iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VIII **Month of publication:** August 2021

DOI: https://doi.org/10.22214/ijraset.2021.37700

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

Traffic Sign Classification Using CNN

Kolachalama Venkata Naga Sreya
Student, Department of Electronics & Communication, Sastra Deemed to Be University, Thanjavur.

Abstract: ***With the increasing necessity of autonomous electric vehicles as time passes by, there are a lot of technical prospects within the structure that have a lot of scope for advancement. One such prospect is traffic sign recognition. Several models have already been developed and are in practice but it is evident to everyone within this field that there is still a lot of untapped potential. In this project we implement feature classification using convolutional neural networks to achieve an efficiency and accuracy higher than that of a conventional model. The system first converts the image into grayscale and then three layers of the image are created. By using skilled convolutional neural network which incorporates crucial data of traffic signs and images, they are parallelly assigned to corresponding classes. Results have shown that this system works with great efficiency.***
Index Terms: Traffic sign, GTSRB, ALVINN, Neural Network, CNN, LeNet, ReLU, Evaluation, Epoch

I. INTRODUCTION

It is an established fact that traffic sign classification is one of the most important aspects to the system of autonomous vehicles as they help prevent accidents and regulate the traffic on roads. As soon as driver-less vehicles came into play a couple of decades ago, traffic sign classification was one of the burning topics as a lot of research was spent into developing an ideal model. Initially, traditional computer vision and machine based methods were used to implement the recognition model but seeing as they did not have enough accuracy and efficiency, they were soon replaced by deep learning based classifiers. ALVINN (Autonomous Land Vehicle in a Neural Network) was the first project to use neural networks for autonomous vehicles navigation. Compared to the previous models which used hundreds of neural network layers, it consists of shallow and fully connected layers. But considering the limitations of that model, new developments have been made which used modern convolutional networks to extract features from the camera frames. These new developments have helped integrate real world scenarios like lane keeping, obstacle detection etc. In recent years, several new developments have been made in every aspect concerning autonomous vehicles and traffic sign classification is no different. Models which used modern convolutional networks incorporated three main stages into the classification. They are :

A. Region Segmentation to obtain regions containing the signs.
B. Shape analysis to classify signs depending on their shapes.
C. Recognition, where signs spotted previously are identified.

We have various classification techniques to implement such as KNN, ANN, SVM (Support Vector Machine), RF (Random Forest) etc. and while some of them have been proved to be very good, they are not cost-effective and have a relatively high processing time. Hence, in this project, we will try to implement traffic sign classification using CNN focusing on the current drawbacks in the existing model such as cost, execution time, accuracy and effectiveness.

II. METHODOLOGY

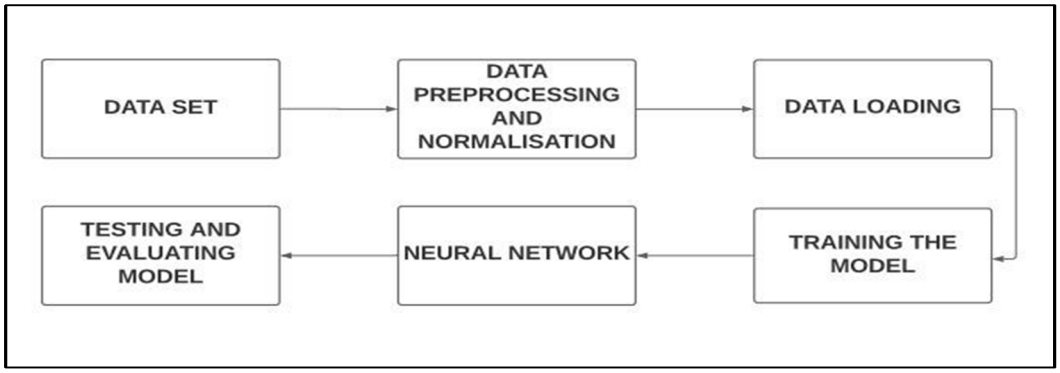
A CNN based model for traffic sign classification is proposed to increase the accuracy of the classification model. A popular dataset German traffic signs detection dataset (GTSRB) dataset is used to implement our model. The consists of 39209 images with 43 different classes.
The GTSRB consists of 39209 images with 43 different classes. The images are distributed unevenly between those classes and hence the model may predict some classes more accurately than other classes.
Initially, Data pre-processing was done, which means Gray Scale image is obtained from an RGB image, Histogram Equalization, and Normalization on the GTSRB dataset. Data pre-processing is followed by Data augmentation. Data augmentation increased the number of training data to prevent overfitting. After that the TS CNN model was implemented and trained with the GTSRB dataset.

III. BLOCK DIAGRAM

Figure 1 shows us the flow of processes followed in our model. The GTSRB dataset is shuffled first in the pre-processing, as the dataset is in ordered way and if not shuffled the algorithm might learn the order of images. After the pre-processing step the data is loaded and the neural networks algorithm is implemented.

1) *Data Pre-processing:* CNN model has few limitations, they cannot be trained on images of different dimensions. Hence, it is compulsory to have images of same dimension in the dataset. Therefore, we have to compress/interpolate the images to a single dimension. Without compressing too much of the data and at the same time without stretching the image too much weneed to decide the dimension which is in between and keep theimage data mostly accurate. Thus, it was decided to have an image of dimension [32*32*1].
2) *Data Loading:* The dataset is loaded and now we need to divide it into training and testing set. And, in validation set. But we cannot divide the model directly because all models will notget trained, as all the traffic signs in the dataset is not randomized. So, first we will randomize the dataset. The dataset will be split into 60:20:20 ratio as training, validation, test dataset respectively
3) *Neural Networks Model:* You can see we've used the Dropout method. It drops some neurons from training and hence helps in preventing overfitting. Dropout(0.2) represents 20% of neuronsto be dropped randomly in every cycle. It is also a regularization method.

Figure 1 - Block diagram.



IV. ARCHITECTURE

Figure 2 depicts the algorithm followed in the model proposed. We first convert the image into a square shaped gray-scale image. Then we create layers of this image. There will be threelayers which will be created in the back-end namely : input layer, hidden layer(i.e pooling layer) and target layer. The pooling layer will acquire the main features and compare them to the target layer.

We used LeNet as my base model and started playing with it. This is the point where we experimented a lot and tried to tune the parameter in the network.

We made 4 convolutional layers and 2 max pooling layers which quite simple to understand. Playing with a number of convolutional filters made us learn more about CNN.

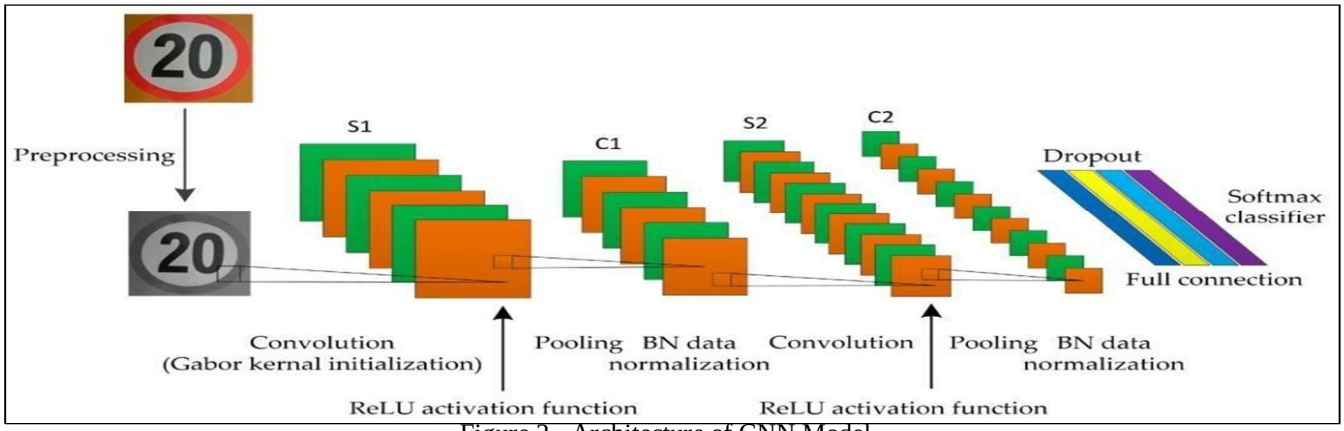


Figure 2 - Architecture of CNN Model

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

V. EVALUATION OF THE MODEL

Table 1 lists the training results of the model for a batch size of 50 images, and for 10 epochs.

Epoch number	Batch size	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
5/10	50	1.9137	44.76	1.4430	51.5730
6/10	50	1.7242	47.92	1.0869	54.6907
7/10	50	1.5093	54.36	0.8670	58.7685
8/10	50	1.3213	59.00	0.6307	65.8349
9/10	50	1.2526	62.40	0.6407	71.8311
10/10	50	1.0768	66.80	0.5168	76.8483

Table 1: Model evaluation for Batch size of 50 images

A batch with a size of 50 images has been tested using 10 epochs and the accuracy at the end was found to be around 67 per cent. Execution time for each epoch was around 18 seconds and the losses have steadily decreased as the number of epochs increased.

In *Table 2* we listed the training results for a batch size of 31367 images and 15 epochs.

Epoch Number	Batch size	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
01/15	31367	1.9311	49.37	0.7337	79.75
02/15	31367	0.9181	72.60	0.5157	84.38
03/15	31367	0.7225	77.75	0.3236	89.80
04/15	31367	0.6002	81.43	0.3021	90.40
05/15	31367	0.5507	83.12	0.2279	93.05
06/15	31367	0.4671	85.44	0.1826	94.04
07/15	31367	0.4396	86.69	0.1634	95.32
08/15	31367	0.4001	87.64	0.1394	95.93
09/15	31367	0.3820	88.36	0.1553	94.89
10/15	31367	0.3521	89.23	0.1362	96.05
11/15	31367	0.3468	89.84	0.1310	95.97
12/15	31367	0.3311	90.07	0.1107	96.62

As we can see the difference above, when we increased the batch size to 31367 from 50, we got around 90% accuracy. The loss decreased by around 5 times. As we can observe from ***Table 2*** the training loss is greater than the validation loss and hence we can say that our model is under-fitting. Also validation loss should be as low as possible. We achieved it by increasing the batch size and number of epochs.

In ***Figure 3*** and ***Figure 4*** we plotted the Training accuracy and Training loss plot for 15 epochs.

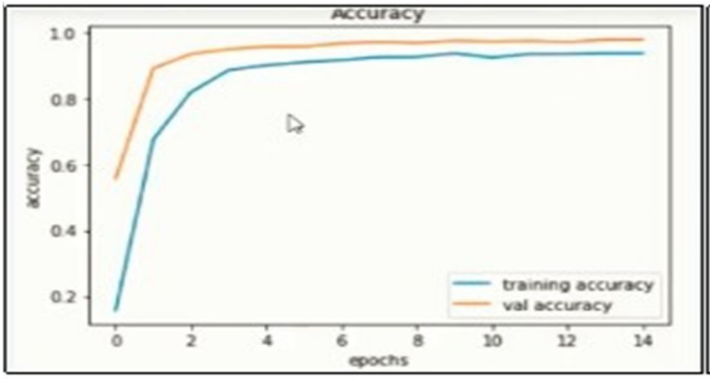


Figure 3: Training Accuracy plot for 15 epochs

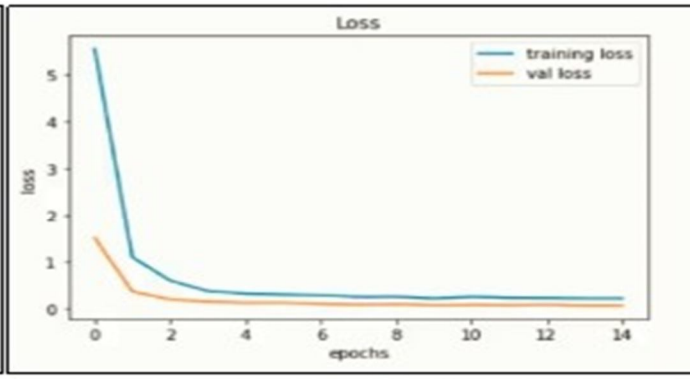


Figure4: Training Loss plot for 15 epochs

From ***Figure 3*** and ***Figure 4*** we can see that as the number of epochs increased the accuracy increased and losses decreases. But increasing the epochs could also increase the computational time which is not desirable. Therefore we fix the number of epochs between 10-12 to get the computation done in optimal time.

VI. RESULTS

To test the model results we took a random image from the internet and loaded it into the model as shown in ***Figure 5.***

```
In [72]: import requests
         from PIL import Image
         url = 'https://c8.alamy.com/comp/A0RX23/cars-and-automobiles-must-turn-left-ahead-sign-A0RX23.jpg'
         r = requests.get(url, stream=True)
         image = Image.open(r.raw)
         #image = Image.open(r"D:\download.png")
         plt.axis('off')
         plt.imshow(image, cmap=plt.get_cmap('gray'))

Out[72]: <matplotlib.image.AxesImage at 0x2615a7e3dc8>
```

Figure 5: Input for validation of result

A random traffic sign image is given as an input to the model and it is run. ***Figure 6*** shows the output of the model and the class number(34) to which the sign belongs.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

```
pred = int(prediction)
plt.imshow(image)
plt.axis('off')

for num, name in data.iteritems():
  name = name.values
  print("predicted sign: "+ str(name[pred]))
```

```
predicted sign: 34
predicted sign: Turn left ahead
```

Figure 6 : Result showing the class and sign

And the sign was predicted accurately. As we can see, in the output side the sign shows that it belongs to the class 34 whichmeans it belong to the class “turn left ahead”.

Traffic Sign Recognition using the CNN model was achieved using Keras Library. The model used 15 epoch, ie all training dataset passed through forward pass and back propagation through the neural network 15 times. Our model was tested using 31367 images in the GTSRB dataset and an accuracy of 90.07% has been achieved.

VII. CONCLUSION

A convolutional neural network model has been trained and implemented to classify traffic signs and the results demonstrate the fact that this technique yields a higher efficiency, accuracy, minimal losses and lesser execution time compared to the traditional model. The accuracy was found to be about 90% after training 10 epochs. A total of 43 classes were included in the training and the results were more than satisfactory. Our project mainly contributes to the development of novel algorithm of modern CNN based traffic sign classification.

REFERENCES

[1] Alexander Shustanova , Pavel Yakimova “CNN Design for Real-Time Traffic Sign Recognition” in 3rd International Conference “Information Technology and Nanotechnology”, ITNT-2017, 25-27 April 2017, Samara, Russia.

[2] Hengliang Luo; Yi Yang; Bei Tong; Fuchao Wu - Traffic Sign Recognition Using a Multi-Task Convolutional Neural Network - IEEE Transactions on Intelligent Transportation Systems (Volume: 19, Issue: 4, April 2018)

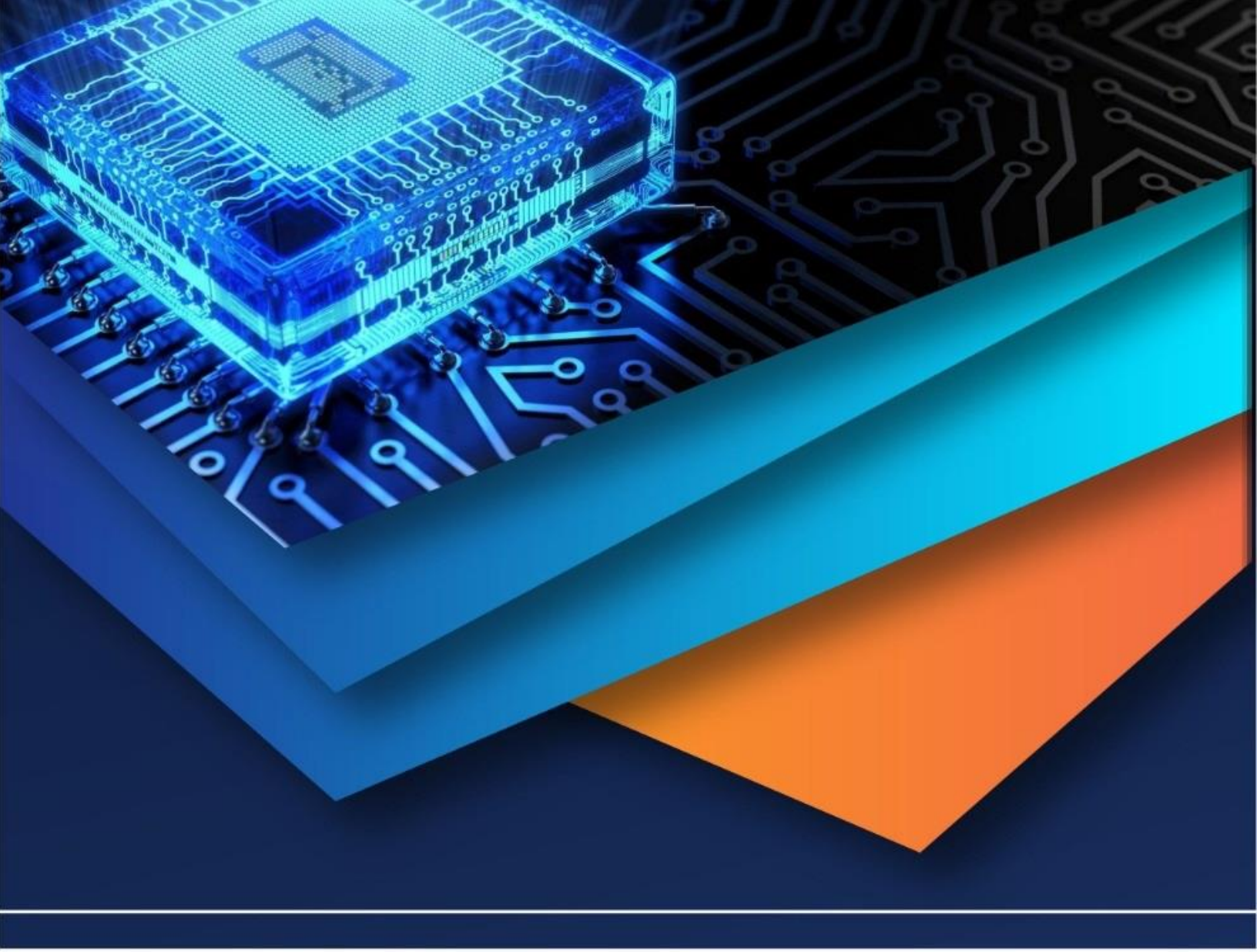
[3] Citlalli Gámez Serna , Yassine Ruichek - Traffic Signs Detection and Classification for European Urban Environments- IEEE Transactions on Intelligent Transportation Systems(Volume: 21, Issue: 10, Oct. 2020

[4] Satilmis, Yusuf & Tufan, Furkan & Şara, Muhammed & Karslı, Münir & Eken, Süleyman & Sayar, Ahmet. (2019). CNN Based Traffic Sign Recognition for Mini Autonomous Vehicles: Part II. 10.1007/978-3-319-99996-8_8.

[5] Vavilin, A., Jo, K.-H.: Graph-based approach for robust road guidance sign recognition fromdifferently exposed images. J. Univ. Comput. Sci. 15(4), 786–804 (2009)

[6] L. Shangzheng, "A Traffic Sign Image Recognition and Classification Approach Based on Convolutional Neural Network," 2019 11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), 2019, pp. 408-411, doi: 10.1109/ICMTMA.2019.00096.

[7] M. A. Vincent, V. K. R and S. P. Mathew, "Traffic Sign Classification Using Deep Neural Network," 2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS), 2020, pp. 13-17, doi: 10.1109/RAICS51191.2020.9332474.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY