iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VI **Month of publication:** June 2022

DOI: https://doi.org/10.22214/ijraset.2022.44047

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: 7.538
Volume 10 Issue VI June 2022- Available at www.ijraset.com

Cataloging of Road Traffic Signs (Building Deep learning model and Android Application)

G. Akhil Reddy[1], K. Nithesh Reddy[2], M. Abhinav[3], P. V. S. K. Yashant[4], Dr. Vijayalakshmi Kakulapati[5]
[1,2,3,4]*Students,* [5]*Professor, Department of Information Technology*
Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India

Abstract- ***The following research centers on the disciplines of applied science such as computer vision, machine learning. As a result of an increase in the usage and popularity of automobiles in numerous cities, the incorporation of intelligent traffic sign recognition technology into these automobiles has become an absolute necessity. The concern arises from drivers' incompetence to notice traffic signs under different circumstances on time. If this problem could be resolved, it may result in a drastic decline in the overall death toll from severe incidents. Thus, the identification of traffic indicators turns out to be the primary focus. Adopting a system that can automatically recognize traffic signs and enlighten the driver is essential, especially when drivers are exhausted, ill or intractable. Hence, the transition of a hazard-involved habitat into a safe and sound habitat requires extensive technological assistance. For a very long time, AI computations have been utilized in a variety of applications. A more precise, information-driven process, such as this one, can be incredibly beneficial for public authorities and citizens seeking to adapt more rapidly. Lastly, this thesis depicts that our new approach is more accurate than the previous ones at predicting traffic signs.***
Keywords- Torchvision, Opencv, ResNet, Image augmentation, CNN, Performance Metrics.

I. INTRODUCTION

The objective of traffic lights, traffic signs, and other traffic equipment is to alert, instruct, monitor, and notify people. They contribute to the achievement of a suitable standard of road traffic management and promote safety by facilitating the orderly and effective mobility of all pedestrians and vehicle traffic. Because their designs and colors are immediately noticeable from their surroundings, traffic signs are intended to be quickly recognized by motorists. Traffic signs could be ideogram-based, consisting of basic ideographs that convey the sign's meaning, or annotation-based, in which case the sign's contents may consist of arrows, text, or other symbols.

Detecting traffic signs is identical to object detection in computer vision, specifically, the identification of image sections with bounding boxes that include traffic signs. Compared to everyday objects, traffic signs are often created with stiff, straightforward forms and consistent, appealing hues. The spatial link with other visual items along the road is another crucial signal that might be used in the establishment of traffic sign detection. Despite the fact that the issue has gained academic attention in the computer vision field for almost two decades, there are still obstacles to overcome, such as varying lighting conditions, open environments, and capture angles. Numerous machine learning algorithms, such as linear discriminant analysis, support vector machines, random forest, and ensemble learning, were used to classify traffic signs. Machine learning approaches ended up providing several advantages for identifying traffic signs, but they were incapable of handling factors such as aspect ratios and picture sizes, which must be executed manually. Thus, the process of generating features was always long overdue and prone to inaccuracy. In contrast, deep learning approaches, particularly CNNs, or Convolutional Neural Networks, have demonstrated outstanding performance in a number of computer vision applications. This thesis focuses on advancements in deep learning and computer vision approaches for detecting and recognizing elements in traffic scenes. Prior to the widespread use of convolutional neural networks, the leading methods for detecting traffic signs centered on a variety of feature extraction techniques and machine learning.

A. ResNet

Kaiming He and his colleagues came up with the idea of the Residual Network, or ResNet, with the objective of developing a more in-depth network that is able to circumvent the impact of the vanishing gradient problem. ResNet topologies are capable of being implemented with a variety of different numbers of layers, including 34, 50, 101, 152, and even a gigantic 1202. The ResNet 16 architecture was the one that was used the most, and it featured a total of fifty layers. The ResNet network is a standard example of one having residual connections. The following equation may be used to describe the output of a residual layer after it has been completely processed: xl is defined as the output of a residual layer. The formula for xl is as follows: xl = (xl−1) + xl−1 . As a

result, xl−1 is produced by using the results from the layer below it as input. After conducting further processes such as batch normalization and convolution with filters of varying sizes, the result is represented by the function (xl−1), which is then used in conjunction with an activation function such as ReLU. The residual networks are often made up of many of the most basic residual blocks. However, the activities that take place inside the blocks might vary to correlate with a variety of residual designs. Recent times have seen the introduction of a number of enhanced residual networks. In addition, a number of scholars have coupled residual units with Inception, and the equation that may be used to explain this combination analytically is as follows: xl = F(xl−1 3×3⊙xl−1 5×5) + xl−1, where ⊙ represents the concentration operations that take place between the two outputs generated by the 3x3 and 5x5 filters. Thereafter, the convolutional operation is carried out, and then the outputs of that operation are connected with the inputs of the xl−1 block.

B. Convolutional Layer

The convolutional layer is the primary component in the architecture of a CNN. It includes using a collection of kernels, also known as filters, that can only accept a tiny portion of the input volume yet extend over the whole of the volume's depth. Each filter may be learned, and it will be used throughout the height and breadth of the input volume, in addition to the calculation of the dot product that will be performed between the entries of the filters and the input. During the first stage of the procedure, a feature map in two dimensions will be created. In the end, the network will learn from the activated filters while identifying certain characteristics at particular spatial positions in the input. The result is created by storing the activation maps for each filter all along the depth dimension. This allows the layer to produce its complete output. The parameters of each neuron in the same activation map will be shared, and the output of a neural network will be a tiny area of the input. The dimensions of the output volume produced by the convolutional layer are determined by the values of three hyperparameters: stride, zero-padding, and depth. The value of the stride parameter determines how the depth of columns is distributed over the spatial dimensions that are defined by the width and height of the input. It is required that the value of stride (S) be larger than both zero and any integer. In actuality, the lengths of S are almost never more than three. Because there will be less overlap between the receptive fields, the output volume will have reduced spatial dimensions even while the stride length will continue to grow. The spatial size of the output volume will be controlled by the zero-padding. The number of neurons in a layer that connect to the same part of the input volume may be limited by adjusting the depth of the output volume.

II. LITERATURE SURVEY

Since the publication of the first study on the subject in road sign recognition has developed into the most significant areas of research. Since that time until the present, there have been a large number of research groups working on the subject. Each of these organizations has sought to find a solution to this issue by using a unique strategy. Although at first glance, the primary steps toward a remedy appear to be very clearly outlined and uncomplicated, the specifics of the techniques that were employed reveal that there are several possibilities and a great number of notions about how accurate results might be obtained. To this point, no one solution technique has established itself as the clear front runner, and it is obvious that it will be some time before solutions start to appear. Detection and recognition are the two primary steps that are required for the process of determining the identity of a road sign. There are three distinct ways of implementing them. The first refers to the colors used on traffic signs as significant pieces of information that may be used to identify and categorize traffic signals. The second depicts that it is possible to recognize traffic signs based just on their shapes, whereas the third outlines that the combination of color and shape is what constitutes the essential component of any road sign recognition system.

The road sign recognition and tracking system that was developed by Fang and colleagues uses a video camera to process color pictures, which are converted into the HSI System. Utilizing a neural network with two layers allows for the extraction of color characteristics from the hue. An edge detection approach collects gradient values in certain color areas in order to generate an edge picture. This edge image is then sent as input to a second two-layer neural network, which is responsible for extracting shape characteristics. A fuzzy method is used to generate an integration map, which includes the combination of color and shape information. Ohara et al. made use of a short and straightforward neural network to determine the color of traffic signs as well as their shape. Initially, a Laplacian of Gaussian filter is applied, and then the original color picture is transformed. After that, a color neural network classifier is used on the picture in order to segment it according to the RGB color space and the color that is being recognized. After that, a shape NN is applied to each picture in order to determine whether or not it has anything that resembles the form of a road sign. After the shape has been located, the final identification is achieved by template matching. Shirvaikar was successful in developing an automated system for the identification and interpretation of traffic signs. An analysis of

spatial features is performed on potential locations of interest. The aspect ratio, fill factor, and area in pixels are all factors that are considered while choosing the appropriate signs to display. In order to get more precise findings, a relational feature analyzer was used. In real time, the system can recognize traffic signals such as yield, speed, and stop indications.

The classification of traffic signs is a common application of neural networks. This is due to a number of factors, but the primary one is the high level of accuracy that can be attained by using this classifier. The decade of the 1990s was a high point for research in neural networks, perhaps due to the fact that knowledge of utilizing neural networks was quite prominent during that era. It should not come as a surprise that a lot of the evaluated studies make use of neural networks since some of the publications examined pertain to that time period. Moreover, numerous researchers intend to steer clear of utilizing conventional classification methods and instead experiment with innovative approaches.

A neural network was used by Vitabile et al. in order to categorize the potential sign areas based on the information contained within them. A feed forward neural network classifier is used to carry out the classification process. In this process, a candidate image that is 36 by 36 pixels in size is supplied into the neural network input.

III. PROPOSED ALGORITHM

A. Data Aggregation

The German Traffic Sign Recognition Benchmark (GTSRB) contains more than 50,000 annotated images of 40+ traffic signs, such as stop signs, bicycle crossings, and speed limits of 30 km/h, 60 km/h, etc. There are some video clips too.

Architecture:

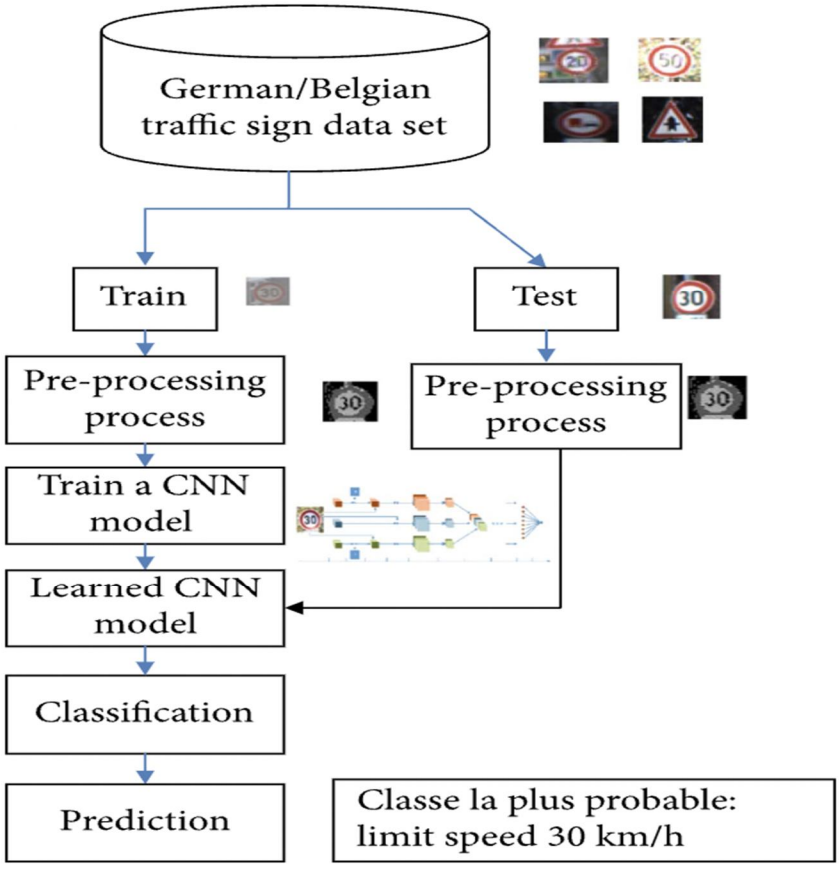


Fig 1: Architecture of our model

1) *Exploration and Building a Dataset:* The images of each traffic sign are kept in their own individual directories. So, including all of these unique directories, we have a total of 43. Now, in order to load and display pictures, we will build three utility methods that make use of OpenCV and Torchvision. Every large existing dataset includes a number of photos that have been annotated with the kind of sign and where they are located. That is the minimal necessity for any dataset pertaining to traffic

signs. The German Traffic Sign Recognition Benchmark (GTSRB) dataset is superior to all other datasets in terms of the number of classes and photos it contains. The KUL set is likewise rather huge and has the largest number of classes of any other set. Nevertheless, the STS and KUL datasets both have the following important advantage: They include the whole frame, which makes them helpful for systems that do detection as well asclassification. This is particularly important in light of the fact that the project is now in the detection phase. In addition to that, the STS data set contains the following extra annotation data: Details on the visibility of a sign, including whether or not it is blurry or obstructed, as well as whether or not it belongs to the main road or a side road. Every dataset stores its annotations as comma- separated text files known as csv-files, but these files might take on a variety of forms. We will transfer the picture files to a new directory so that it is simpler for us to make use of the dataset assistance provided by Torchvision. Eighty percent of the photographs will be set aside for training. Ten percent will be used for validation, and ten percent will be used for testing for each class. We will copy each picture into the appropriate directory for the dataset.

Fig 2. Traffic Signs-1

Fig 3. Traffic Signs-2

B. Data Pre-Processing

Data preprocessing is the initial stage in the workflow of deep learning, whose primary aim is to prepare the available data into a format that our network can understand. This step generally includes filling the missing values, treating the outliers and soon.In case of image datasets, preprocessing operations can be done using tools like MATLAB and other ToolBoxes. By using the datastores and functions that are offered by MATLAB and the Deep Learning Toolbox, we are able to preprocess the images that are being

input with actions such as resizing. When it comes to labelling, processing, and supplementing deep learning data, other MATLAB toolboxes include functions, datastores, and applications, respectively. We are also able to handle data for areas such as image processing, object identification, semantic segmentation, signal processing, audio processing, and text analytics by using specific tools that are included inside other MATLAB toolboxes. In order to train a network and then generate predictions based on fresh data, the size of our pictures has to correspond to the size of the network's input. In the event that we need to modify the size of your photographs so that they are compatible with the network, we may either rescale or crop your data to achieve the desired size. By applying randomized augmentation to your data, we are able to effectively expand the quantity of training data that is available. The ability to train networks to be invariant to distortions in picture data is another benefit of using augmentation. For instance, we may make a network insensitive to the presence of rotation in input pictures by rotating them in a random manner before feeding them into the network. An augmented image data store offers a straightforward method for applying a limited number of augmentations to two- dimensional pictures so that classification issues may be addressed. We may begin with a built-in data store in order to do more complex preprocessing procedures, such as preprocessing photos for regression issues or preprocessing three-dimensional volumetric images. By using the transform and combine features, we also have the ability to preprocess photos in accordance with our own workflow.

C. Image Augmentation

The size of our training dataset will be artificially inflated with the help of various picture augmentation methods that we will employ. We resize it at random, rotate it, and then flip it horizontally. Finally, we standardize the tensors by applying certain predetermined values to each channel. This is necessary in order for the pre-trained models to function properly in Torchvision. Image augmentation is a method that involves modifying already-present data
in order to generate more data for the process of training a model. To put it another way, it is the process of artificially increasing the dataset that is accessible for use in the training of a deep learning model.

D. Pre- Trained Model

Our model will be given unprocessed picture pixels, and it will attempt to categorize those pixels as belonging to one of four different traffic signs. Therefore, it is not an easy task to construct a model from beginning. In this section, we will use transfer learning in order to replicate the structure of the widely used ResNet model. In addition to that, we will make use of the weights that the model has learnt as a result of its training on the ImageNet dataset. Torchvision simplifies the operation of all of these features.
Transfer learning is a method that enables us to utilize a model that has been trained for one job as a starting point for a machine learning model that will be used for another task. This helps us to save time and effort when developing new models. Take, for instance, the case where a model is trained to do image classification using the ImageNet dataset. In that scenario, we may take this model and "retrain" it to identify classes even though it was never originally programmed to do so!
Deep Residual Network was one of the most innovative pieces of work to come out of the deep learning and computer vision communities in the last several years. It is conceivable to train ResNet with hundreds or even thousands of layers, and it will still achieve appealing performance. ResNet makes this capability possible. The performance of a wide variety of computer vision applications, including picture classification, has been improved by taking advantage of its great capacity to represent information. These applications include object identification and face recognition.

E. Training

In this step, you will load a ResNet34 model that has been pretrained using the ImageNet dataset. Establish image augmentation as described in the previous paragraph. It is a method that improves the model's ability to generalise to new situations. You supplement the training set with a significant number of fabricated cases. These examples are derived from ones that already exist. However, you alter them in some way, such as by rotating them by a few degrees, altering the lighting, zooming in on them, etc. I combined the following alterations: rotating the image by up to 20 degrees; adjusting the lighting by up to 80 percent; and zooming in by up to 20 percent. Lighting augmentation is quite important. In the early phases of the project, I've seen that very dark images have the most incorrect predictions. More than a three-percent increase in validation accuracy may be achieved by the use of aggressive illumination enhancement. Lighting is changed by directly changing the values of the R, G, and B channels.

1) *Learning Rate:* An optimization algorithm has a tuning parameter known as the learning rate. This parameter defines the size of the step that is taken at each iteration as the algorithm works toward minimising a loss function. In an effort to apply discriminative fine-tuning, in which different learning rates are applied to various components of the model, we did our best. In

this particular scenario, we want to train the model's earliest layers less than its later layers. The first layers are more generalised than the subsequent ones. While being trained on the ImageNet dataset, these layers gained knowledge about patterns that are highly helpful for our purpose, and it is important to us that we do not forget this information. On the other hand, the most recent layers are highly specialized for a certain job, and we need to retrain them for our particular task. The stats did not improve as a result of this, unfortunately. If you apply a high learning rate to each layer of the model, the training process will go much more smoothly. We guess that the reason for this is that traffic signs are very different from dogs, cats, and airplanes, which means that information in the lower layers is not as useful as in other applications of computer vision. We will write 3 helper functions to encapsulate the training and evaluation logic. In that, we start with the train_epoch function primarily and some helper functions. We start by turning our model into training mode and going over the data. After getting the predictions, we get the class with the maximum probability along with the loss, so we can calculate the epoch loss and accuracy.

Figure 4: Epoch loss and Accuracy

For Unseen and Unknown classes:

We miss the traffic signals for a lot of different reasons. Some of the most common reasons are not being able to focus, being tired, or not getting enough sleep. Some of the other things that can cause people to miss the signals are bad lighting, the influence of the outside environment, and the weather. Still, it doesn't make sense to have someone next to us tell us to watch out for the traffic signs. Instead, it would be much better to have a system that can read the traffic signs and alert and guide the driver. So, we need to figure out what to do when we see images or signs that aren't clear or can't be seen. This will make our model better.

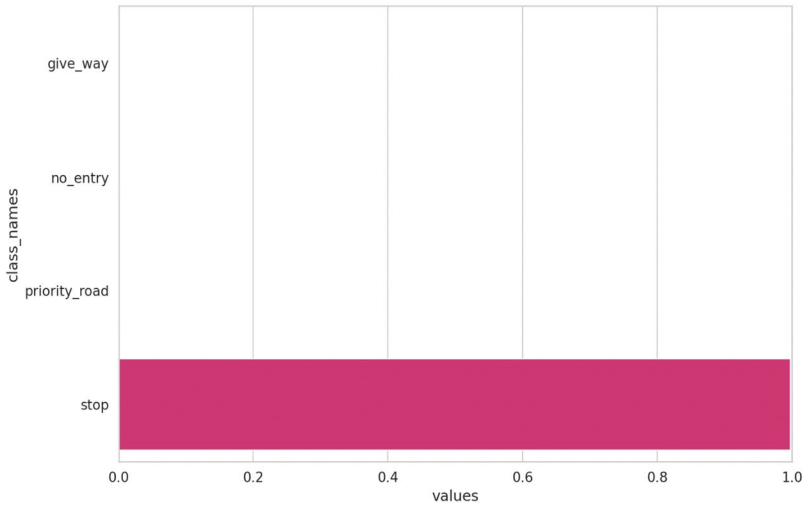


Figure 5: class_names and values-1

2) *Classifying Unknown Traffic Signs:* One of the challenges for our model is when it encounters a traffic sign that it hasn't seen before. So, here we classify those signs and we perform a prediction on them very first.

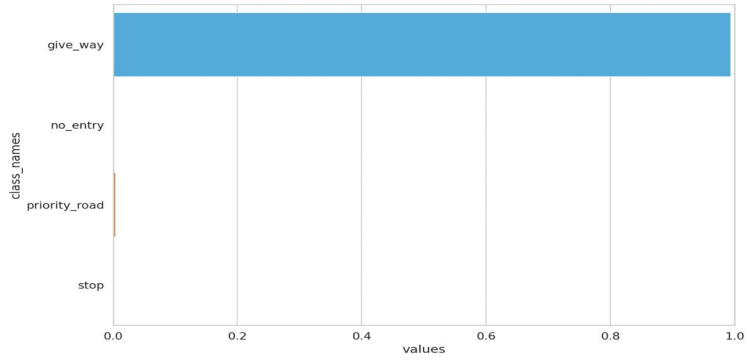


Figure 6: class_names and values-2

We are going to handle this situation in a straightforward manner. First, we will get the indices of all traffic signs that weren't included in our initial dataset. Next, we will make a new folder for the unknown class and copy some of the images there. Although there are many ways to handle this situation, we are going to do something straightforward. Now we are going to repeat the processes from before and go on to train the new data set, which comprises of signals that are unknown to us.

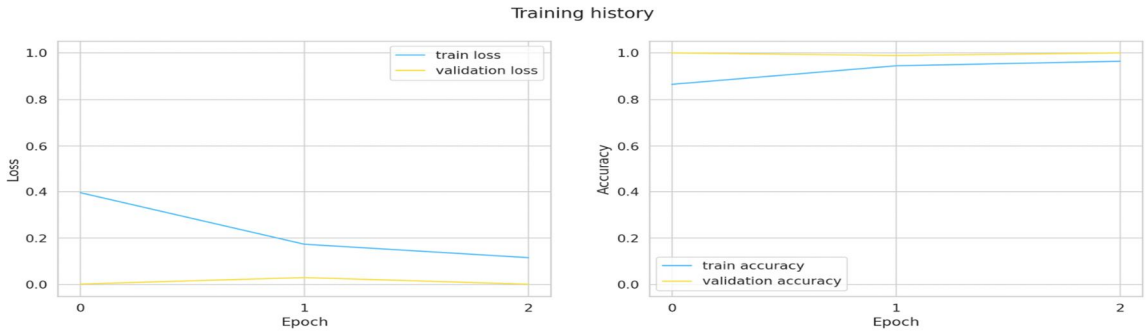


Figure 7: Training History

F. Evaluation and Performance Metrics:

Evaluating the performance of our algorithm for deep learning is an important component of our study. When we evaluate our model using one metric, such as accuracy score, we may get satisfactory results. However, when we compare it to other metrics, such as logarithmic loss or any other similar metric, we may get less than satisfactory results. The majority of the time, we gauge the effectiveness of our model based on its classification accuracy, yet this metric alone is insufficient to provide an accurate assessment of our model. Therefore, it is necessary for us to verify the classification report, confusion matrix, f1 score, and any other relevant metrics.

1) *Classification Accuracy:* Classification Accuracy is what we usually mean when we use the term accuracy. It is the ratio of the number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

Figure 8: Classification Accuracy

2) *Logarithmic Loss:* Log loss, often known as logarithmic loss, is a method that minimises the impact of incorrect classifications. It functions quite well as a multiclass categorization system. While working with Log Loss, the classifier is required to give a probability to each class for each and every sample. Considering that there are N samples for M classes, the Log Loss can be calculated as follows:

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

Figure 9: Logarithmic Loss

3) *Confusion Matrix:* As its name indicates, the Confusion Matrix provides us with a matrix as its output and summarizes the overall performance of the model. Let's pretend there is an issue with our binary categorization system. We have some examples that fall into one of two categories:"YES" or "NO." Additionally, we have our very own classifier that makes a prediction about the category that a certain input sample belongs to. The following is the result that we receive when we test our model on 165 different samples.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Figure 10: Confusion Matrix

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalSample}$$

Figure 11: Accuracy Formula

$$\therefore Accuracy = \frac{100 \dot{+} 50}{165} = 0.91$$

Figure 12: Computation of Accuracy

Confusion Matrix forms the basis for the other types of metrics.

IV. RESULTS

A. *Output with Jupyter Notebook (Predictions by our Model)*

Figure 13: Predictions by Classifier-1

Figure 14: Predictions by Classifier-2

1) Classification Report before Adding Unknown Signs:

	precision	recall	f1-score	support
give_way	1.00	1.00	1.00	216
no_entry	1.00	1.00	1.00	111
priority_road	1.00	1.00	1.00	210
stop	1.00	1.00	1.00	78
accuracy			1.00	615
macro avg	1.00	1.00	1.00	615
weighted avg	1.00	1.00	1.00	615

Figure 15: Classification Report before Addition of Unknown Signs

2) Classification Report After Adding Unknown Signs:

	precision	recall	f1-score	support
give_way	1.00	1.00	1.00	216
no_entry	1.00	1.00	1.00	111
priority_road	1.00	1.00	1.00	210
stop	1.00	1.00	1.00	78
unknown	1.00	1.00	1.00	169
accuracy			1.00	784
macro avg	1.00	1.00	1.00	784
weighted avg	1.00	1.00	1.00	784

Figure 16: Classification Report after Addition of Unknown Signs

3) Confusion Matrix before Adding Unknown Signs:

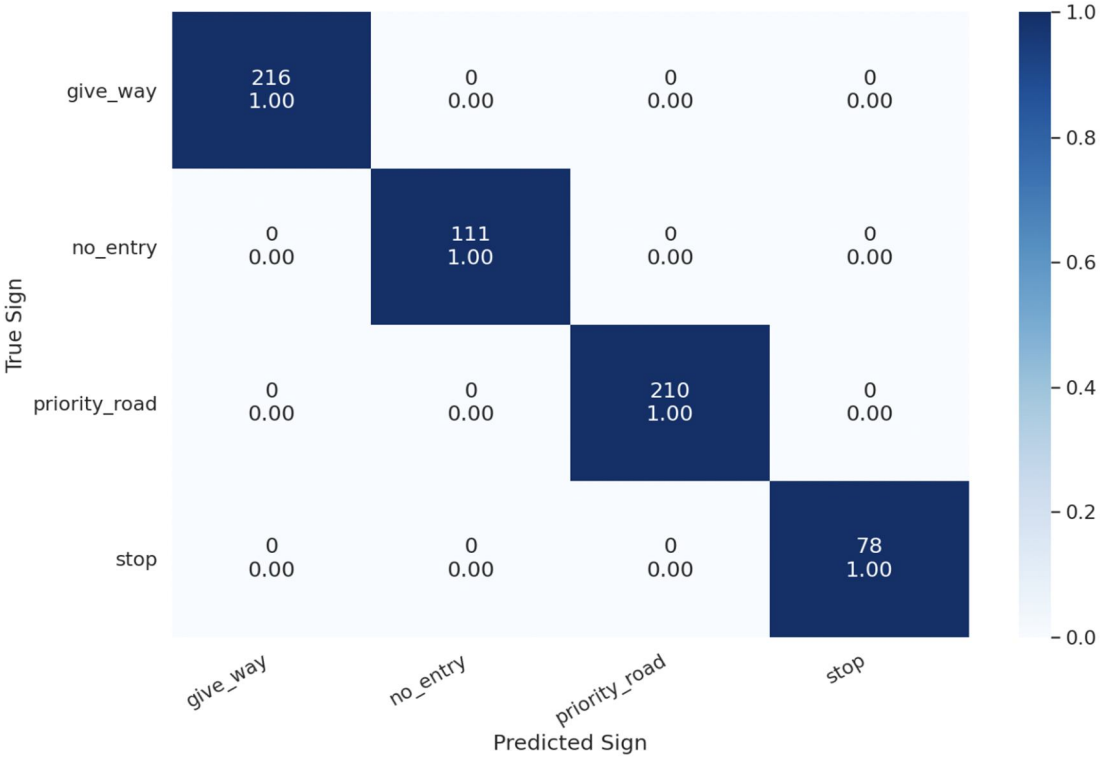


Figure 17: Confusion Matrix before Adding Unknown Signs

4) Confusion Matrix after Adding Unknown Signs:

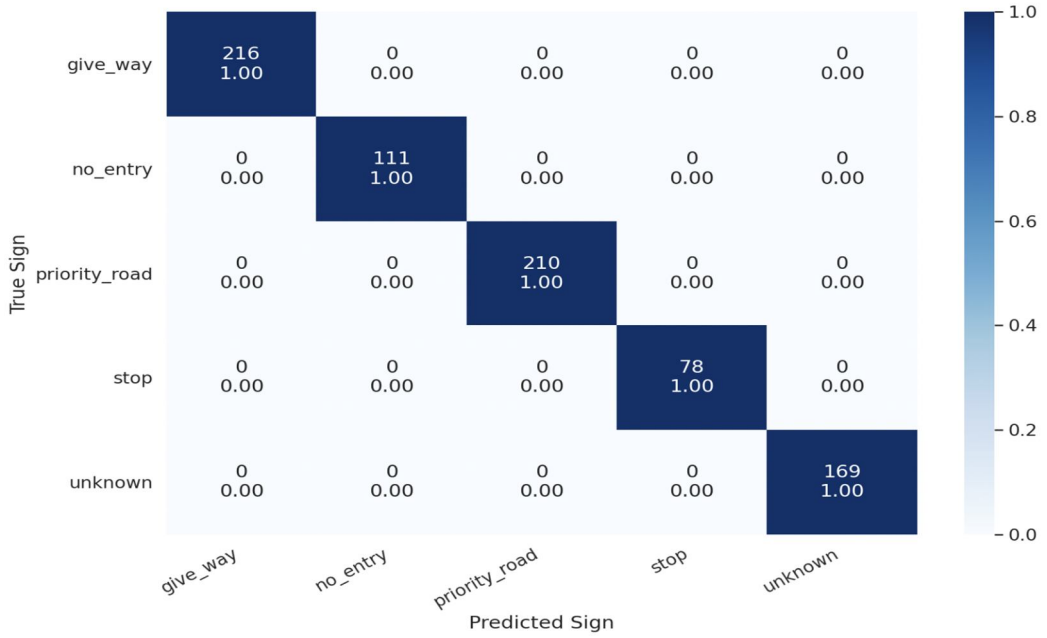


Figure 18: Confusion Matrix after Adding Unknown Signs

5) Accuracy Score:

```
accuracy_score(y_test, y_pred)*100

99.67479674796748
```

Figure 19: Accuracy Score of our Model

B. Output with Android Studio (App)

Figure 20: Start Screen (or) Splash Screen

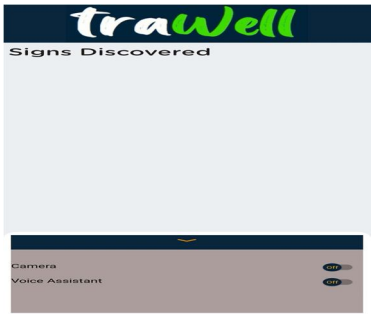


Figure 21: User Interface of the app "traWell"

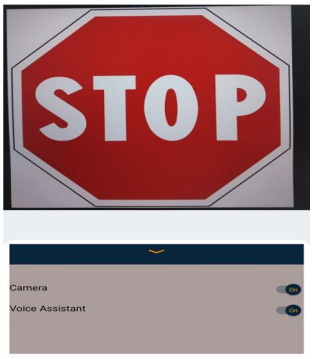


Figure 22: Interface after Launching Camera

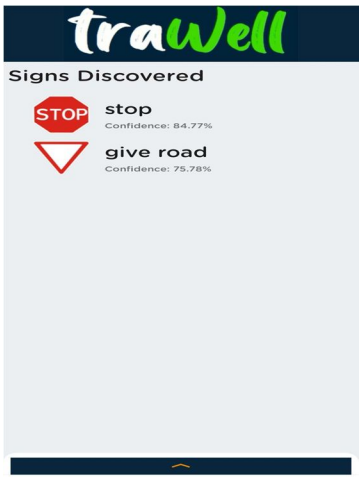


Figure 23: Interface after Signs Discovered

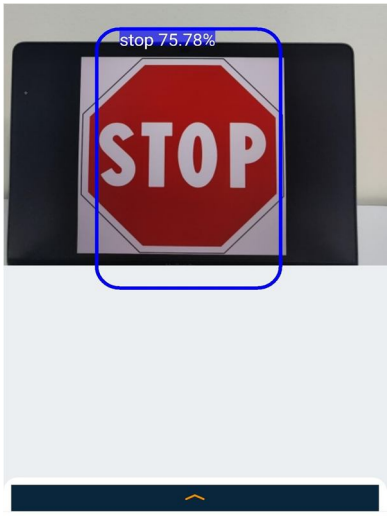


Figure 24: Detection and Recognition of Traffic Sign on Camera

V. CONCLUSION

This project aims at the implementation, testing, and deployment of a road traffic sign identification system that could assist the driver or a machine (in the case of a driverless vehicle) in detecting the road traffic signs. In order to achieve the end goals of this project, the methodologies and algorithms related to computer vision and deep learning technologies are primarily used along with Android app development. Since we used the resnet model as a part of sign classification, the performance of the final application was very efficient and satisfactory. The statement is supported by a 99 percent accuracy rate. The deployment of our application on Android devices presented us with a smooth workflow and accurate results. As with any work, our project has a few limitations. For instance, the camera being used might not be capable of capturing high-resolution pictures or videos, which might lead to a considerable downfall in performance. However, there is a lot of scope for the addition of fresh features to the application with the gradual development in technology as well as with the easier availability of technological resources.

VI. FUTURE SCOPE

First and foremost, the project may be expanded to identify traffic signs from multiple nations. It's possible that this will include the identification of road signs printed in a variety of languages. Despite the fact that outstanding results have been achieved, the challenge of traffic text recognition has not been tackled. It is possible for a recurrent neural network (RNN) to recognise traffic text. The performance of the application can be enhanced further with the help of the results of the traffic text recognition. If this kind of technology is combined with a global positioning system (GPS), it will be possible to provide the driver pertinent

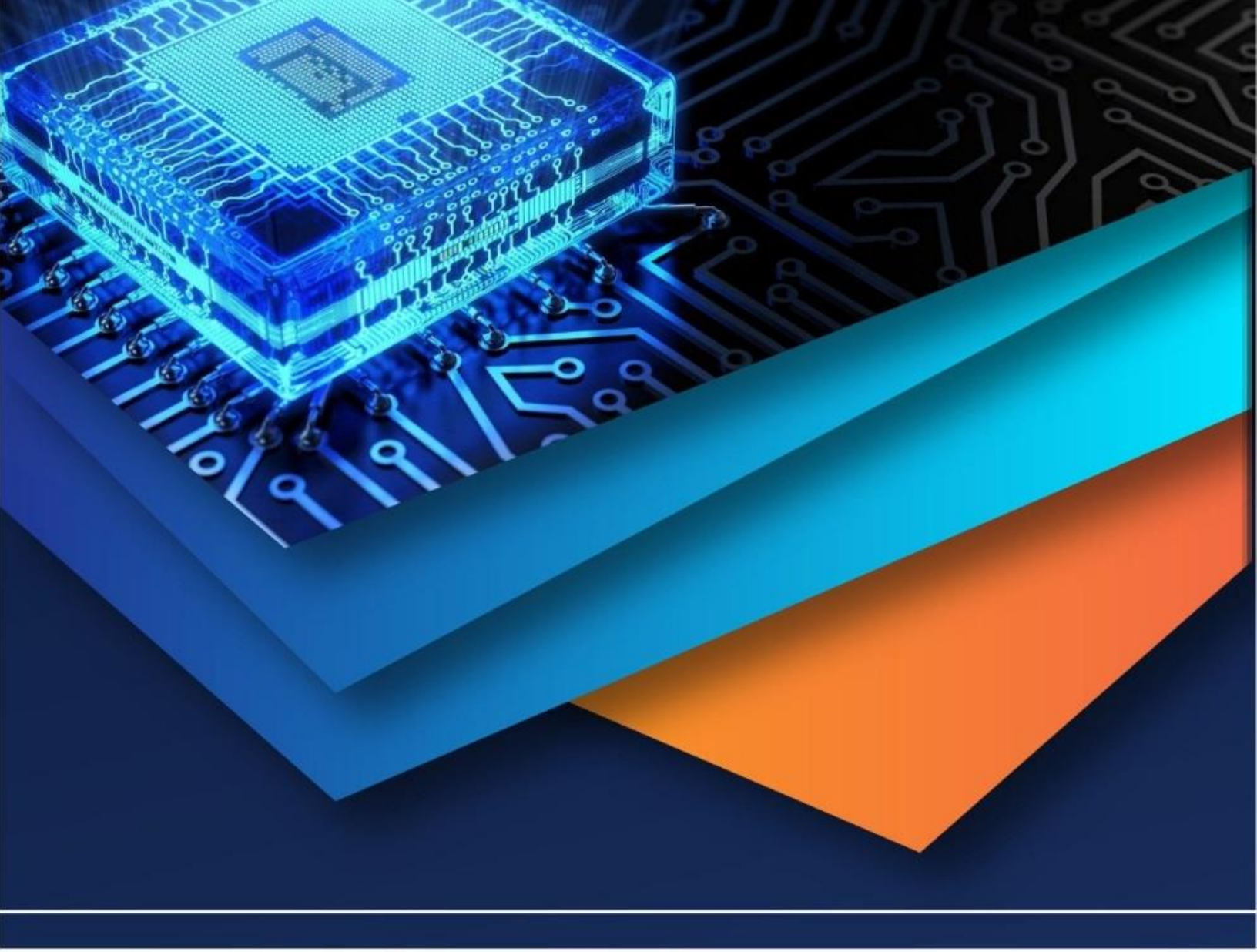
information on the real speed limit that is in effect on a particular route. When the speed reading from the GPS is compared to the posted limit, the driver might get a warning if either the speed limit is exceeded or if the vehicle does not come to a complete stop before a stop sign.

ACKNOWLEDGMENT

We would like to express our deep gratitude to our professor, Dr. Vijayalakshmi Kakulapati, for guiding us through the process of writing this research paper.

REFERENCES

[1] A. de la Escalera, J. Armingol, and M. Mata, "Traffic sign recognition and analysis for intelligent vehicles," Image and Vision Comput., vol. 21, pp. 247-258, 2003.
[2] Bedi Rajni et al., "Neural network based smart vision system for driver assistance in extracting traffic signposts" in Cube International Information Technology Conference, ACM, pp. 246-251, 2012.
[3] C. Fang, S. Chen, and C. Fuh, "Road-sign detection and tracking," IEEE Trans. on Vehicular Technology, vol. 52, pp. 1329-1341, 2003.
[4] E. Dickmans, "Machine perception exploiting high-level spatio-temporal models", AGARD Lecture Series 185, 1992-Sept.-17.
[5] J. Crissman and C. E. Thorpe, "UNSCARF a color vision system for the detection of unstructured roads", Proc. IEEE Int. Conf. Robotics and Automation, pp. 2496-2501, Apr. 1991.
[6] Jia Yangqing et al., Caffe: Convolutional Architecture for Fast Feature Embedding, pp. 675-678, 2014.
[7] J. Miura, T. Kanda, and Y. Shirai, "An active vision system for real-time traffic sign recognition," presented at 2000 IEEE Intelligent Transportation Systems, Dearborn, MI, USA, 2000.
[8] L. Davis, "Visual navigation at the University of Maryland", Proc. Int. Conf. Intelligent Autonomous Systems 2, pp. 1-19, 1989.
[9] "Neural network based autonomous navigation", Vision and Navigation: The Carnegie Mellon Navlab,1990.
[10] P. Paclik and J. Novovicova, "Road sign classification without color information," presented at Sixth Annual Conf. of the Advanced School for Computing and Imaging, Lommel, Belgium, 2000.
[11] P. Paclik, J. Novovicova, P. Pudil, and P. Somol, "Road sign classification using Laplace kernel classifier," Pattern Recognition Letters, vol. 21, pp. 1165-1173, 2000.
[12] Peng jinzhang, "A study on Convolutional Neural Networks for Traffic Sign Recognition", Beijing Jiaotong University, 2017.
[13] Research and Application of Traffic Sign Detection and Recognition Based on Deep Learning | IEEE Conference Publication | IEEE Xplore.
[14] Road traffic sign detection and classification | IEEE Journals & Magazine | IEEE Xplore.
[15] S. Vitabile, G. Pollaccia, G. Pilato, and F. Sorbello, "Road sign Recognition using a dynamic pixel aggregation technique in the HSV color space," presented at 11th Inter. Conf. Image Analysis and Processing, Palermo, Italy, 2001.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY