1 Score:  0.9166026429010449
```

Figure 1: Model Architecture

The loss or error estimate in classification is calculated using categorical cross entropy function. This function will compare distribution of predictions in the output layer with the true class distribution. Based on this comparison, the loss is calculated in each training epoch. This loss is minimised through optimisation of network parameters which is performed using AdaGrad optimiser. This improves the classification performance of the model.

II. Results

The convolutional neural network model is evaluated using the test data which contains images with similar dimensions as those in the training set. Initially, the evaluation is performed using classification accuracy on test data. The network achieved classification accuracy of 90% when evaluated using test data.

Evaluating a model using accuracy is not always enough, as it can be misleading, if the class distribution in training data is not balanced. Classification accuracy alone cannot be used to determine the performance of a model. Hence, F1 score metric was also used to evaluate the model. Evaluation results using this technique showed that the model achieved a F1 score of 0.92 when used to classify unseen images. This is average F1 score for all classes (0-9) in test data.

Figure 2 shows details of results achieved after running the model on test data.

```
Predicted Number: 5
```

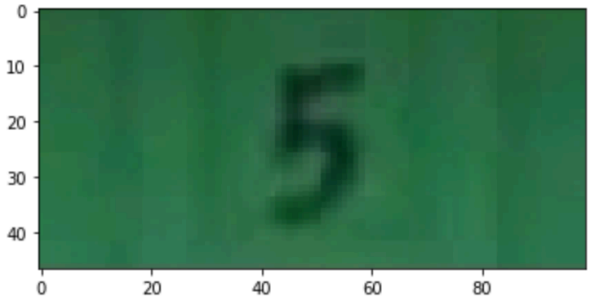

Figure 2: Prediction Using Model

III. CONCLUSION

This experiment was performed in order to understand the structure and functioning of Convolutional Neural Network for image classification. Street View House Numbers dataset was used as a starting point to understand how an image is processed before feeding it into the network for classification, how the network identifies various features present in the given image, and how those features are combined to identify the number present in the image. It also helped to understand the differences between architectures of a standard neural network and a convolutional neural network. Instead of having a standard fully connected hidden layer between input and output layers, convolutional neural network has convolutional layers, which has units that learn specific features of an image independently. These units are essentially clones of each other with same weights.

The ReLU activation function used in the convolutional layers helps to deal with vanishing gradient problem, which is a common problem that occurs when training a deep network. In such networks, if error value diminishes by the time it reaches the input layer as it is propagated back through the network. ReLU activation function deals with this issue by forcing all the negative values to zero.

In order to improve the existing implementation, the network was built using different optimisation techniques such as Adam Optimiser and AdaDelta Optimiser. However, this did not improve the performance of the classifier.

The experiment was performed on a CPU with 8 GB RAM, Intel I5 Processor and 1.8 GHz clock speed. As the processing capacity on a single CPU is limited, the classification was performed using a smaller dataset that had cropped images of the original dataset. The performance of the classifier can be improved if the model is trained for more epochs using a better CPU or using a cluster of GPUs depending on the availability of resources. Also, adding more convolutional layers for detecting image features can help to improve the performance of the classifier.

IV. REFERENCES

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng Reading Digits in Natural Images with Unsupervised Feature Learning *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*

https://www.kaggle.com/oricou/svhn-avec-cnn

http://ufldl.stanford.edu/housenumbers/