INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE

ISSN: 2057-5688

Vehicle Traffic Sign Recognizer Using Deep learning-based CNN

[1]HARIKA RAMISETTY, [2]M.NARESH

[1]PG Scholar, Dept. of MCA, Newton's Institute of Engineering, Guntur, (A.P)

[2]Associate professor, Dept. of CSE, Newton's Institute of Engineering, Guntur, (A.P)

Abstract: Different applications, such as autonomous vehicles, traffic management, and traffic monitoring, have vastly different requirements for recognising traffic signs. The ADAS framework for traffic sign recognition (TSR) is one such component. Drivers may benefit from the TSR framework because of the wealth of information provided by street signs. The automotive industry has expanded greatly, and some companies are aiming to develop fully autonomous cars, in which traffic sign recognition is a crucial component. A model is developed that uses a convolutional neural network to understand what the traffic signals mean. This model may also be used in everyday cars to flag the driver down about potentially hazardous traffic signs based on their content.

Keywords: **Convolution neural network, Adam optimizer, Traffic Sign.**

I INTRODUCTION

Typical traffic sign recognition (TSR) frameworks include two stages: recognition and arrangement. Some TSR frameworks also have a third step designed between detection and Classification to handle video sequences. When it comes to resolving issues with picture classification and identification, deep learning emerges as the most effective subset of machine learning. There is no denying deep learning's success in the realm of autonomous vehicles. DNNs have been used in a wide variety of domains, including scene semantic division [2], traffic signal identification [3], crosswalk arrangement [4], [5], traffic sign position [6], walker investigation [7], vehicle heading course evaluation [8], and many more.

The planned study will concentrate on methods for identifying traffic signs. Customers on the street might learn a lot by just looking at the traffic coming in. The creation of a trustworthy automated traffic sign detecting system is crucial. While Deep Learning has many benefits, it's not without certain risks. Training a deep neural network demands a big, evenly distributed data set, which in turn necessitates a huge computing investment.

Some traffic signs are unusual and infrequently observed in everyday driving situations, thus a comprehensive dataset covering all aspects of traffic is necessary to complete this assignment.

In general, traffic sign recognition may be broken down into the following three methods: (i) "AdaBoost based detection," (ii) "Support Vector Machine," and (iii) "Neural Network" based detection.

Images are utilised from the German Traffic Sign Detection Benchmark (GTSDB) [9], which features real-world situations of German streets with traffic signs.

II **LITERATURE SURVEY**

In spite of the widespread use of deep learning techniques for Handwritten Character Recognition in languages like Chinese, Arabic, English, etc., the majority of previous work in this area has instead relied on more conventional methods when applied to Tamil. Pre-processing, character segmentation, feature extraction, classification, and character prediction are the usual steps in a standard HTCR workflow that use classic machine learning methods.

Tamil character recognition was first given by Siromoney et al. (1978), who proposed converting characters from a binary matrix to an encoded string and then building a dictionary. When a new character was introduced, it was checked against the dictionary strings.

An OCR system for Tamil printed letters was created by Aparna and Ramakrishnan (2002) based on spatial occupancy and extensions of strokes. The Tamil text was broken up into three halves, and the segments' occupancies were recorded in order to categorise the text into four groups. On the basis of the extensions, subgroups of the classes were created. Geometric moments and DCT were used for feature extraction, and a nearest neighbour classifier was used for classification. Using sample passages from periodicals and books, the system was able to achieve a 98% rate of correct identification.

A single-hidden-layer multilayer perceptron neural network-based method was described by Sutha and Ramaraj (2007). Fourier descriptors were used in the feature extraction process. The system's accuracy, after being trained with a unique dataset, was 97%.

Bhattacharya et al. (2007) introduced a two-stage recognition technique in which unsupervised clustering was utilised in the first stage to classify characters into groups, and a supervised classifier was used in the second stage to recognise them, leading to an accuracy of 89.66%. The first step included dividing the input picture into 7x7 blocks and scanning it horizontally and vertically to count the number of white-to-black and black-to-white transitions, the latter of which would become the feature vector. The k-means clustering technique was used to categorise this feature vector of all the training data. Clusters created by valid sample data from several character classes were processed further in a second step.

Chain code histogram was used for feature extraction, and an MLP classifier was used for classification. Overall, after calculating it step by step, the recognition accuracy was found to be 92.77 percent.

Pre-processing, feature extraction, and classification were the three standard procedures used by Shanthi and Duraiswamy (2010). They had pre-processed the data by skeletonizing it, thresholding it, separating it into lines and characters, and standardising its size. The character picture was segmented into n-by-n zones, and the pixel density in each zone was used to determine which features should be extracted. Excluding the Sanskrit loanwords, a total of 106 classes were classified using the Support Vector Machines (SVM) classifier. They used their own 35,441-character dataset to train the model, while their test set of 6,048-character dataset only included 34 unique Tamil characters. For 34 characters, they were able to attain an accuracy of 82.04%.

Character glyphs' height, width, number of horizontal lines, number of vertical lines, number of circles, number of horizontally oriented arcs, number of vertically oriented arcs, centroid of image, position, and pixels in various regions were all extracted by Suresh Kumar and Ravichandran (2010). Several pre-processing methods, including skew detection, smoothing, thresholding, and skeletonization, were used before the features could be extracted. Input was the whole printed document, which was scanned in its entirety. Therefore, the picture is divided into its component characters, and then the characteristics are retrieved from those characters. Classifications of the retrieved features were performed using SVM, SOM, Fuzzy network, RCS algorithm, and Radial basis function. However, the training and testing dataset sizes were not specified.

With wavelet transform feature extraction and backpropagation neural network classification, Jose and Wahi (2013) obtained 89% recognition accuracy.

As far as we are aware, only Vijaya Raghavan and Sra's (2014) work using convolutional neural networks has been published. Using the HP Labs dataset, a CNN with 35 classes was built. It was stated that a combined effort of tanh activations, dropout, probabilistic weighting, tanh normalisation, and local response normalisation yielded an accuracy of 94.4%.

Offline Tamil handwritten character identification using structural and statistical characteristics was achieved by Raj et al. (2016). Quad tree construction relied on calculating pixel densities from the segmented character images. The SVM classifier was then used to these characteristics to determine their classification.

Deepa and Rao (2019) created a neural network and SURF descriptors-based closest interest point classifier for offline Tamil handwritten character recognition. For 156 classes, they achieved an accuracy of 90.2% using the HP Labs dataset.

III. CONVOLUTION NEURAL NET WORK Kera's

As a supervised learning technique, training a CNN requires the input of both data and an expected outcome. Their rankings serve as a scholarly template for further data analysis and are used to categorise the results.

A typical convolutional layer, pooling layer, and fully connected dense network make up the three main components of a CNN. A component map is obtained by passing the information image through a Convolutional layer, which uses mix channels. Next, the element map is sent into the maximum pool layer, which is primarily used for dimensionality reduction and selects just the most salient features from the original.

Last but not least, we smooth out all the highlights and provide them as input to the fully connected thick neural organisation, which learns the loads by backpropagation and provides the characterisation yield.

The idea of CNN is based on how the visual brain works; much as the visual cortex highlights one object in an image while blurring out the others, the CNN focuses on one window of data at a time. At all times, the convolutional layer of the CNN will provide a component map for every segment. To accomplish highlight extraction, the Pooling layer filters out superfluous highlights in favour of selecting the most important ones. Therefore, using CNNs eliminates the requirement for a separate component extraction step.

When compared to other characterising computations, CNNs take less setup time.

Traditional MLP (Multi-Layer Perceptron) computations are quite precise for image classification, Remarks Despite having entirely connected hubs, they suffer from the curse of dimensionality and so cannot be scaled to high-resolution images. To overcome these challenges posed by MLP, CNNs use the spatial connection between images. To do this, a network of interconnected neurons in close proximity to one another is constructed as a case study. Thus, when compared to other approaches, CNNs end up being unparalleled in Image order, Video Analysis, Natural Language Processing, and a large spectrum of diverse applications.

IV RELATED WORK

Before, just forms and colours were employed for traffic sign detection, which was not very effective or reliable. Accuracy in traffic sign identification has improved because to the development of deep convolutional datasets made possible by the explosion in the amount of available data. The process of recognising traffic signs consists mostly of detection and recognition. The fuzzy classification module suggested by Yuga hatolkar et al. [10] functions as an optimizer module for results acquired by CNN, making it a useful tool for traffic sign identification. It was a Loss and Accuracy gauge. Over the last decade, Convolutional Neural Networks (CNN) have made great strides towards improving the traffic detection system's accuracy and dependability. Wu et al. [11] first used convolutional neural networks (CNNs) to tackle traffic sign identification as a potential area classifier. In their unique architecture, given by Zhu et al. [12], the area of interest is created using a fully convolutional network. Proposals and CNNs were utilised for categorization of small traffic signs. Accuracy, precision, and recall % were determined using a genetic algorithm and convolutional neural network domain transfer learning by Jain A. et al. [13]. When calculating accuracy, Zhang J [14] employed CNN and knowledge distillation, but found that 70% accuracy was lost after trimming the model. S. Mehta [15] tested and trained accuracy and loss using the CNN SoftMax activation function, the RELU activation function, and the Adam optimizer. On the GTSRB dataset, Jin et al. [16] suggested using a Neural Network's Hinge Loss technique for Traffic Sign Recognition, which resulted in an accuracy of 99.65%. R-CNN is a novel method presented in recent years by academics. In [17], RBG presented a framework called R-CNN. As deep learning has progressed, computers' ability to recognise and recognise patterns utilising various convolutional network layers has improved. Motivated by these studies, we want to create a robust traffic sign identification framework that makes use of various aspects of layers of CNN in the same way.

The Methods Section

1)Dataset

In this publication, we make use of the German traffic sign dataset (GTSDB)[9], a database that has been employed in the vast majority of previous studies to classify and categorise traffic signs. The total number of photos utilised throughout training, validation, and testing is 31367.There are a total of 43 distinct categories in the dataset, including "1:'Speed limit (20km/h)', "2:'Speed limit (30km/h)', "3:'Speed limit (50km/h)', "4:'Speed limit (60km/h)', "5:'Speed limit (70km/h)', "6:'Speed limit (80km/h)', "7:'End of speed limit (80km/h)',

Passing banned (10), passing forbidden (11), right-of-way at junction (12), priority road (13), yield (14), stop (16), no vehicles (17), no vehicles above 3.5 tonnes (18), no entrance (19), general caution (20).

Twenty-odd degrees of danger on the left, twenty-one on the right, twenty-two on the double: The road is bumpy/slippery/narrowing on the right/road work/traffic signals/pedestrians/etc. The signs read as follows: "29: Children Crossing," "30: Bicycles Crossing," "31: Beware of Ice/Snow," "32: Wild Animals Crossing," and "33: End Speed + Passing Limits." Right ahead (34), left ahead (35), straight ahead (36), right or straight ahead (37).

Straight or left, right or left, roundabout required, no more passing, no more passing veh > 3.5 tonnes, and so on.

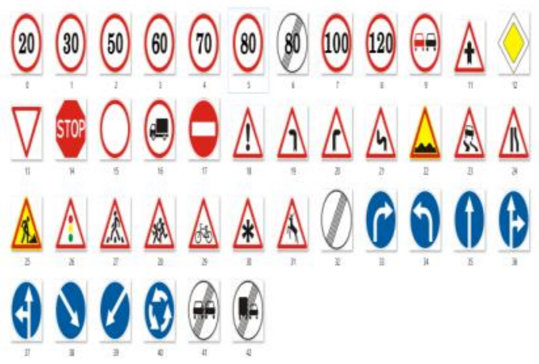

Figure 1: Preview of German Dataset

Below are the three histograms representing the class wise samples of images taken for (i) training, (ii) validation and (iii) testing. We had taken 80% of the dataset for training purpose and 20% for testing

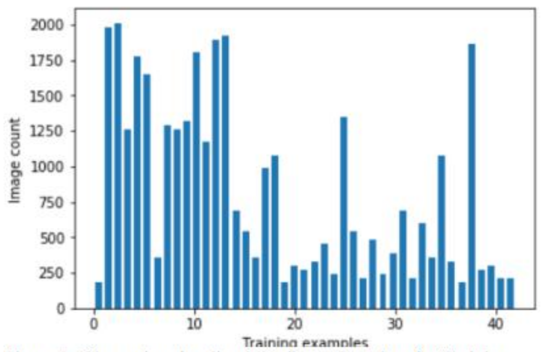

Figure 2: Class wise visualization of samples taken for Training

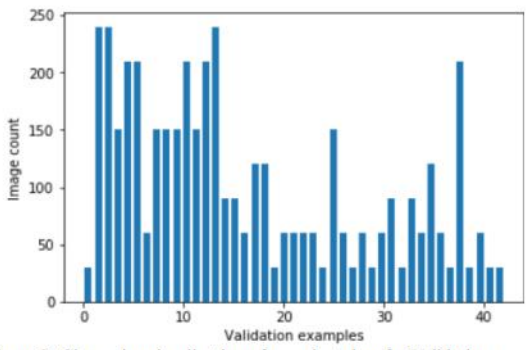

Figure 3: Class wise visualization of samples taken for Validation

2. The Laboratory Environment

The TensorFlow and keras libraries are used to realise the proposed work. The system has a 64-bit operating system, 8 GB of RAM, an Nvidia GeForce GTX 1050Ti graphics card, and a 7th-generation Intel Core i7 processor. Because of its redundancy, portability, and safety, we utilised the pillow library to read, manipulate, and save photos, and the OS module to iterate the dataset. It allows photos to be processed more quickly than in any other module.

3. The Structure of CNN

Our suggested CNN architecture, shown in Figure 6, was designed to categorise and recognise German traffic signs. The optimal filter size is determined via a series of trials.

When deciding on a filter size, it's important to keep computational overhead in mind. A 32-filter, 5-by-5-kernel is used to build the first three layers of a convolutional neural network. The dot product is calculated using a subset of the input photos as the kernel is slid over the dataset, and the result is a matrix representation of the dot product. Each layer employs the ReLu Activation function. The ReLU initiation function was repeatedly applied to the convolutional layer's output. This operation selects the output that must be sent to the hidden layer's neurons.

For inputs less than zero, this function returns zero. The ReLU function is shown graphically in the following.

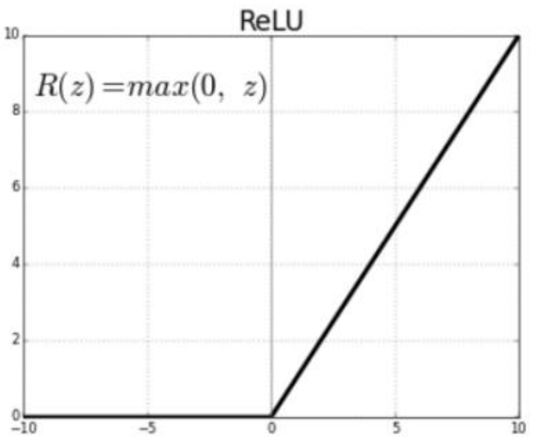

Figure 5: Graphical representation of ReLU Function

V CONCLUSION

In this research, we present a Convolutional Neural Network (CNN)-based system for recognising traffic signs, and early findings on the German dataset (GTSDB) show an accuracy of 96.19 percent. There are several layers of contextual information and convolutional features in the CNN architecture. During detection, the fused feature map with enough data is utilised to create area recommendations.

The aforementioned accuracy was achieved with the aid of the ADAM optimizer, which reduced computational and training costs.

We hope that by visualising any remaining faults and using more optimisation approaches, we can improve the project's accuracy and efficiency in the future.

REFERENCES

[1] C. Liu, S. Li, F. Chang and Y. Wang, "Machine Vision Based Traffic Sign Detection Methods: Review, Analyses and Perspectives," in IEEE Access, vol. 7, pp. 86578-86596, 2019, doi: 10.1109/ACCESS.2019.2924947.

[2] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff and H. Adam, "Encoder-Decoder with Atreus Separable Convolution for Semantic Image Segmentation", European Conference on Computer Vision (ECCV), 2018.

[3] K. Behrendt, L. Novak and R. Botros, "A deep learning approach to traffic lights: Detection tracking and classification", International Conference on Robotics and Automation (ICRA), pp. 1370- 1377, 2017.

[4] R. F. Berriel, F. S. Rossi, A. F. de Souza and T. Oliveira-Santos, "Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning

Approach", Computers & Graphics, vol. 68, pp. 32-42, 2017

[5] R. F. Berriel, A. T. Lopes, A. F. de Souza and T. Oliveira Santos, "Deep Learning-Based Large-Scale Automatic Satellite Crosswalk Classification", Geoscience and Remote Sensing Letters, vol. 14, no. 9, pp. 1513-1517, 2017.

[6] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li and S. Hu, "Traffic-Sign Detection and Classification in the Wild", Conference on Computer Vision and Pattern Recognition (CVPR), 2016

[7] R. Guidolini, L. G. Scart, L. F. Jesus, V. B. Cardoso, C. Badue and T. Oliveira-Santos, "Handling Pedestrians in Crosswalks Using Deep Neural Networks in the IARA Autonomous Car", International Joint Conference on Neural Networks (IJCNN), 2018.

[8] R. F. Berriel, L. T. Torres, V. B. Cardoso, R. Guidolini, C. Badue, A. F. D. Souza, et al., "Heading direction estimation using deep learning with automatic large-scale data acquisition", International Joint Conference on Neural Networks IJCNN, 2018.

[9] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing and C. Igel, "Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark", International Joint Conference on Neural Networks (IJCNN), 2013.

[10] Prasadu Peddi (2021), "Deeper Image Segmentation using Lloyd's Algorithm", ZKGINTERNATIONAL, vol 5, issue 2, pp: 1-7.

[11] Y. Wu, Y. Liu, J. Li, H. Liu and X. Hu, "Traffic sign detection based on convolutional neural networks," The 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, 2013, pp. 1-7, doi: 10.1109/IJCNN.2013.6706811.

[12] Y zhu,C zhang "Traffic sign detection and recognition using fully convolutional network".

[13] Jain, A., Mishra, A., Shukla, A. et al. A Novel Genetically Optimized Convolutional Neural Network for Traffic Sign Recognition: A New Benchmark on Belgium and Chinese Traffic Sign Datasets. Neural Process Lett 50, 3019–3043 (2019).

[14] Zhang, J., Wang, W., Lu, C. et al. Lightweight deep network for traffic sign classification. Ann. Telecommun. 75, 369–379 (2020).

[15] S. Mehta, C. Paunwala and B. Vaidya, "CNN based Traffic Sign Classification using Adam Optimizer," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 2019, pp. 1293-1298, doi: 10.1109/ICCS45141.2019.9065537.

[16] Prasadu Peddi (2023), Using a Wide Range of Residuals Densely, a Deep Learning Approach to the Detection of Abnormal Driving Behaviour in Videos, ADVANCED INFORMATION TECHNOLOGY JOURNAL, ISSN 1879-8136, volume XV, issue II, pp 11-18.

[17] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 1, pp. 142-158, 1 Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[18] Prasadu Peddi (2022), A Hybrid-Method Neighbor-Node Detection Architecture for Wireless Sensor Networks, ADVANCED INFORMATION TECHNOLOGY JOURNAL ISSN 1879-8136, volume XV, issue II.

[19] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.