Computational Machine Learning – Assignment 2

Classification of Road Traffic Signs

Group

Arjun Padmanabha Pillai (s3887231)

Maleshwar Sastri Saraswathula (s3857746)

Background

This project is aimed at classifying images of European road traffic signs. We made use of a modified version of Belgium Traffic Sign Classification Benchmark. The images are first divided by their shapes and are then sub divided based on their type.

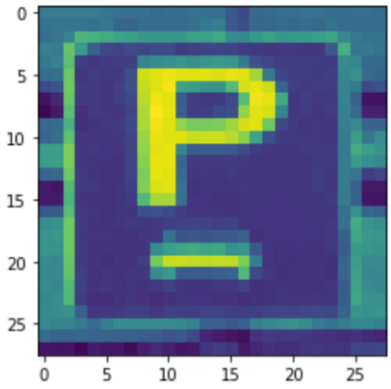

Figure 1. Parking Sign from the dataset

Traffic Sign Classification based on shape

Our dataset consists of signs coming under 5 different kinds of shapes. Namely diamond, hex, round, square and triangle. We first start off by plotting the distribution of data across the different kinds of shape to develop an understanding and set an evaluation framework.

Our pre-processed dataset of images comes in grayscale with dimensions 28x28. We model this dataset with 2 layers of Conv2D the convolutional layer applies filters to the input so that the features and their locations are detected. keeping input shape as 28x28. We use relu activation because when compared to sigmoid it is simple, fast and converges much faster.

We use softmax as activation function in the output layer, and this classification is done based on the weights from the previous layers, The final dense layer will give 5 outputs based on weights assigned in the previous layers. Softmax is generally used for classification.

A loss function evaluates how well the algorithm has modelled the dataset. Ideally we aim for a low loss function. To compile our model we use categorical crossentropy as loss function since we are dealing with a multiclass classification model with more than two output labels.

We use RMSprop as optimizer which automatically adjusts learning rate, its uses adaptive learning rather than treating the learning rate as a hyperparameter.

While fitting the model we took the number of epochs as 50, and steps per epoch was calculated use training dataset size by batch size which came to 116, the above steps resulted in producing a model with 0.9992 accuracy which was confirmed with the function model.evaluate().

To test our model, we created a test dataset out of images found from various internet sources, this is stored and converted to a csv which is read and passed through an iterator to generate predictions. The generated predictions are written into a csv file which contains the correct classifications. We can compare our predictions with the correct values to estimate the performance of our model.

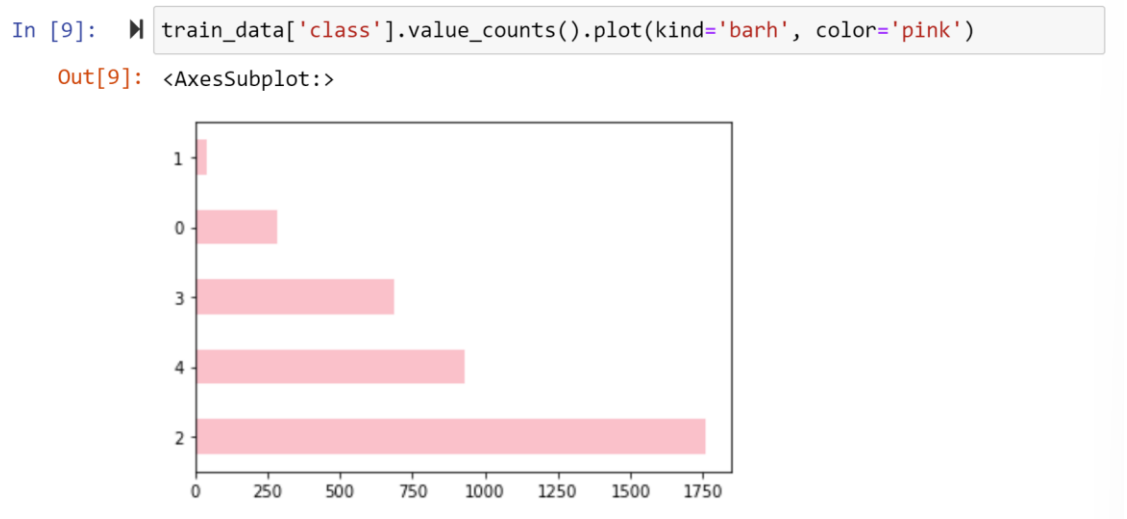

Figure 2. Distribution of data based on class

Traffic Sign Classification based on type

The dataset under inspection came with 16 different types of traffic signs. We create a csv of the dataset which contains the path and labels. This csv will be used for further processing and for modelling. No changes were made to the dataset the contents are all of dimension 28x28 and grayscale.

While modelling the dataset we make use of 4 convolutional layers. Increasing the depth increases the capacity of the model, more layers make the model more efficient. In this scenario we are presented with very few samples to train our model which will adversely affect predictions further down the line.

The model is compiled with categorical crossentropy and RMSprop as optimizer as with the previous shape based classification. We fit the model with 231 steps per epoch over 50 epochs. The accuracy comes out to be 1.00 as confirmed by the model.evaluate function.

We observed when it came to making predictions the model did not perform as well as it should. Making correct predictions only 4 out of 10 times. We attributed this to the lacking of training data. While some types of road signs had a surplus of training data other types of road signs had very few to go on. This severely affected the model's performance.

```
1/1 [==============================] - 0s 46ms/step
1
prediction:
```

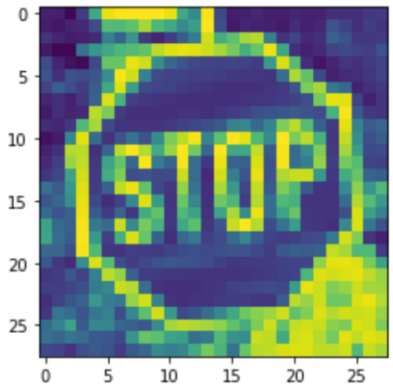

Figure 3.1 Model2 makes a correct prediction.

The model performed well when subjected to images that had an appropriate number to train but performed poorly when it came to other signs which lacked in training data,

```
1/1 [==============================] - 0s 21ms/step
8
prediction:
```

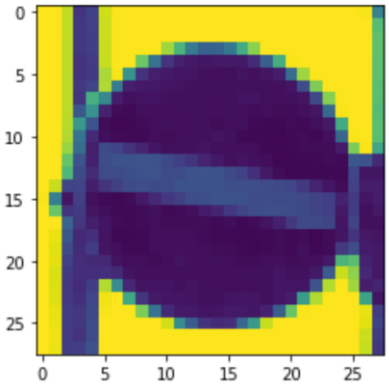

Figure 3.2 Model2 makes an incorrect prediction.

As depicted above the class labels 1 and 8 correspond to Stop and traffic directive but the model was only able to classify one of the images correctly.

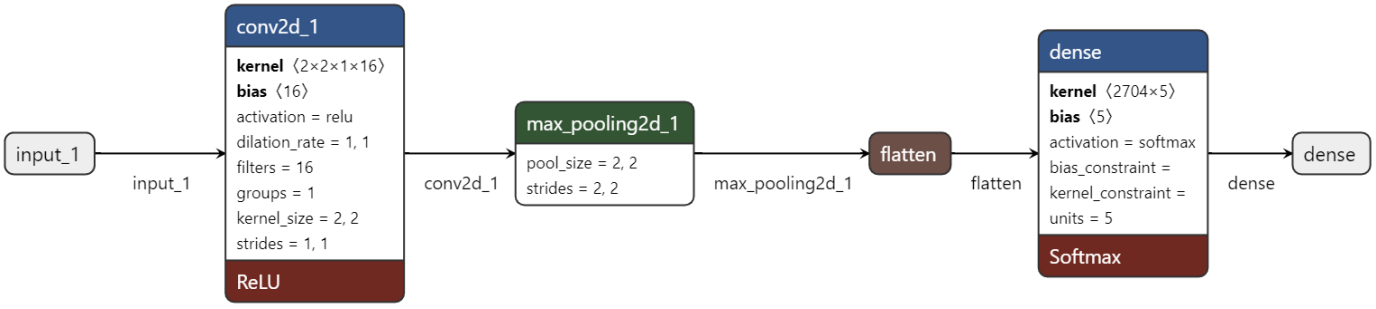

Figure 4.1 Model to classify shape

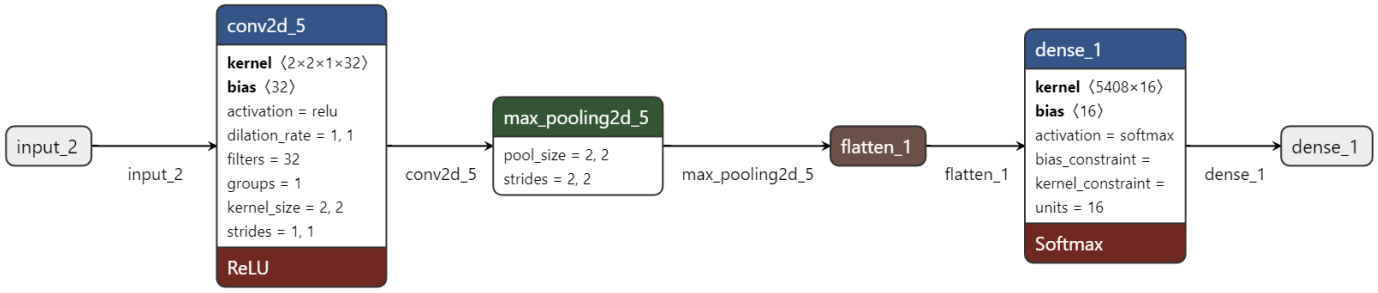

Figure 4.2 Model to classify based on type

Ultimate Judgement

- We can attribute the performance of the first model to having less labels to classify hence resulting in comparatively more data per class.
- While we saw exceptional performance in the first model which classified traffic signs based on shape the second model however did not perform as well, despite the addition of multiple convolutional layers.
- Data Augmentation is a solution to the above problem, rotation, focus, zoom, changing angles and orientation will significantly increase the data we have for training, this in turn improves performance and variation in model. This further reduces overfitting.

References

[1] Tensorflow documentation. Viewed on 30th May, 2022.
https://www.tensorflow.org/tutorials/images/classification