WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Deep Learning Based Real Time Traffic Sign Board Recognition and Voice Alert System

Lalitha Bhavani Konkyana[1], Drakshayani Korada[2], Bhumika Sakalabhakthula[3], Padmaja Pusarala[4], Tejaswara Rao Kaja[5]

[1]Associate Professor, Department of Electronics and Communication Engineering, Aditya Institute of Technology and Management, Tekkali, Andhra Pradesh.

[2,3,4,5]Student, Department of Electronics and Communication Engineering, Aditya Institute of Technology and Management, Tekkali, Andhra Pradesh.

ABSTRACT:

Road signs are crucial for ensuring safety and predicting traffic flow. Visual attention driving, such as failing to read or misinterpreting traffic signs is a major factor in road accidents. The suggested method warns drivers through speaker and aids in helping them recognize traffic signs. This can be done by using the pyttsx3 library for him or her to choose suitable options. This project trained and classified the traffic signs using a Convolutional Neural Network (CNN). This was done using Open CV and in real-time using a webcam. In this project, the German Traffic Sign Benchmarks Dataset is employed. Traffic signs are detected with over 35,000 images of 43 different classes with the help of tensor flow and keras. A set of classes is built and trained in order to increase the accuracy of a system given data. About 98 percent of the execution was accurate. When the system detects a sign, a voice alert is sent to the driver through the speaker. The proposed system also has a feature where drivers of moving vehicles are informed of nearby traffic signs so they are aware of the laws they should observe.

Keywords: Convolutional Neural Network, Tensor flow, Keras, Open CV, Traffic Sign, GTSRB Dataset, Pyttsx3

1. INTRODUCTION

Autonomous vehicles have recently attracted a great deal of interest in research and development. Thus the first and most crucial idea to incorporate into any autonomous vehicle is a Traffic Sign Recognition system [1]. When it comes to road safety, drivers occasionally tend to miss traffic signs along the way. Under such circumstances, the automatic classification of traffic signs can highly reduce the number of traffic accidents while also ensuring total safety [2]. This technology was used in automobiles by several well-known names in automation using computer vision and machine learning techniques, deep learning techniques based on classifiers have largely replaced this approach. [3]Deep convolutional approaches have recently been visible to be the most successful in detecting objects. It is advantageous to view the classification and recognition of traffic signs from a deep learning perspective. Traffic sign classification is a challenging task that requires deep learning techniques and a large dataset [4]. The dataset is initially divided into a ratio where a certain number of images will go through one process and another number of images will go through a different process, showing the accuracy of the classes before it is further recognized. Despite the significant study history and ongoing effort on this topic, there are still several issues and drawbacks that need to be resolved.

Convolutional Neural Networks (CNNs) are a kind of neural networks that are frequently used in image categorization, object detection, and other computer vision applications [5]. The structure of CNN is typically composed of several layers, each of which performs a specific operation on the input data. Convolutional layers are the fundamental components of CNN. They take the input image and run a series of filters (sometimes referred to as kernels or weights) to extract features. [6]. These filters are learned through back propagation during training, allowing the network to identify patterns in the input data. A Convolutional Network's architecture is comparable to the human brain's neuronal connectivity pattern [7]. The arrangement of neurons in the human brain's visual cortex had an impact on the design itself. Neurons only react to input in a certain region of the receptive field, or field of vision. The visual region is made up of a variety of these receptive fields that enable humans to view objects. After being trained over a number of training epochs, or iterations, the model becomes capable of differentiating between some low level features and the dominant features in the images. The model uses the Soft max Classification [8] approach to classify them based on this training.

2. Literature survey

Real-time traffic sign recognition and voice alert systems have been extensively researched in the past decade, employing various methods such as image processing, machine learning, and deep learning algorithms. In this section, recent research papers that have used convolutional neural networks (CNN) and Python for traffic sign recognition and voice alert systems are presented. P. V. Pachpute et.al implemented a real-time traffic sign recognition system using deep convolutional neural networks (CNN). The system uses a pre-trained CNN model to classify traffic signs in real-time [9]. The authors achieved

an accuracy of 97.7% on the German Traffic Sign Recognition Benchmark (GTSRB) dataset. The system also includes a voice alert system that informs the driver about the traffic sign. T. Nguyen et.al presented a real-time traffic sign recognition system using a Raspberry Pi and deep learning techniques [10] . The system uses a pre-trained CNN model to classify traffic signs in real-time. The authors achieved an accuracy of 97.85% on the GTSRB dataset. The system also includes a voice alert system that informs the driver about the traffic sign. K. B. Dinh et. al worked on a real-time traffic sign detection and recognition system using You Only Look Once (YOLOv3) and CNN [11]. The system first detects traffic signs in real-time using YOLOv3 and then classifies them using a pre-trained CNN model. For the GTSRB dataset, the authors had a 99.1% accuracy rate. The system also includes a voice alert system that informs the driver about the traffic sign. Shi et .al studied a real-time traffic sign recognition system using transfer learning and deep neural networks [12]. The method employs a CNN model that has already been trained, which is then improved using the GTSRB dataset. With respect to the GTSRB dataset, the authors' accuracy was 99.02%. The system also includes a voice alert system that informs the driver about the traffic sign. Various researchers developed real time traffic sign detection using convolutional neural networks [13], [14], [15], and [16].

3. Methodology:

3.1Algorithm and flow of the model:

Pycharm IDE is used to write Python code for this project. Necessary libraries are imported one by one. Then the data set directory is created where the data is stored. The code is then separated into two.py files, one of which is Training_code.py and the other is Testing_code.py includes all necessary parameters, libraries for dividing the data set, a CNN model, as well as methods for image pre-processing and enhancement. The classification of the traffic signs is done continuously when looping in the file Testing_code.py.Importing each library one at a time in Training_code.py is the first step. Following that, some processing-related parameters were specified, including the path for the dataset and the CSV labels file, the image dimensions, the epochs, the validation ratio, etc. Then import all the images that will essentially count the number of classes. Then, using the ratio that was previously established in the parameters, data is divided into testing and validation. Also, all of the labels are put in the y variable, and all of the images are in the x variable. Then if the number of images and labels in the data set match is determined. Before the training process has begun, first CSV file is read and then plot and visualize ensuring that necessary data is being gathered and classified correctly. Moving onwards, preprocessing the images is conducted by making them grayscale and then equalizing them for consistency. Afterward, save the preprocessed photos to the x variable. Finally convolutional model is created. Following model compilation, training begins right away. A plot of the trained model will be displayed once the training is complete. It will primarily plot the accuracy and validation loss. Finally, exporting the results after testing our testing dataset. Some libraries are loaded into TrafficSigns2 Test.py. The settings like threshold and frame sizes are then easily set. Then adjust the camera's settings. The trained model file is imported after setting up the camera. The images are then once more preprocessed as before. Then class names need to be presented. Finally, while loop, which will continuously run and provide us with web camera photos, forecasts the image, the class name, and the likelihood that the classification was accurate

3.2 CNN model architecture:

In this paper, this work is implemented by considering the Lenet model and making some changes to it. There are seven layers in the LeNet-5 CNN architecture. The layer composition consists of three convolutional layers, two subsampling layers, and two fully linked layers. The LeNet model received two dropout layers and one additional convolutional layer. In this experiment, four convolution layers, two max-pooling layers, two dropout layers, two fully connected layers, and one flattening approach were used. This is shown in Figure 1. In this work, the first and second convolution layers with ReLU activation employ 32 kernels of size 3*3. There are 64 kernels of size 3*3 used for the third and fourth layers. Pool the max-pooling layers for both. Two drop-out layers are used with 0.25 and 0.5 drops, i.e., by setting the neuron values to zero, 50% of the neurons will shut down at each training phase. Two dense layers, one of which engages in "ReLu" activation and the other of which works in "soft max."

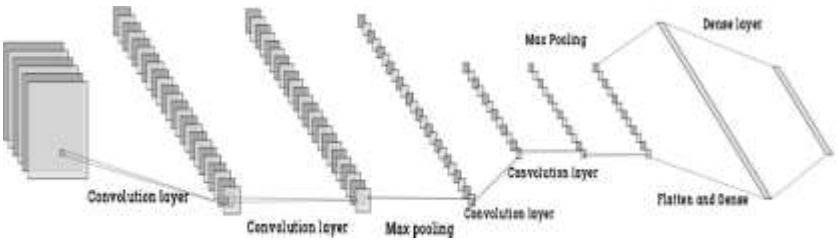


Fig 1 CNN model architecture

4. IMPLEMENTATION:

The implementation of this work can be divided into several steps, which are presented in Fig 2. These steps are: dataset collection, importing images and splitting them into training, validation, and test sets, checking the data for consistency, displaying imported images, preprocessing the images to normalize and standardize the lighting, building a convolutional neural network model, training the model using training and validation sets, evaluating the model's performance on the test set in real-time, and using the trained model to predict samples.

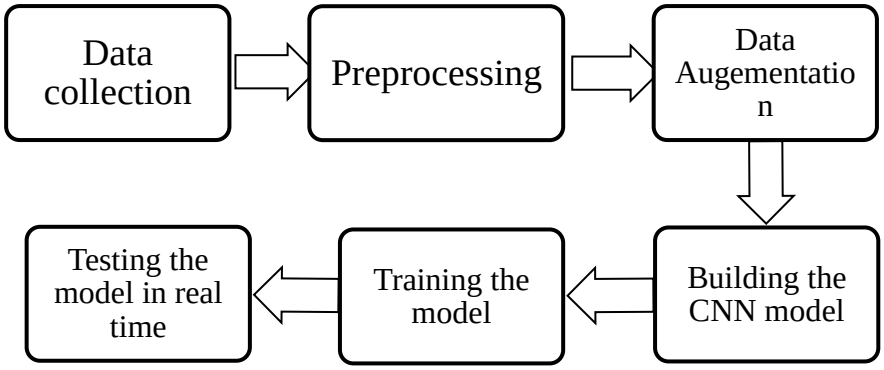


Fig 2 steps involved in implementation

4.1 Data collection:

The German Traffic Sign Benchmark (GTSRB) is a publicly available dataset for training and testing traffic sign recognition algorithms. It contains more than 35,000 images of 43 different traffic sign classes. The data used for this CNN is stored in a folder with separate subfolders for each class of image. Each subfolder contains images belonging to a particular class. Fig. 3 presents a sample of the GTSRB dataset.

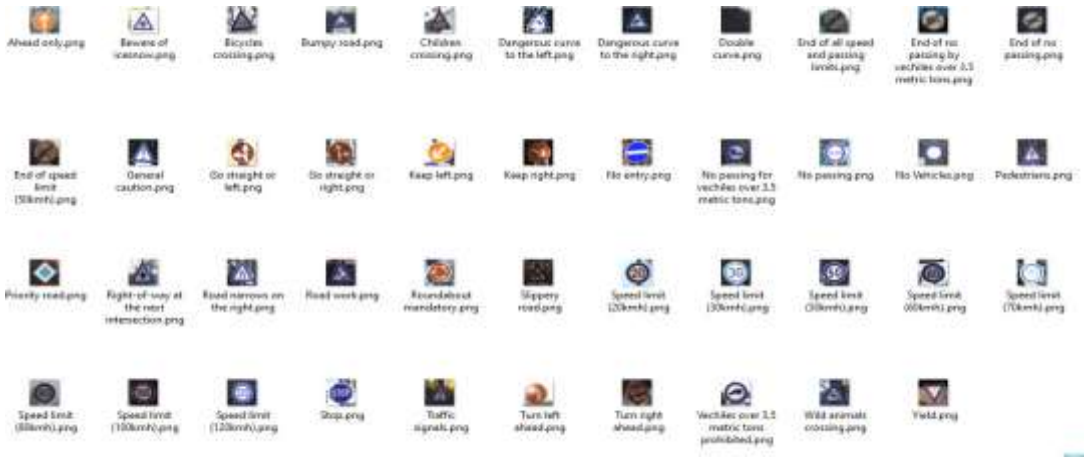


Fig 3 Sample for GTSRB DATA SET

Images should be resized to 32x32x3 pixels. This particular dataset has 43 classes detected, so it will first determine the number of classes it contains. Then it will import each class folder one at a time and combine them all into a matrix with the appropriate class ID. The data will then be split into training, testing, and validation, which were given the names x train, y train, x test, y test, x validation, and y validation, respectively. The number of photos utilized for training is about 22,000; for validation, it is about 5,500; and for testing, it is about 7,000. The training dataset distribution is displayed in Fig 4.

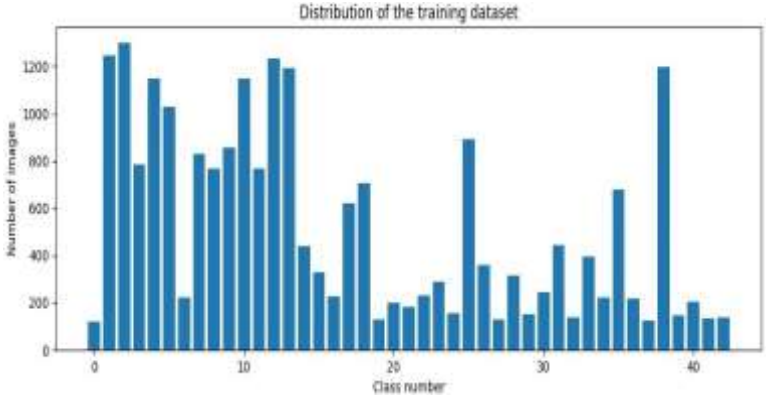


Fig 4 Distribution of Training dataset

4.2 Preprocessing:

The process of transforming raw data into a format that algorithms can understand clearly and analyze is known as data preprocessing, and it is a crucial step in both data analysis and machine learning. By iterating through the images, the preprocessing methods are applied to the training and testing datasets, respectively. The image is initially transformed into an array of pixels. The grey-scale technique receives the array of pixels as input. Grayscale images are created by converting RGB images to grayscale. The histogram equalization method is now applied to the transformed grayscale array as input. By distributing the intensity values that are most common. This technique enhances contrast in images. The pixel values are initially greater in some locations and lower in others. Histogram equalization increases the pixel range by distributing the values of the pixels equally. Using the cumulative distribution function, it transforms the grayscale distribution into the histogram equalization distribution. Following histogram equalization, the arrays of pixels are divided by 255 to produce pixels with values ranging from 0 to 1. A sample preprocessed image is shown in Fig. 5.

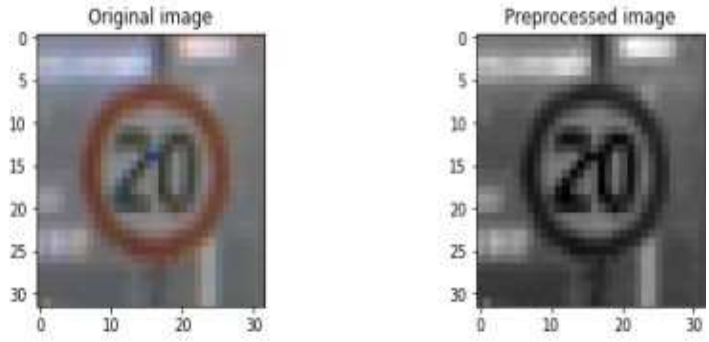


Fig 5 Preprocessing image

4.3 Data Augmentation:

By applying random modifications to the input data, such as rotation, scaling, and flipping, expanding the size of the training set is done using a technique called data augmentation. This reduces over fitting and enhances the model's capacity for generalization. It can generate additional images from a single image, giving the CNN model a large number of images to train on. With some variants, it turns a single image into many duplicates. For executing image augmentation, Keras offers a method called Image Data Generator. It has certain parameters (such as zoom, rotation, etc.), and different images are produced depending on these parameters. In Fig. 6, enhanced photos are displayed. Augmented images are shown in Fig. 6.

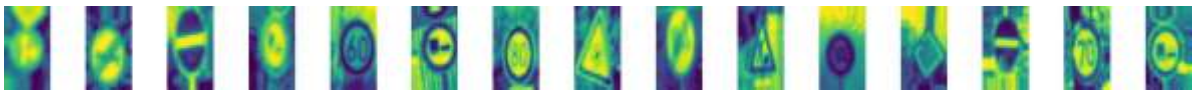


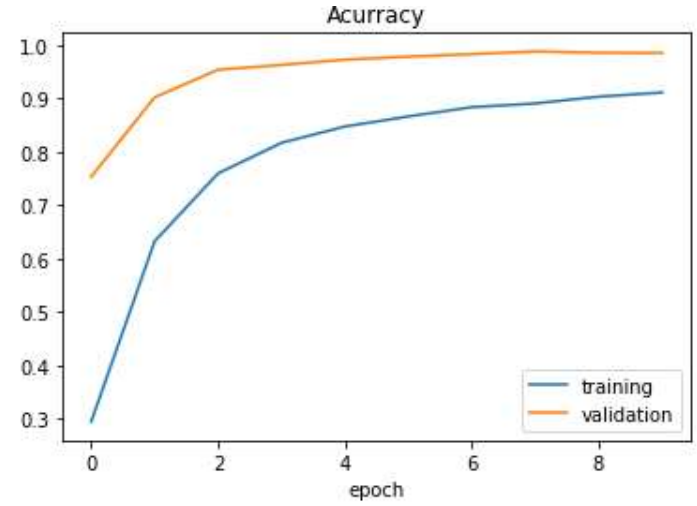
Fig-6: Augmented images

4.4 Building the CNN Model:

With the help of the Keras Sequential API, the CNN model is developed. The model has two convolutional layers with maximum pooling, two dense layers, and an output layer with an output size equal to the number of classes. The categorical cross-entropy loss function is used to compute the loss, and the model is built using Adam optimizer.

4.5 Training the Model:

Categorical cross entropy is used as the loss function during the model's training together with Adam optimizer. In order to produce a large number of identical images from a single image, data augmentation is employed while training the model. For the purpose of training the model, augmented images produced using the flow method of the image data generator classes are employed. Using the validation dataset, validation is done concurrently with the train. Using training and validation datasets at increasing epochs, graphs for loss and accuracy are shown after training. After training is finished, testing is performed using the testing dataset. Figure 7 shows the accuracy of the model and training validation data. When the learning rate on the Y-axis is set to 0.1 and the number of epochs on the X-axis is set to 10, Figure 8 shows the loss of the model for training and validation data. When the learning rate on the Y-axis is set to 0.5 and the number of epochs on the X-axis is set to 10.




```
Test Score: 0.05687808245420456
Test Accuracy: 0.9839080572128296
```

Fig 7 model accuracy

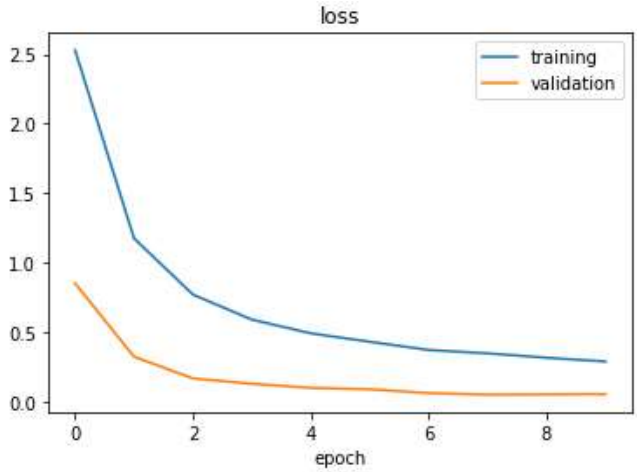


Fig 8 Loss occurred

4.6 Testing the model in real time:

The live camera is then set up using Open CV, which then accepts the image as input. Live webcam video might be used to test the model in this situation. Once access to the web camera has been gained, each frame of the live video is processed. Model that has been recorded to predict the probability that the processed image will. If the probability of the forecast is greater than the cutoff value of 0.75, the image is recognized as a traffic sign. The amount of the prediction probability and the matching class name are displayed together. The size of the training dataset and the design of the model both have an impact on the probability. If the dataset of linked class images is large, get a high probability prediction value for that.

5. Results:

Using a webcam, this model evaluated a real-time video. To search for traffic signs, it analyses each frame of the video. When a prediction probability value exceeds a threshold, the class label and probability are both displayed. In Fig. 8, a stop sign image is presented to the camera to illustrate how traffic signs can be recognized. Together with the prediction probability, the class name is displayed along with a voice alert. A real-time test is administered for each traffic sign across 43 classifications. Probability ratings for a few real-time images of traffic signs are displayed in Table 1. Some have probability scores that consistently exceed 90%, while others have numbers that fluctuate.

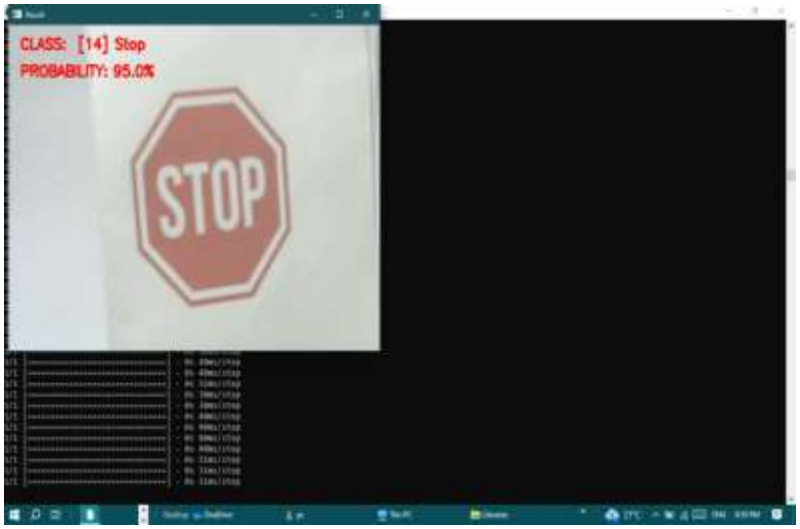


Fig-9 Traffic sign recognition real time using webcam

Table 1: Probability scores with traffic sign images in real time

S. No	Traffic Sign	Probability of recognition
1	yield	98%
2	No passing	98%
3	Ahead only	97%
4	No passing for vehicles over 3-5 metric tons	97%
5	Priority road	97%
6	Speed limit (30km/h)	96%
7	stop	95%
8	Road Work	92%
9	Go straight or right	92%
10	Speed limit (20km/h)	86%

6. Conclusion:

Voice alerts and traffic sign recognition help inexperienced drivers with the use of the Driver Assistance System, and in the case of self-driving cars, they offer safe, reliable navigation. The classification accuracy rate for images of traffic signs from the GTSRB dataset for the CNN model proposed in this thesis is 95%. The model is used to predict the class labels for such images and provide a voice alert when a traffic sign image is likely to be introduced to the web camera.

References:

[1] Ms. Nethravathi, Akshay K, Awati Sanman,"Traffic Sign Recognition System using Machine Learning", in Proc.IRJMETS,2022,pp.2582-5208.

[2]Pal R, Ghosh A, Kumar R, et al. Public health crisis of road traffic accidents in India: Risk factor assessment and recommendations on prevention on the behalf of the Academy of Family Physicians of India. J Family Med Prim Care.2019;8(3):775783.doi:10.4103/jfmpc.jf

[3]Wang C (2018): Deep learning-based research and applications for identifying and detecting traffic signs. International Conference on Robots &; Intelligent System

[4]J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark:a multi-class classification competition," in Proc. IEEE IJCNN, 2011, pp. 1453–1460

[5] Zhang J, Hui L, Lu J, Zhu Y (2018)Attention-based neural network for traffic sign detection. International Conference on Pattern Recognition.

[6] Valentyn Sichkar, Sergey A. Kolyubin, "Accuracy of classifying traffic signs is affected by different dimension convolutional layer filters".

[7]. D. Michie, D. J. & C. C. Taylor, "Machine learning, neural and statistical classification", (1994).

[8]functions to improve deep neural networks," arXiv preprint arXiv:1412.6830, 2014.

[9 P. V. Pachpute and P. M. Patil, "Real-time traffic sign recognition using deep convolutional neural networks," International Journal of Scientific Research in Computer Science, Engineering, and Information Technology, vol. 6, no. 1, pp. 15-22, 2020.

[10]. T. Nguyen and T. Nguyen, "Real-time traffic sign recognition using deep learning on Raspberry Pi," in Proceedings of the 2019 International Conference on Advanced Technologies for Communications (ATC), Hanoi, Vietnam, 2019, pp. 59-64.

[11]. K. B. Dinh and N. N. Hieu, "Real-time traffic sign detection and recognition using YOLOv3 and CNN," in Proceedings of the 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Hue, Vietnam, 2021, pp. 174-179.

[12]. B. Shi and X. Chen, "Real-time traffic sign recognition using transfer learning and deep neural networks," in Proceedings of the 2019 International Conference on Machine Learning and Data Engineering (iCMLDE), Changsha, China, 2019, pp. 17-21

[13]Yi Yang, Hengliang Luo, Huarong Xu and Fuchao Wu, Towards Real-Time Traffic Sign Detection and Classification 2014, IEEE

[14]J. Han, D. Zhang, G. Cheng, L. Guo, and J. Ren, "Using poorly supervised learning and high-level feature learning, object detection in optical remote sensing pictures," IEEE Transactions on Geoscience and Remote Sensing, vol. 53, no. 6, pp. 3325–3337, 2014

[15]Kai Li, Weiyao Lan, "Recognizing traffic signal indicators in light of context and shape". Department of Automation Xiamen University, China, 2011, IEEE

[16]S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolu- tional neural network," in 2017 International Conference on Engineering and Technology (ICET). Ieee, 2017, pp. 1–6.