Capstone Project Report

Traffic Sign Detection

1. Project Overview

Self-driving cars has become a buzzword in many areas of the world right now. These cars are reshaping the automobile industry, which formerly relied solely on drivers to monitor their vehicles. In today's world, as the number of vehicles increases, so do the number of road accidents, and according to reports, India ranks first in the world for the highest number of accidents per capita. This is caused by a variety of factors, including poor law enforcement, carelessness, and so on. One reason for this is that people do not recognise or follow traffic sign boards. As a result, I will be using a deep learning approach and build a model that detects traffic sign ahead and direct the driver to follow it, which further has the potential to reduce traffic accidents.

2. Problem Statement:

Nowadays, there is a lot of attention being given to the ability of the car to drive itself. One of the important aspects for a self driving car is the ability for it to detect traffic signs in order to provide safety and security for the people not only inside the car but also outside of it. There are many various types of traffic signs, such as speed limits, traffic lights, designating directions (left or right), and so on. The task may be stated simply as follows: categorize the types of traffic signs. The main objective is to design and construct a computer based system which can automatically detect the road signs so as to provide assistance to the user or the machine so that they can take appropriate actions. The proposed approach consists of building a model using convolutional neural networks by extracting traffic signs from an image using color information. I will be using convolutional neural networks (CNN) to classify the traffic signs.

3. Metrics

Some criteria must be used to evaluate the model's quality and predictability of the response. I have used accuracy will as a metric. The accuracy will be assessed as follows after training.

$$acc(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i)$$

4. Data Exploration

The GTSRB-German traffic sign detection benchmark dataset was utilized in this study. This dataset is available on platforms such as Kaggle. The dataset contains over 50,000 images of various traffic signs. It is further subdivided into 43 distinct classes. The dataset is approximately 300 MB in size. The dataset is divided into two folders: a train folder that contains images for each class, and a test folder that will be used to test our model.

Visualization of Data

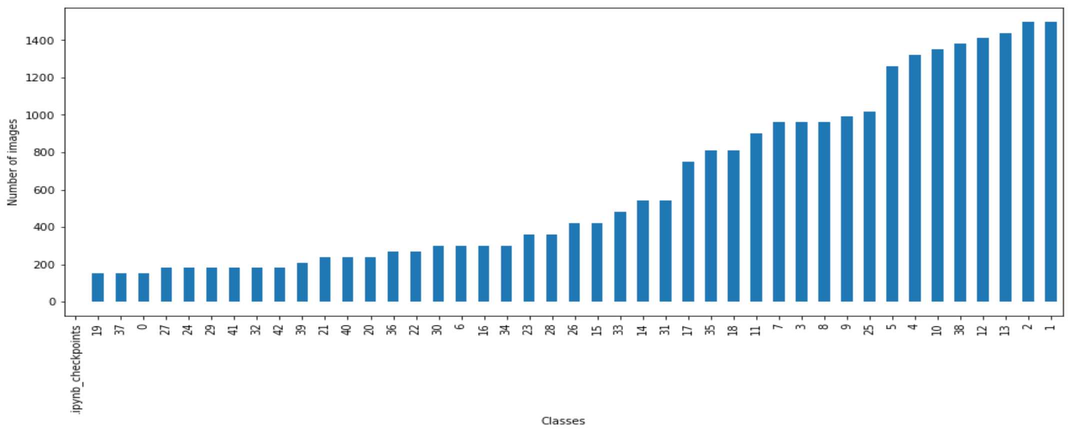


Fig: Train Data with Corresponding Classes in grid format

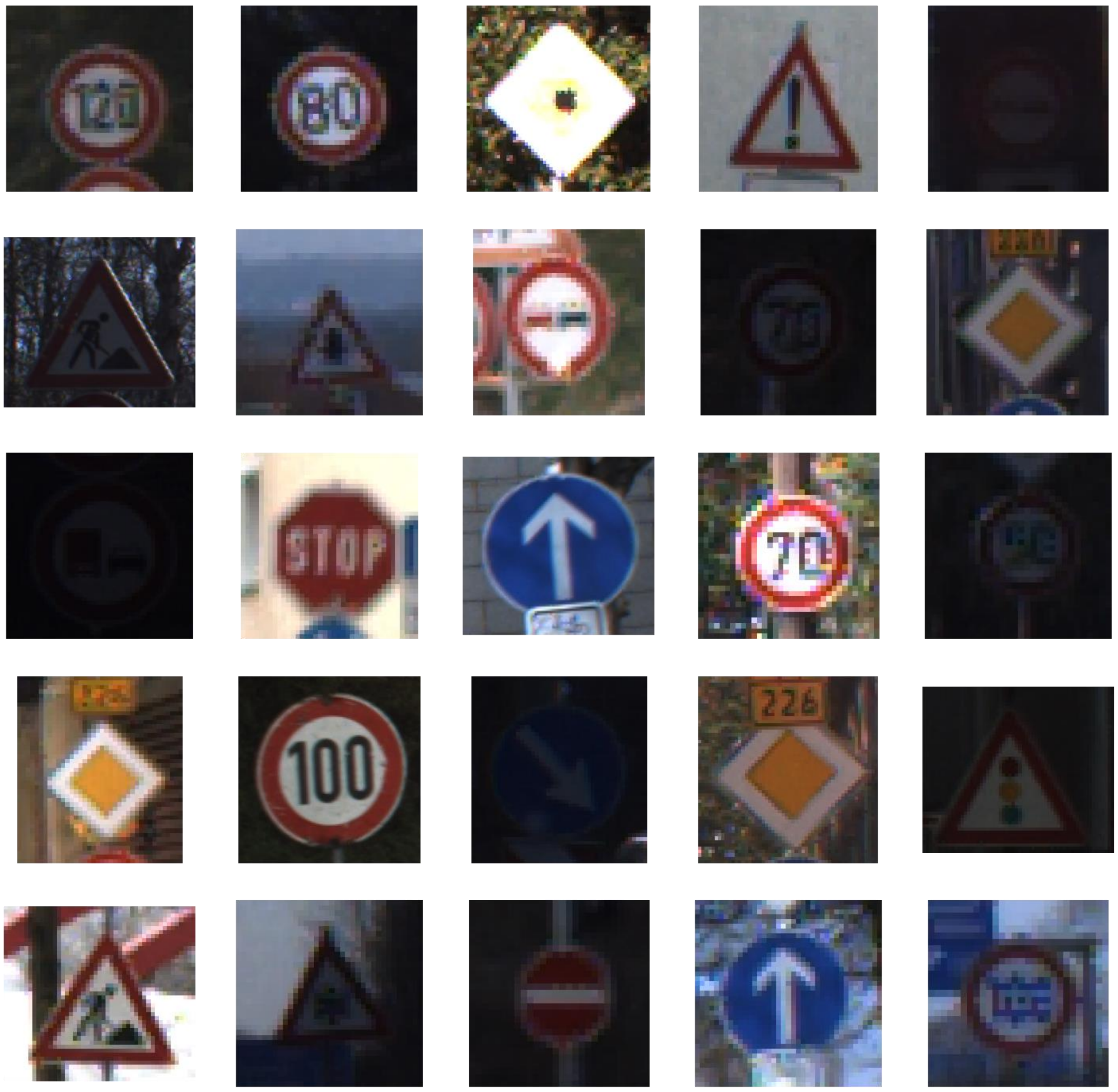


Fig : 26 Test Images

The dataset is pre-processed and Images are resized to 30X30

5. Algorithm used

A Convolution Neural Network (CNN) was used in this study, which is a cutting-edge architecture for solving problems like pattern recognition in images. CNN is a type of Neural Network that is commonly used to solve visual imaging problems. CNN excels at this task in part due to its spatial invariance, which enables it to detect patterns that have been translated, moved, or distorted. This spatial invariant ability, which enables CNN to perform so well for images, is due to the convolution layer's ability to recognise high-level image features such as edges, curves, straight edges, and colour. To accomplish this, the convolution layer employs filters (also known as convolution kernels) that slide across the entire image from top to bottom, delivering the input to the activation layer and resulting in a 2-dimensional activation map. The model developed for this project uses CNN.

6. Bench Mark

Many different techniques have been applied to detect traffic signs. Most of these techniques are based on using HOG and SIFT features. Several methods for recognising traffic signs have been published.

Moutarde et al. present a neural network-based system for recognising speed limit signs in Europe and the United States [1]. They do not, however, provide individual classification results. On average, the overall system, including detection and tracking, achieves a performance of 89 percent for US speed limits and 90 percent for European speed limits.

A number-based speed limit classifier is trained on 2,880 images in [2]. It has a 92.4 percent correct classification rate on 1,233 images It is unclear, however, whether images of the same traffic sign instance is shared by multiple sets.

In my approach I used biologically inspired convolutional neural networks to build a model which can predict the type of traffic sign with better accuracy than already existing models and Feature Extraction is worth to be noted as well.

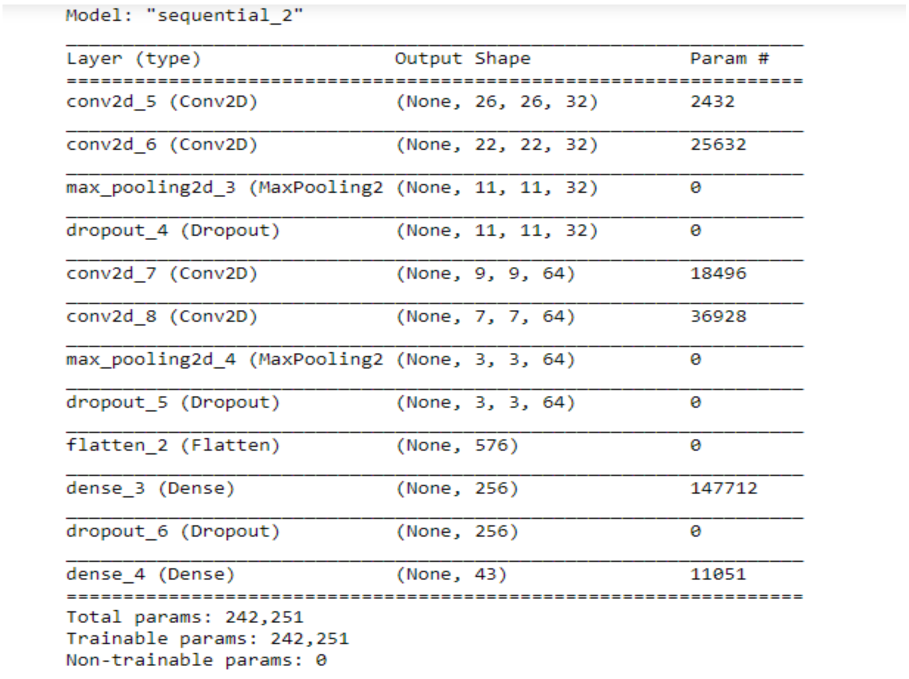
7. Methodology

Data Preprocessing

I have used 43 different types of traffic signs. Initially, open the directory containing the dataset and resized each image. Each image is added to the data list, and the associated classes are added to the labels list. We now make numpy arrays out of these lists. Each image is added to the data list, and the associated classes are added to the labels list. The dataset was then divided into two parts: 80 percent for training and 20 percent for testing. The class labels for the train and test datasets are now converted into a single hot encoding into a 43x43 vector representation using to categorical().

Implementation

The model I created is a sequential model. Conv2D generates a 2D convolutional kernal with a kernel size of (5,5) (the dimensions of the kernal for processing the image) and an activation function of Rectified Linear Unit which is a max function which sets all negative values to zero and other values constant as arguments.. Maxpool2D reduces image dimensions, and the pool size I have used is (2,2), which corresponds to a 2x2 window. Dropout is used to ignore some neurons (data points) from the dataset at random while fitting the model. It was used to deal with model overfitting. To convert a nD array to a 1D array, I used Flatten(). To create a multiclass classifier, the activation function softmax is used. The model is compiled using categorical crossentropy as the loss function. The categorical crossentropy function is used to reduce the distance between the predicted and actual class label. Adam, the optimizer, is used to determine the individual learning rate for each parameter. An epoch is the number of iterations required to fit the model. In a batch size of 32 images, the model is fitted for the trained dataset.



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_5 (Conv2D)            (None, 26, 26, 32)        2432      
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 22, 22, 32)        25632     
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 11, 11, 32)        0         
_________________________________________________________________
dropout_4 (Dropout)          (None, 11, 11, 32)        0         
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 9, 9, 64)          18496     
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 7, 7, 64)          36928     
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 3, 3, 64)          0         
_________________________________________________________________
dropout_5 (Dropout)          (None, 3, 3, 64)          0         
_________________________________________________________________
flatten_2 (Flatten)          (None, 576)               0         
_________________________________________________________________
dense_3 (Dense)              (None, 256)               147712    
_________________________________________________________________
dropout_6 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_4 (Dense)              (None, 43)                11051     
=================================================================
Total params: 242,251
Trainable params: 242,251
Non-trainable params: 0
```

Fig: Summary of Model

8. Evaluation and Results

I have used matplotlib to plot graphs between epochs and accuracy for training and validation accuracy, as well as between epochs and loss for training and validation loss. Because I did not use dropout for the validation set, the validation accuracy is higher than the training accuracy.

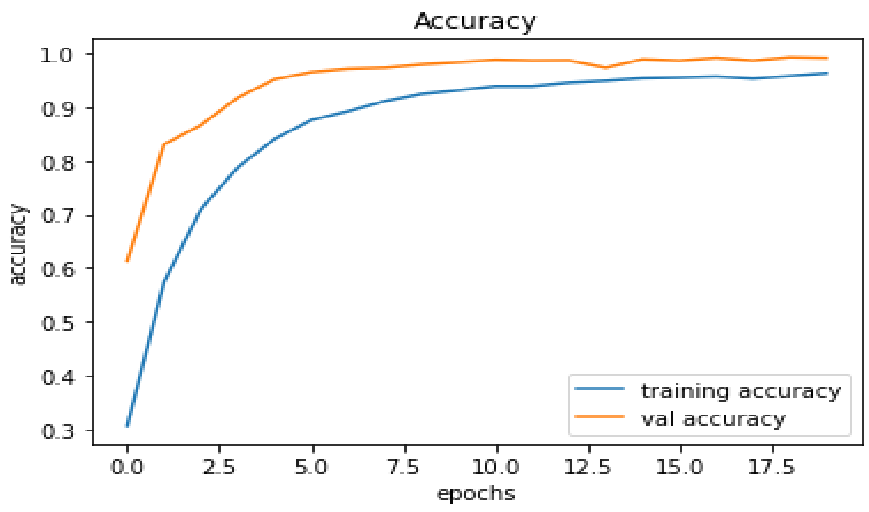


Fig : Accuracy Plot

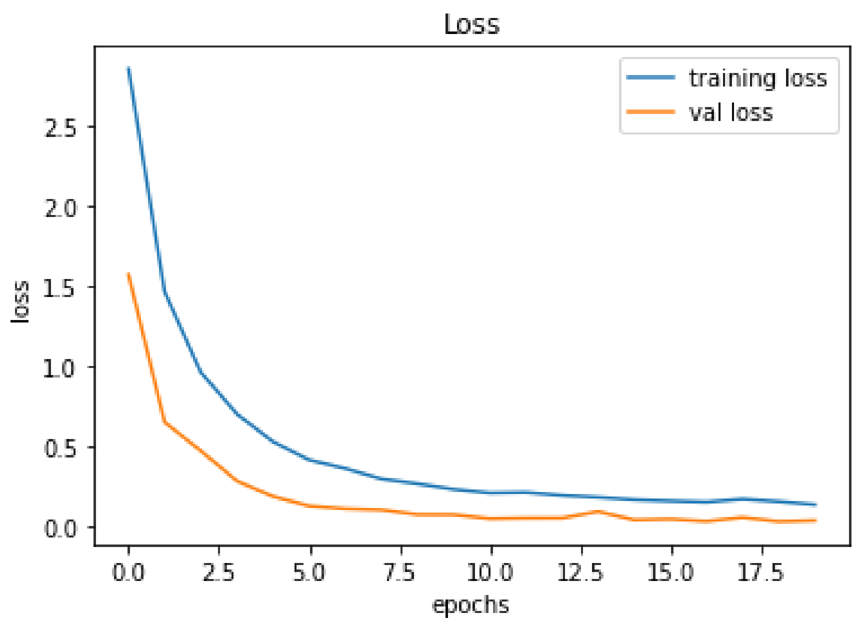


Fig: Loss Plot

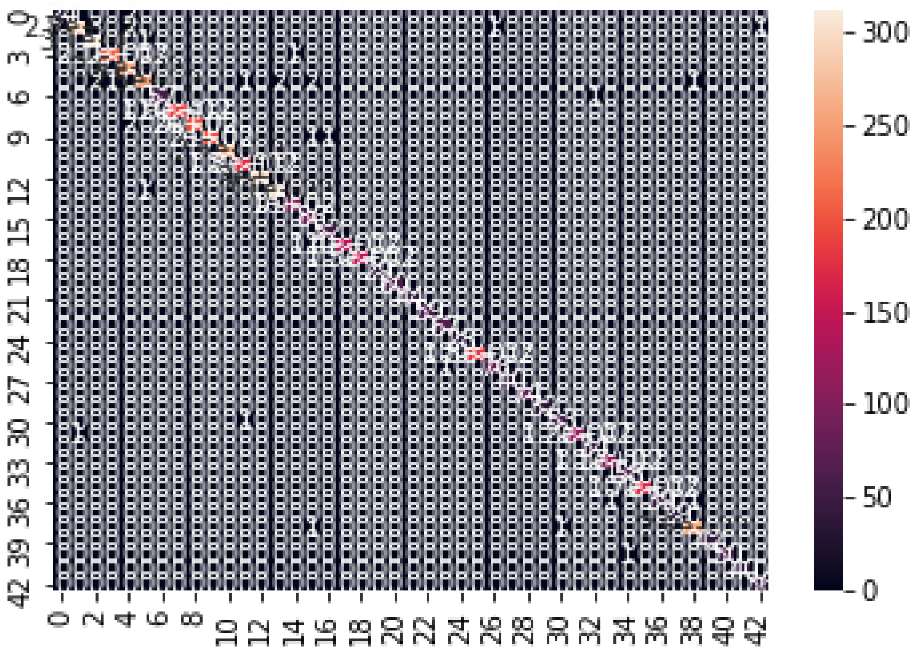


Fig: Confusion Matrix

The model outperformed other benchmark models by giving us accuracy close to 98 percent, which was higher than all other benchmark models.

9. Conclusion:

The proposed model has shown better accuracy in detecting various Traffic signs. The most difficult aspect was developing the models and conducting adequate training. This was initially due to the large amount of resources required, as well as the fact that the model was frequently too sophisticated or too basic to detect patterns in data. Training, overfit regulation, and convergence were all difficult at first, and it is a never-ending process. However, once a model begins to converge, increasing accuracy and decreasing loss, it is simply a matter of time and fine tuning to improve it. Of course, we can always make changes to the model. we'll have to figure out when it's good enough for our needs because there isn't a clear end point. Deep learning approach significantly reduces the time cost of training negative samples, ensures the balance of positive and negative samples, and improves the accuracy of the Softmax classifier.

10. Future Work

There are a few things that might be done to improve the project. One possible improvement is to improve model further by using Data Augmentation and image prepossessing to improve model result quality. Another improvement is to learn how to identify the text portion of traffic signs and use that information to build a real-time traffic sign system. Another goal for the future is to be able to apply this to self-driving cars.

11. References

[1] A. Broggi, P. Cerri, P. Medici, P. P. Porta, and G. Ghisio, “Real time road signs recognition,” in Proceedings of the IEEE Intelligent Vehicles Symposium, 2007, pp. 981–986.
[4] C. G. Keller, C. Sprunk, C. Bahlmann, J. Giebel, and G. Baratoff, “Real-time recognition of U.S. speed signs,” in Proceedings of the IEEE Intelligent Vehicles Symposium, 2008, pp. 518–523.

[2] A. S. Muhammad, N. Lavesson, P. Davidsson, and M. Nilsson, “Analysis of speed sign classification algorithms using shape based segmentation of binary images,” in Proceedings of the International Conference on Computer Analysis of Images and Patterns, 2009, pp. 1220–1227

Datset link : https://benchmark.ini.rub.de/gtsrb_news.html