Road Traffic Sign Detection & Recognition

Namitha Menon Kakkanat

AI Data Scientist , London , UK

Abstract

Traffic signs are indications that are placed along or above streets to notify drivers about the state of the roads and their destination. These signs are crucial to our safety while driving. They provide the drivers on the road with vital information. As a result, Drivers will need to gradually control their driving behavior and make sure they strictly adhere to the traffic laws that are in effect at the time without endangering other drivers or pedestrians. One of the helpful techniques for new drivers and autonomous cars is Traffic Sign Recognition (TSR). Automatic Traffic Sign Recognition is a feature of Advanced Driver Assistance Systems (ADAS). For safe navigation, ADAS and autonomous vehicles need to classify traffic signs effectively. This thesis proposes a Traffic Sign detection & recognition model to classify the road traffic signs present on the roads effectively and accurately. Our major concept is to categorise traffic signs using CNN in order to identify and detect traffic signs quickly and accurately. Convolutional Neural Networks (CNN) will be employed in the model as it has demonstrated promising performance in the image classification. The use of CNN, which is more effective and accurate than many deep neural networks, allows neural networks to be operated directly on images. Image processing is used to identify traffic signs on the roads. With the help of Tkinter, a GUI is created to detect and recognize the real-time road traffic signs and predict the signs using speech output. We classify the 43 kinds of traffic signs with remarkable accuracy using the GTSRB dataset for training and testing purposes. Final proposed model has achieved an accuracy of 94.15% with minimum loss of 21.20% without overfitting.

Keywords: Convolutional Neural Networks, Image Processing, Traffic Signs, Tkinter

1. Introduction

There are many various kinds of traffic signs, including those that indicate speed limits, no entry restrictions, traffic signals, left or right turns, school ahead, and restrictions on passing large vehicles, among others. Classifying traffic signs involves figuring out which category each one falls into. Traffic signs are divided into several types based on the information they transmit. Generally, there are 7 different types of traffic signs: Warning signs, priority signs, restrictions, mandatory signs, special regulatory signs, educational signs, directional signs, and other panels (Adam and Ioannidis, 2014).

Warning signs highlight potential hazards, obstacles, and road conditions. The information on the warning signs may reveal hazards that the driver might not

immediately notice concerning the condition of the road. Priority signs show the direction that a vehicle should travel, or the way it should pass another vehicle to avoid a collision. Stopping signs, intersection signs, one-way traffic signs, etc. fall under this category. Restrictive signs are used to forbid specific vehicles from driving on roads as well as to restrict specific maneuvers. No-pedestrian entry allowed, no-motorcycle, no-entry and other similar signs are included. Mandatory signs are the exact opposite of prohibitive signs that instruct drivers on what they must do. Special signs that indicate rules or give information about how to use something. Information about different possible destinations is provided by directional signs which includes turning to the right, left, etc.

There have been various attempts in recent years to identify traffic signs and alert drivers. Traffic Sign Recognition controls traffic signs and alerts drivers to any signs that are identified for safe and effective navigation. In the case of self-driving cars, an intelligent automatic road traffic sign detection in real-time can assist the driver or offer reliable navigation. By offering a quick method of detecting, classifying, and recognizing signs, automatic traffic sign detection and recognition can significantly help achieve this goal. Once this is completed, it will be simpler for humans to spot distorted or hidden signals.

Even though the computer vision research community and business have made great advancements in recognising and detecting traffic signs the past few decades, there are still two fundamental problems that are unavoidable. One is the low resolution, motion blur, and noise-induced poor image quality. The other is uncontrollable external elements, such as bad weather, a complicated background, multiple traffic symbols blocking each other, varying lighting, fading traffic sign colors, etc. Due to them, the process of recognising and detecting traffic signs is still problematic. Fig. 1 shows some sample images of these problematic traffic signs located on the road.

To address these issues, this thesis proposes a traffic sign identification process that consists of two parts. Traffic sign detection is the first step, which involves finding and gathering size information about traffic signs from photographs of road scenes. In this stage, image processing is used to identify the traffic sign in a picture. The second stage, known as traffic sign classification, entails classifying detected traffic signs into the appropriate sub-classes. This identified image is classified in the second stage using a model of an artificial neural network known as Convolutional Neural Network model.

Fig.1 Challenging traffic signs images for detection.

2. Background

The ability to recognize traffic and road signs has been the subject of extensive investigation. Numerous strategies and methods have been suggested in attempt to overcome these barriers. Three kinds of computer vision techniques for detecting traffic signs can be made: deep learning, classical machine learning, and colour and shape-based techniques (Hechri and Mtibaa, 2020). In Swaminathan et al. (2019), the thesis was based on Belgium Traffic sign dataset. Here the authors aim to create a design that uses image classification to capture the traffic sign and classified using a Convolutional neural network and the output is given through Arduino controlled car. The accuracy that got in this research was 83.75% by using the CNN approach. For classification, Safat et al. (2015) utilised the K-means technique along with kernel-based supervised SVM. They made use of their own information that was recorded by the camera on the Malaysian highways. Image preprocessing, detection, and recognition are the system's three active development phases. Support vector machine (SVM) classifier followed by form matching and RGB color segmentation was used in the system demonstration, and it produced encouraging results with an accuracy of 95.71% (Wali, Hannan, Hussain and Samad, 2015).

Qin and Yan (2021) the identification of traffic signs in New Zealand is measured against a benchmark. The Faster R-CNN and YOLOv5 models are two types of deep learning models that can be used to test which deep learning model is best for the TSR problem. The accuracy rate is slightly lower with YOLOv5 than with the Faster R-CNN, but it is still more sufficient and significant (Qin and Yan, 2021). The thesis by Sun et al. (2019) presents a deep learning-based traffic sign recognition method that focuses primarily on the identification and categorization of circular traffic signs. To emphasise important details, an image is initially pre-processed. Hough line transform is also employed for area detection and localization. At last, the identified road traffic signs are classified using deep learning and thus the model able to provide the detection of circular symbol with more than 98.24% accuracy (Sun, Ge and Liu, 2019).

Another study by Chiu et al. (2019) was on the YOLOv3 framework-based traffic sign detection and recognition technique. Then evaluate using various image resolutions. Image datasets of Taiwanese road scenes are used to train the networks (Chiu, Lin and Tai, 2019). Hechri and Mtibaa (2020) uses a two-step technique in traffic sign detection. In the first step, they employ Linear SVM on the HOG feature to classify signs into triangular & circular shapes. In the second step, these images are fed into the CNN classifier. The dataset used was based on the Chinese Traffic sign dataset. In the recognition step, a technique called joint tranceform correlation (JTC), which is often used for pattern matching, is used to

identify the detected traffic symbol. The accuracy of the model was 95.37% (Hechri and Mtibaa, 2020).

Authors William et al. (2022) has taken an approach to detect and recognize the traffic sign in real-time with respect to the weather, illumination, and visibility challenges. These challenges are met through transfer learning. The model is created by Faster Recurrent Convolutional Neural Networks (F-RCNN) and Single Shot MultiBox Detector (SSD). The data set used the German Traffic sign dataset which contains 39,200 images. The accuracy of the model was 99.81% by using the F-RCNN approach (William et al., 2022).

Several methods for color-based road sign detection are used because red, yellow, and blue are the most common colors used on road signs. Deshmukh et al. (2013) uses the RGB space to separate the road sign sections from their background by using the color thresholding technique. However, the RGB color space is extremely sensitive to a variety of factors, such as auto lighting or influences from natural illumination. Numerous additional color spaces are employed to solve this issue (R.Deshmukh, K. Patnaik and E. Patil, 2013). Park and Kim (2013) also uses techniques with advanced SVM to improve the processing time and precision of grayscale sign images.

3. Methodology

The goal of this research is to identify and recognize traffic signs for autonomous vehicles in real-time. We will use the Deep Learning, multi-layer Convolutional Neural Network, and Keras libraries to design a model that categorizes the traffic signs in the dataset into different groups. Later with the help of GUI (Graphical User Interface) model will be used to detect and recognize the uploaded images and predict the symbol using speech output.

3.1. Ethics

The study was conducted according to the guidelines of the Declaration of Helsinki and approved by the Research Ethics Committee of The University, United Kingdom.

3.2. Objectives

- The first objective is data augmentation to balance disproportionate images count in each class and split the dataset for test and train the model.
- To build cost effective Convolutional Neural Network model using modified LeNet model and hyperparameter tunning to achieve best accuracy for the model prediction.

- To build the GUI that can use the trained model to predict the uploaded traffic sign through speech output.

3.3. Data Collection

The dataset used for the thesis is from public dataset that is available in Kaggle. The dataset is based on German traffic signs that contains more than 50,000 images of different traffic signs available on the road. There are 39,209 training images and 12,630 test images (Fig.2) in the German Traffic Sign Recognition Benchmark (GTSRB) dataset, which consists of 43 classes of traffic signs (Vagadia, 2022). Some of the images are taken under varying light conditions such as blurred, poor visibility, deformed to train the model under different circumstances (Fig.2). 3-channel RGB pictures are used throughout (Vagadia, 2022).

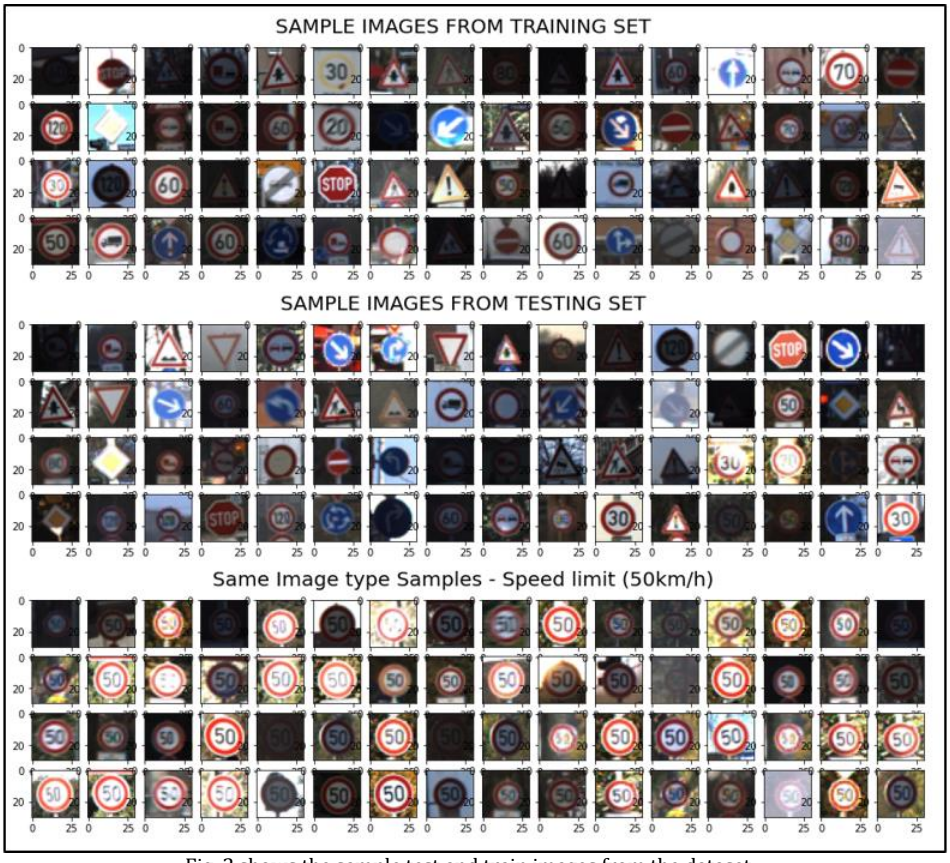


Fig. 2 shows the sample test and train images from the dataset.

Some of the challenges that came up while using the dataset are:

- Uneven distribution of the target classes.

- The amount of traffic sign images in each class is disproportionate. Approximately 2500 sign images are used in certain classes, whereas just 250 sign images are used in others (Fig. 3).
- A lower quantity of images can make training less effective. To fix this issue Data Augmentation technique can be used.

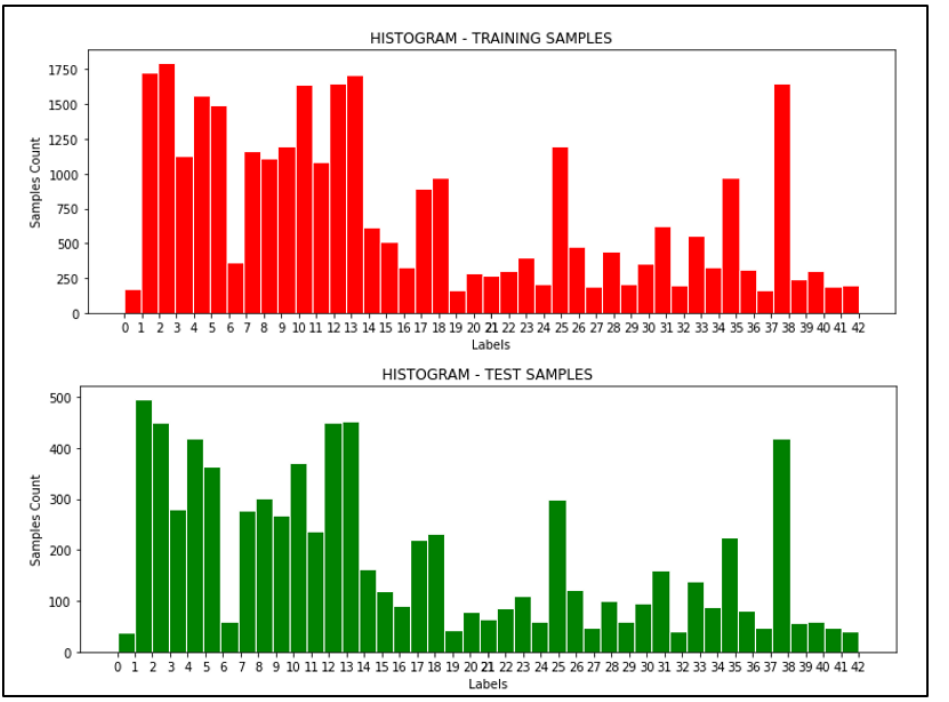


Fig.3 shows class imbalance due to different image counts.

3.4. Architectural Flow

Our thesis used the experimentation approach to accomplish its objectives. The experimentation has two stages (Fig. 4).

Stage 1: Comprises model building, testing & training the model with default parameters and finally hyperparameter optimization to improve the accuracy of the model.

Stage 2: Comprises GUI creation and testing the previously trained model through GUI and predicts using speech output.

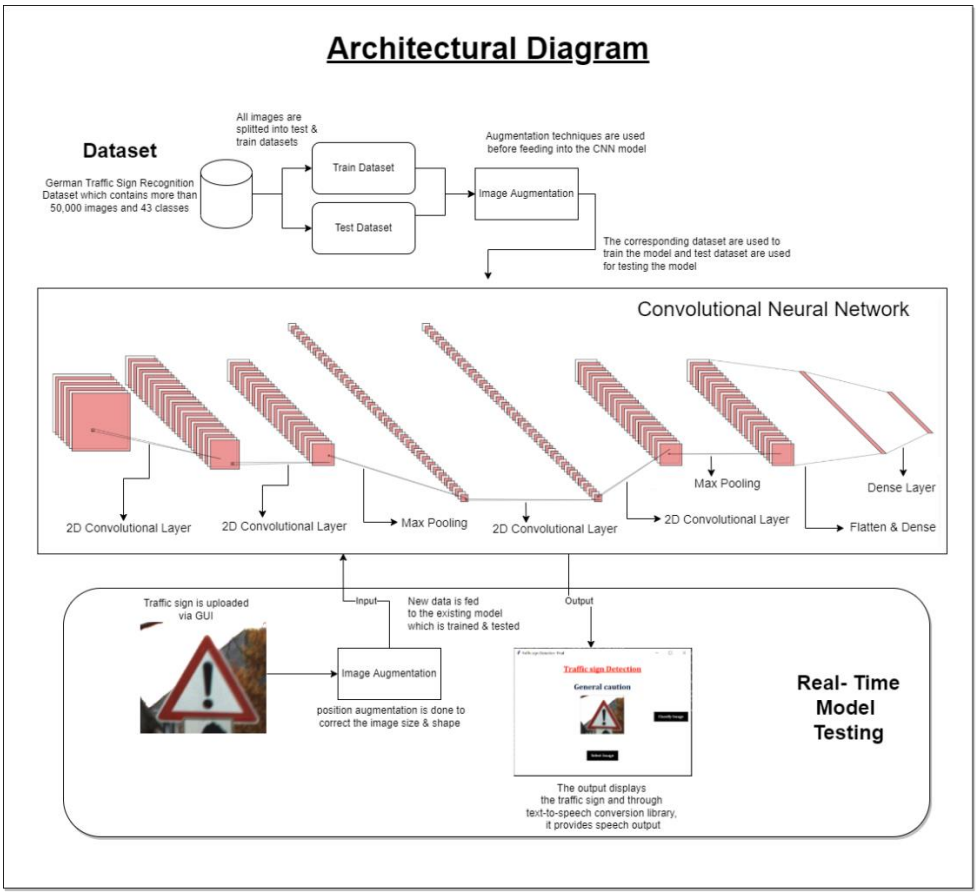


Fig. 4 overall architectural diagram that involves Stage 1 and Stage 2.

3.5. Data Pre-processing

Data pre-processing is an essential part of image classification. It concentrates on fixing flaws that could interrupt the learning process like omissions, noise, and outliers (García, Luengo and Herrera, 2015). The major goals of image preprocessing, which is a crucial component of the TSDR system, are to eliminate reflections, normalize the intensity of the individual particle images, remove low-frequency background noise, and mask certain areas of the images. Since the training dataset's image sizes varied, it was necessary to resize the images in the dataset before feeding them into the model. PIL is a library that provides a number of common techniques for modifying images. Then, using the OS module, it goes over each class, adding images and their labels to the data & labels list.

The resultant data's shape is (31367, 30, 30, 3), which indicates that there are 31,367 images, each measuring 30 pixels by 30 pixels, and that the data also includes colored images (3 stands for RGB value). Images are then splitted into testing and training data.

3.6. Building CNN Model

Convolutional Neural Network model is built based on the LeNet model. Some modifications were given to the lenet model. LeNet-5 CNN architecture has seven layers. The layer composition comprises of three convolutional layers, two subsampling layers, and two fully linked layers (Saxena, 2021). As a modification to the model, one more Conv2D layer along with two dropout are added to the lenet model. So as final lenet architecture contains Four Conv2D layers, two maxpool2D layers, two drop out layers, two fully connected layers and one flatten method (Fig. 5). In the first and second convolution layers, using ReLU activation function, 32 kernels of size 5*5 were utilised. For third and fourth layer, using ReLU activation function, 64 kernels of size 3*3 are used. For maxpool2D, 2*2 pool size are used. Three dropout layers are used with 0.25% and 0.5% dropout i.e.., by zeroing out the neuron values, 25% or 50% of the neurons will shut down at each training phase. Two dense layers performs ReLU and softmax activation functions respectively.

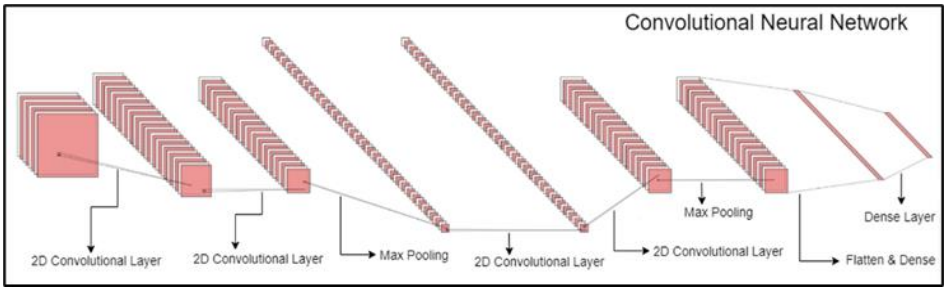


Fig. 5 Proposed Convolutional Neural Network.

- The array of pre-processed 3D images is multiplied element by element in the 2D convolution using 5x5 sized kernel, which then added up the array and replace the sum in the correct position. The picture array is updated for each pixel value in the same manner. In this layer, 60 filters are used. The size changes to 28x28 after this layer. Later, the ReLU layer activates all an output image's pixel values using a function max (0, x).
- The one layer of Conv2D is subjected to the same operation with the ReLU activation function. The size will be scaled down to 24x24.
- The largest element from the region of the feature map covered by the filter is obtained by the max-pooling method. The main purpose of the max pooling is to obtain the most salient features from the prior feature map. Specifically, 2x2 filter size is given.
- After maximum pooling, two Conv2D layers using ReLU activation function, 64 filters of size 3x3 and were applied one after the other.

- Max pooling was applied once more using a 2x2 filter size. The connections are then cut in half using a 0.5 drop off. After drop out, the picture array is flattened, which reduces it to a single dimension.
- The final output array is obtained by combining two completely connected layers-one with activation function as ReLU and the another using activation function as softmax and 0.5 of dropout.

Finally, the model was then compiled with the Adam optimizer, along with accuracy metric, and loss.

3.7. Train & Test CNN Model

Compilation of the model is done using adam optimizer and loss is categorical_crossentropy since there are many classifications that need to be categorized. The Adam optimizer is easier to develop than any other optimization technique, that has a faster running time, uses less memory, and needs less tuning (Gupta, 2022). In some classes, there aren't many images (Fig.4), so data augmentation is utilised to create several duplicate images from a single image while training the model. The model is then trained with following parameters:

- Train data
- Batch size
- Number of epochs
- Validation is carried out concurrently with training using the validation dataset.

Following training, graphs for loss and accuracy are plotted using increasing epochs across training and validation datasets. Testing is carried out using the testing dataset following the completion of training. These are the unseen traffic sign that are not used in training. The model is then saved using the save function with the ".h5" extension.

Hyperparameter tuning is done to improve the accuracy of the trained model. One of the tunings that provides significant improvement in the accuracy of the model were when the batch size changed from 32 to 64. The model performed well when the batch size was 64. Also, after 15 epochs the model accuracy was found to be in stable state.

3.8. Model Testing with Test Dataset

After the model training, using the test dataset. The image path and its labels are extracted using pandas library. For test prediction, image resizing needs to be done into 30x30 px. After which all the images are put into numpy array to feed into trained model. The accuracy score of predicting the labels is then obtained.

3.9. GUI Implementation for Traffic Sign Detection

For traffic sign classification in real time, graphical user interface is created to upload the images for the prediction in text and speech output simultaneously. For the GUI implementation, tkinter interface is used. Appropriate labels and buttons are provided to easier understanding and usability. The classify function transforms the image size into the (1, 30, 30, 3) shape's dimension, to fit into the trained model. Prediction will be provided by returning number between 0-42. This predicted number represents the class it belongs to. Using text to speech conversion library(pyttsx3) the output is provided in speech format.

4. Results

The results section comprises of two sections. The section 1 involves results during the evaluation of the model before and after the hyper parameter tuning. The section 2 involves results during the prediction and classification of traffic sign through GUI classifier.

4.1. Section 1: Model Training and Tuning

A CNN model using a modified LeNet model is built in this thesis using Four Conv2D layers, two maxpool2D layers, two drop out layers, two fully connected layers, two dense layers and one flatten method (Fig. 6). Data is split during the training phase, with 70% used as training data and rest 30% used as testing data.

The model achieved an accuracy of 94.15% during training and 93.33% in testing phase without overfitting.

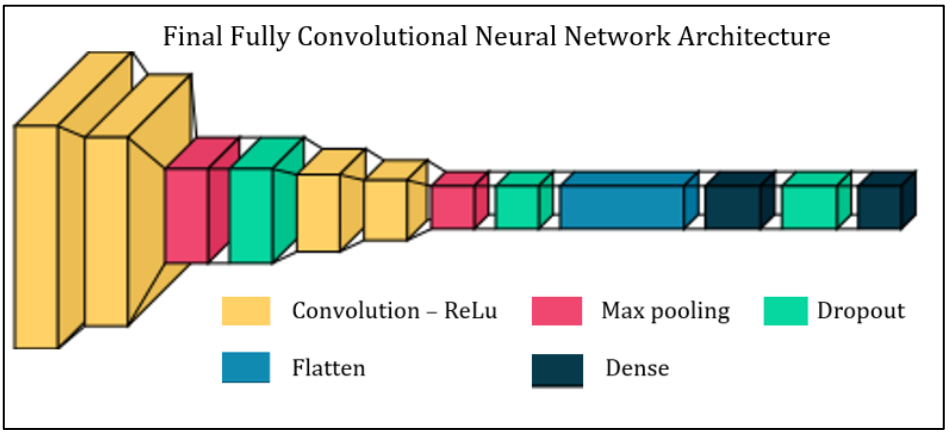


Fig. 6 shows final architecture of CNN.

4.1.1. Model training with default parameters

The model runs with the configured default parameter values, which are 15 epochs and 0.001 for the learning rate. Fig. 7 shows the model's accuracy for train and test data and its loss for those same data, respectively. Here blue line shows the training data and orange line shows the testing data. The model has achieved accuracy of 95%.

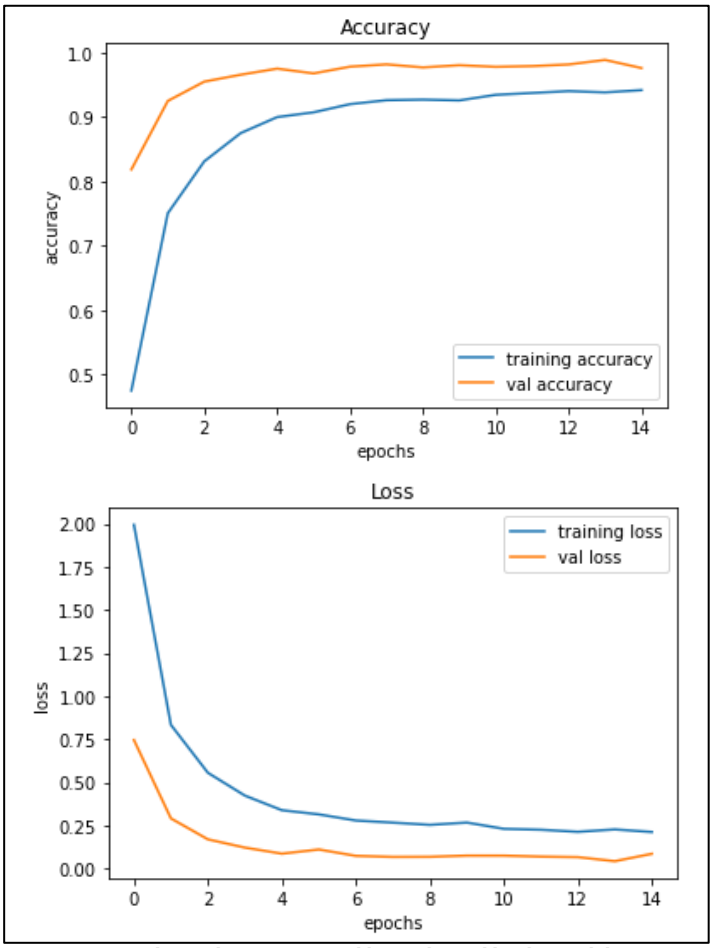
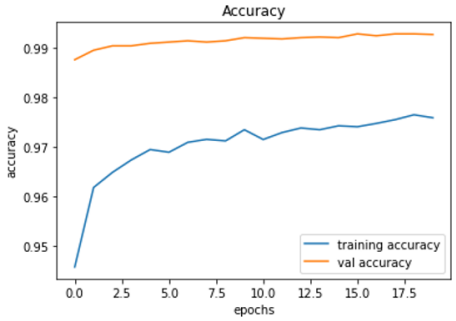


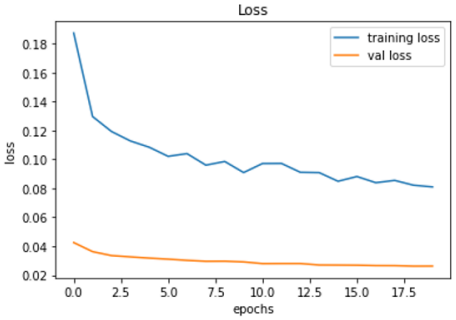
Fig.7 Shows the accuracy and loss achieved by the model.

4.1.2. Model training after parameter tuning

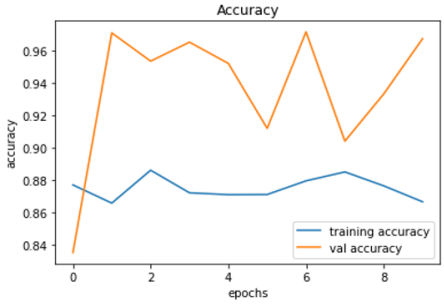
Epochs and learning rate are the two main hyperparameters that are tuned. Each hyperparameter tuning demonstrates differences in the model's accuracy and loss. The accuracy and loss of the model during hyperparameter optimization are shown in the graphs below (Fig. 8). Here blue line shows the training data and orange line shows the testing data.

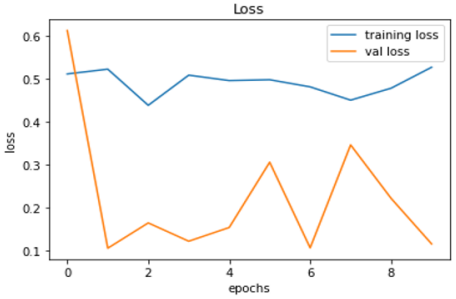
For learning rate: 0.00001 and epoch: 20



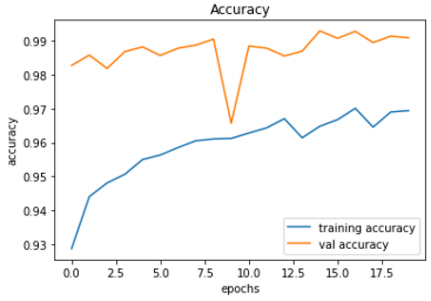


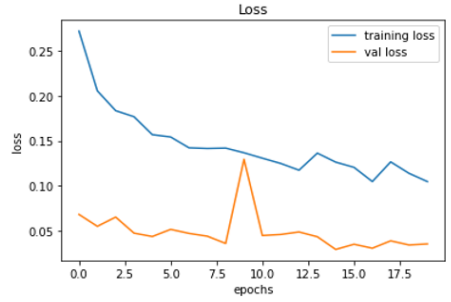
For learning rate: 0.002 and epochs: 10



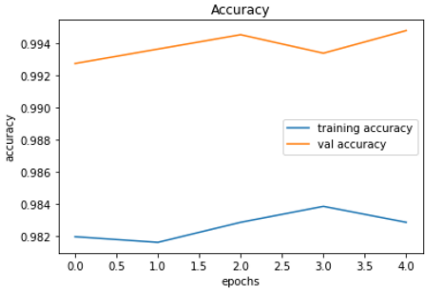


For learning rate: 0.0005 & epochs: 20





For learning rate: 0.0001 & epochs: 5



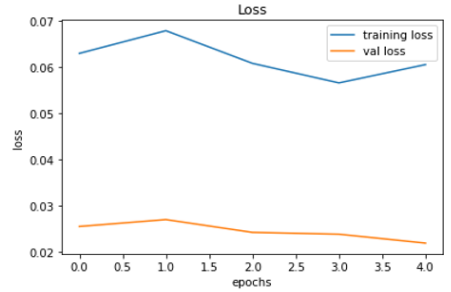


Fig.8 The accuracy and loss achieved by the model on different parameter tuning.

4.2. Section 2: Using Model in GUI

In this section, the trained model (Fig. 7) with good accuracy and minimal loss will be used in the GUI to classify the uploaded traffic signs and predicts the output through text and speech. Uploaded traffic sign images are new images that are not used while training or testing data. For understanding the accuracy, one images from each class including blurred or poor visibility images are tested through GUI. Buttons and labels are provided in the GUI for easy human use. Classify button is provided to predict the output via text and speech. Fig. 9 shows the output predicted on different type of traffic signs.

Fig. 9 The predicted results by model in GUI.

5. Discussion

To detect and recognize the traffic signs using the GTSRB dataset, we have suggested a CNN model whose architecture is based on the LeNet model. By deriving specific conclusions from the findings, the questions are addressed below.

Question 1: *How much can Hyperparameter optimization increase the model's accuracy?*

The Fig. 8 shows the accuracy and loss for the test and train data during different combination of parameter tuning. Mainly epochs and learning rate are tuned in different combinations to understand the changes in the accuracy and loss values. In every combination of parameter tuning, the model revealed a noticeable difference in the accuracy (Table 1).

#	Epochs	Learning Rate	Train Accuracy	Train Loss	Test Accuracy	Test Loss
1	15	0.001	0.9415	0.212	0.9758	0.0854
2	20	0.00001	0.9759	0.081	0.9927	0.0262
3	10	0.002	0.8665	0.5264	0.9674	0.1164
4	20	0.0005	0.9694	0.1046	0.0352	0.9909
5	5	0.0001	0.9829	0.0606	0.9948	0.022
6	30	0.0001	0.9877	0.0432	0.9941	0.0221

Table 1: Accuracy & Loss for train & test during hyperparameter tunning.

When the learning rate is adjusted, certain intriguing patterns in the model's accuracy are found. The learning rate is a hyperparameter that regulates how much to alter the model in response to the predicted error each time the model weights are updated (Brownlee, 2019). The learning rate is often an adjustable hyperparameter used to train the model, with a range of 0.0 to 1.0. The accuracy of the model is decreased when the learning rate is changed from the optimal default value of 0.001. Even though a high learning rate accelerates model learning, it also produces weights that are not ideal (Brownlee, 2019). Also, it may take a long time for the model to acquire an ideal set of weights with a lower learning rate (Brownlee, 2019). Therefore, increasing the learning rate above the default value reduces model accuracy.

On the other hand, Table 1 shows unequivocally, that a rise in the number of epochs leads to high accuracy. But too much increase in the epochs can lead to overfitting. The model rapidly learns the patterns of the sample data when the number of epochs used for training is increased (Baeldung, 2022). This make model is less effective on test data as a result. The model performs well on train

data due to overfitting but falls short on test data. Underfitting happens when there are not enough epochs which also results in low accuracy (Baeldung, 2022). Therefore, increasing the epochs considerably increases the model's accuracy up to a point. 95% Model accuracy is attained when the optimum learning rate and epochs are both set to 0.001 and 15 respectively.

Question 2: *How do the results compare with the wider literature?*

Comparing with the other (Swaminathan et al., 2019; Hechri and Mtibaa, 2020; William et al., 2022), our proposed system performs very well in terms of accuracy cost effective and loss. Swaminathan, et al. (2019) proposed a CNN model based on Belgium Traffic sign dataset which is costly in implementation and accuracy found to be 83.7%. Hechri and Mtibaa (2020) proposed system has fewer images for training but uses Faster Recurrent Convolutional Neural Networks and achieved 99.8% with some overfitting. William et al. (2022), uses Linear SVM on the HOG feature to classify signs into triangular & circular shapes. The dataset used was based on the Chinese Traffic sign dataset. The accuracy of the model was 95.37%.

Based on the literature, our system performed well in terms of accuracy, loss and implementation cost. The developed system's key drawback is that it can only be used for image classification and not for video input. Traffic sign detection and recognition in live video can be suggested as future work.

6. Conclusion

Through the Driver Assistance System, traffic sign recognition assists new drivers, and in the event of self-driving cars, it provides safe, effective navigation. The goal of this thesis is to accurately identify and recognise traffic signs, which will increase the potential of autonomous vehicles. In order to achieve the goal, we have developed an efficient & cost-effective Traffic sign detection & recognition system based on German traffic sign database from Kaggle. The sign images are uploaded in real-time through GUI. The traffic sign recognition process is achieved by Convolutional Neural Network using LeNet architecture. The developed system shows promising result in terms of 94.15% accuracy and minimal loss of 21%. The model performance is evaluated by plotting the graphs for accuracy and loss. The effectiveness of the implementation is demonstrated by a comparison of the simulation results with the current approaches.

We can infer from the results that change in number of epochs has a noticeable impact on the accuracy of the model. The outcomes demonstrate that altering the learning rate causes the accuracy of the model to going down. Consequently, the model has a high accuracy of 94.15% with learning rate = 0.001 and epoch = 15.

In future, additional parameters, like rate of dropout, batch size etc., can be adjusted to further improve the accuracy of the model. Also, the model needs to look for a larger dataset to train existing system to understand lesser-known road signs used in other countries. A larger dataset may yield better predictions since it allows for the collection of more information for each class label. The system's capacity to recognise on live input video is another thing that might be enhanced in future. For advancement, the model will be trained to predict signs on live input at high speeds and increase maximum distance of recognition to recognise the traffic signs earlier. So that with the help of a voice prompter as the output, this model can be used with the ADS system.

References

[1] Adam, A. and Ioannidis, C., 2014. Automatic road sign detection and classification based on support vector machines and HOG descriptors. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-5, pp.1-7.

[2] Aoyagi, Y. and Asakura, T., 2006. A study on traffic sign recognition in scene image using genetic algorithms and neural networks. *Proceedings of the 2006 IEEE IECON. 22nd International Conference on Industrial Electronics, Control, and Instrumentation*, (3), pp.1838-1843.

[3] Baeldung, 2022. *Epoch In Neural Networks*. [online] Available at: <https://www.baeldung.com/cs/epoch-neural-networks> [Accessed 16 August 2022].

[4] Brownlee, J., 2019. *Understand The Impact of Learning Rate on Neural Network Performance*. Available at: <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/> [Accessed 17 August 2022].

[5] Chiu, Y., Lin, H. and Tai, W., 2019. Implementation and Evaluation of CNN Based Traffic Sign Detection with Different Resolutions. *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pp.1-2.

[6] Escalera, A., Armingol, J. and Salichs, M., 2001. Traffic sign detection for driver support system.

[7] García, S., Luengo, J. and Herrera, F., 2015. Data Pre-processing in Data Mining. 1st ed. Springer Cham.

[8] Garrido, G., Ocaña, M., Llorca, D., Sotelo, M., Arroyo, E. and Llamazares, A., 2011. Robust traffic signs detection by means of vision and V2I communication. *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems (ITSC '11)*, pp.1003–1008.

[9] Gupta, A., 2022. A Comprehensive Guide on Deep Learning Optimizers. [online] Analytics Vidhya. Available at:<https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/> [Accessed 17 August 2022].

[10] Greenhalgh, J. and Mirmehdi, M., 2012. Traffic sign recognition using MSER and Random Forests. *Proceedings of the 20th European Signal Processing Conference (EUSIPCO '12)*, pp.1935–1939.

[11] Hannan, M., Wali, S., Pin, T., Hussain, A. and Samad, S., 2014. "Traffic sign recognition based on neural network for advance driver assistance system. *Przegląd Elektrotechniczny*, 1(12), pp.169–172.

[12] Hechri, A. and Mtibaa, A., 2020. Two-stage traffic sign detection and recognition based on SVM and convolutional neural networks. *IET Image Processing*, 14(5), pp.939-946.

[13] Li, Y., Pankanti, S. and Guan, W., 2010. Real-time traffic sign detection: an evaluation study. *2010 International Conference on Pattern Recognition*, pp.3033 - 3036.

[14] Ohara, H., Nishikawa, I., Miki, S. and Yabuki, N., 2002. Detection and recognition of road signs using simple layered neural networks. *Proceedings of the 9th International Conference on Neural Information Processing*, 2(ICONIP '02), pp.626-630.

[15] Park, J. and Kim, K., 2013. Design of a visual perception model with edge-adaptive Gabor filter and support vector machine for traffic sign detection. *Expert Systems with Applications*, 40(9), pp.3679-3687.

[16] Qin, Z. and Yan, W., 2021. *Traffic-Sign Recognition Using Deep Learning*. Auckland, New Zealand: First International Symposium, pp.13-25.

[17] R.Deshmukh, V., K. Patnaik, G. and E. Patil, M., 2013. Real-Time Traffic Sign Recognition System based on Colour Image Segmentation. *International Journal of Computer Applications*, 83(3), pp.30-35.

[18] Saxena, S., 2021. Lenet-5 | Lenet-5 Architecture | Introduction to Lenet-5. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/03/the-architecture-of-lenet-5/> [Accessed 21 August 2022].

[19] Sun, Y., Ge, P. and Liu, D., 2019. Traffic Sign Detection and Recognition Based on Convolutional Neural Network. *2019 Chinese Automation Congress (CAC)*, pp.2851-2854.

[20] Swaminathan, V., Arora, S., Bansal, R. and Rajalakshmi, R., 2019. Autonomous Driving System with Road Sign Recognition using Convolutional Neural Networks. *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, pp.1-4.

[21] Vagadia, H., 2022. German Traffic Sign Recognition Benchmark. [online] Medium. Available at: <https://medium.com/analytics-vidhya/german-traffic-sign-recognition-benchmark-5477ca13daa0> [Accessed 17 August 2022].

[22] Wali, S., Hannan, M., Hussain, A. and Samad, S., 2015. An Automatic Traffic Sign Detection and Recognition System Based on Colour Segmentation, Shape Matching, and SVM. *Mathematical Problems in Engineering*, 2015, pp.1-11.

[23] William, M., Zaki, P., Soliman, B., Alexsan, K., Mansour, M., El-Moursy, M. and Khalil, K., 2022. Traffic Signs Detection and Recognition System using Deep Learning. *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pp.160-166.

[24]