A picture containing text

Description automatically generated

Table

Description automatically generated

Abstract

Everyday roadside traffic signs are vital to our safety on the roadways. They provide crucial information to drivers. This, in turn, mandates that they regulate their driving behaviours and strictly comply to the traffic laws as they are now enforced so as not to create difficulties for other vehicles or pedestrians. The objective of traffic sign detection and categorization is to notify and prepare drivers to comply with traffic rules. The proposed solution tackles a number of concerns with previously used categorization algorithms, including erroneous predictions, costly hardware, and frequent maintenance. In the suggested method, a convolutional neural network is employed to develop an algorithm for classifying traffic signs.  The result “on the German traffic sign identification benchmark dataset shows this” model achieves a high accuracy rate.

Keywords: Deep Learning, CNN, Traffic sign classification and recognition

List of figures

Figure 1: Flowchart of methodology of this research

Figure 2: Warning type traffic signs

Figure 3: Regulatory type traffic signs

Figure 4: Direction type traffic signs

Figure 5: Sample of traffic signs

Figure 6: Architecture of CNN model

Figure 7: Input images are processed by a neural network, with each layer identifying and highlighting unique characteristics.

Figure 8: The top layer of the CNN classifies the image based on the features collected by convolutional layers.

Figure 9: Confusion matrix

Figure 10: Code snippet of loading dataset

Figure 11: Code snippet of splitting the data

Figure 12: Code snippet of determine y\_train and y\_test

Figure 13: Result of dataset

Figure 14: Code snippet of Model

Figure 15: Code snippet of plot of model

Figure 16: plot of model

Figure 17: Summary of mode

Figure 18: Code snippet of training model

Figure 19: Result of model

Figure 20: Code snippet of summary of model

Figure 21: result of summary

Figure 22: Code snippet of getting plot of accuracy and loss

Figure 23: Plot of accuracy

Figure 24: Plot of loss

List of tables

Table 1: The existing CNN model on traffic signs datasets

Table 2: The existing traffic signs datasets

Table of Contents

[Introduction 5](#_Toc120066237)

[Problem statement 7](#_Toc120066238)

[Research Aim and Objectives 7](#_Toc120066239)

[Literature review 7](#_Toc120066240)

[Introduction 7](#_Toc120066241)

[Related Works 8](#_Toc120066242)

[Methodology 11](#_Toc120066243)

[Dataset 12](#_Toc120066244)

[Conventional Neural Network method 14](#_Toc120066245)

[Evaluation metric 17](#_Toc120066246)

[Data mining tool 17](#_Toc120066247)

[Implementation 17](#_Toc120066248)

[Pre-processing 18](#_Toc120066249)

[Model Building 19](#_Toc120066250)

[Model Visualization: 21](#_Toc120066251)

[Accuracy Plot 24](#_Toc120066252)

[Results 24](#_Toc120066253)

[Future research 25](#_Toc120066254)

[Conclusion 25](#_Toc120066255)

[References 26](#_Toc120066256)

# Introduction

The development of AI in the present decade has made it possible for a change in how people regard AI's ability to enhance human life. This method is effective for automatically detecting and recognising traffic signs from highway scenes (Li & Wang, 2019). Drivers must pay attention to and observe all posted traffic signs in order to ensure the safety of everyone on the road. As image processing has evolved, the potential benefits of a traffic sign detection and identification system have boosted interest in this field. Autonomous cars are now a trendy issue, which has heightened people's interest in this field of research. Automatic traffic sign detection and identification technology allows for the development of intelligent vehicles and driving. This technology has the potential to minimise the number of accidents caused by human error, however a human driver is still necessary to operate a vehicle. The incorporation of such a technology into vehicles will unquestionably decrease the number of accidents on the road, hence saving lives and reducing the monetary impact of accidents on society. Eventually, both highway and city street traffic may be handled by computerised systems(Alghmgham et al., 2019).

Computer vision technologies have been used to identify and classify traffic signs, but doing so manually requires a great deal of time-consuming, arduous work creating important aspects in images. Shape- and colour-based techniques are also feasible options, but they suffer from the same shortcomings when the scene's lighting or colour scheme is adjusted, or when the scene's size, rotation, or translation is altered. However, machine learning may also be utilised to address comparable problems, although a substantial amount of annotated data is required. In recent years, deep learning's exceptional performance of classification and its potential for representational learning from data have attracted broad attention, making it the more effective alternative(Radu et al., 2020).

In addition, deep “neural networks have been used for the identification and recognition of generic objects with “better results than earlier techniques. Because of this, the autonomous vehicle industry took note. In this work, researcher will examine how the Convolutional Neural Networks (CNN) model has been used to the detection of traffic signs. A CNN” typically consists of two major components, both of which are multilayer neural network models.

To organise the remaining research, researcher shall proceed as follows. The second section contains a literature review and any relevant earlier publications, while the third section outlines the proposed methodology. The implementation part and results are presented in section 4 and 5. Section 6 and 7 are related to limitation and future work. Section 8 concludes the study.

## Problem statement

Today, automation has almost simplified every area of life. Drivers who maintain constant focus on the road are more likely to disregard roadside signs that might avert accidents and save lives. This problem would not exist if there were a reliable technique of informing drivers without diverting their attention. Traffic Sign Detection and Recognition (TSDR) plays a significant function here by detecting and recognising a sign, therefore notifying the driver of any future signs, since it offers vital information for road users. This guarantees that drivers modify their behaviour to comply with the traffic laws imposed on the road sector. In addition to guaranteeing the safety of everyone on the road, this also calms the driver while traversing unknown area. The inability to comprehend the sign's meaning is an additional challenge often encountered by individuals. Due to its ADAS software, motorists will no longer need to be concerned with not being able to see the sign.

## Research Aim and Objectives

The aim of this study is to design a deep learning network of CNN model to recognition the traffic sings dataset.

The following are the objectives of the study:

1. To evaluate the CNN model and parameters optimization
2. To optimize the model and get the accuracy plot

# Literature review

## Introduction

Researchers have spent the last decade attempting to determine how to recognise traffic signs. Researchers have tried a variety of techniques in an attempt to increase recognition accuracy and solve the problem. CNN classifiers are used by most contemporary approaches. In the past, traffic sign recognition included collecting attributes from images of the signs and sending them to a classifier. These approaches are not as precise as humans. CNNs have significantly improved human classification accuracy via the application of deep learning techniques.

## Related Works

The authors of (Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, 2016), produced a new dataset consisting of one hundred thousand photographs and described a method for identifying and categorising traffic signs using a potent end-to-end CNN on this dataset. 84% of the time, the strategy was correct. The authors of (Hatolkar et al., 2018) examined the techniques presently used by CNN-based traffic sign recognition systems. In their study, they described not just the limitations inherent to CNN techniques in terms of time complexity and accuracy, but also a number of suggested solutions for overcoming these hurdles. They also proposed a system that made use of sophisticated edge detection to highlight the boundaries of the traffic symbols, which would then be put into a convolutional neural network (CNN) for classification.

The authors of (Yang et al., 2017) proposed a traffic sign detection and classification system based on a scale-aware convolutional neural network (CNN). Two convolutional neural networks (CNNs) make up their system; one is used to recommend where traffic signs should be placed in a specific zone, while the other is used to categorise those signs. Moreover, scale invariant detection is achieved with the aid of a fully convolutional network (FCM). With a precision of 99.88%, this approach is quite precise. In (Yi et al., 2018), an efficient method for recognising traffic signs on low-cost embedded devices was reported. In their technique, they use colour thresholding, shape detection, and sign validation. Based on the red-blue angle colour transformation (RBAT) and normalised red, they used an efficient colour thresholding strategy. To identify spherical indications, the ellipse fitting approach is also utilised. For verification reasons, the HOG algorithm is utilised. Using this method, 97% of precision was accurate.

The technique described in (Zhu et al., 2018) takes a unique approach by identifying text-based traffic signs using two convolutional neural networks. The first CNN is in charge of identifying signs and crucial spots, while the second is responsible for locating text. The second convolutional neural network employs the architecture described in (Liao et al., 2017). After five phases of convolutional processing, the first convolutional neural network (CNN) model suggests the area of interest (ROI) (regions of interest). This approach offers a 90% accuracy rate and a 0.15-second classification time.

Researchers in (Hu et al., 2017) enhance the deep CNN with a branch-output approach for quick TSR testing. This method, which was inspired by a biological process, increases speed and accuracy by using past knowledge of indicators and their characteristics. In contrast to other approaches that require processing the whole image before categorising it, it may be sufficient to identify a small portion of a sign or even its shape. Prior to the development of Fast Branch, the majority of procedures were based on the incorrect assumption that the system needed a comprehensive view. The CNN architecture used by this model consists of three sets of convolutional layers, a pooling layer, and a normalisation layer. After the convolution and before to the pooling layer, the branches are generated. In theory, each set of layers may have an arbitrarily large n-count. Obtaining an accuracy of 99.5%.

“Using a deep hierarchical residual CNN model with a dilated skip connection, Saha conducted research entitled Total Recall: Understanding Traffic Signs with Deep Hierarchical Convolutional Neural Networks. There were an enormous 6,256 million model parameters. This approach "achieved" an identification accuracy of 99.33% on the GTSRB dataset and 99.17% on the BTSC dataset (Saha et al., 2019).”

In their research titled "Lightweight deep network for traffic sign classification," Zhang et al. proposed a knowledge distillation based lightweight DNN model. The conventional approach starts with training a network of teachers on the given data. To improve the teacher model, the authors incorporated a module that combines the feature stream with a thick layer. The filtered findings from the teacher network are then used to train a shallow network, or "student network," using the test data. Peak recognition accuracy on the GTSRB and BTSC for the student network was 99.61% and 99.13%, respectively, using 0.8 million and 7.4 million parameters. This approach goes through a 300-epoch training period. However, it faces convergence issues when obtaining competitive precision with a limited number of parameters. “Due to the fact that the student model is trained by minimising a loss function compared to the teacher model, convergence issues are inevitable”. The effectiveness of a student's training depends on the credentials of the teacher. Consequently, it is vital to determine whether the student models are getting enough general data. Additionally, each dataset requires the training of a teacher model. Before achieving high accuracy, it may need a significant number of training experiments with the student model due to instability issues (Zhang et al., 2020).

In another article, the authors offer RIECNN, a convolutional neural network that identifies traffic signals using real-time visual processing. The novel, real-time approach known as RIECNN beats state-of-the-art models in terms of recognition rate and execution time when applied to several traffic sign datasets. A set of tests using the Belgian Traffic Sign Classification (BTSC), the German Traffic Sign Benchmark (GTSRB), and the Croatian Traffic Sign (rMASTIF) classification systems. According to their study, their approach has "the highest recognition rate across all Benchmarks," with an accuracy of 99.75% for GTSRB, 99.25% for BTSC, and 99.55 % for rMASTIF” (Abdel-Salam et al., 2022).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Related works | Authors | Architectures | Evaluation metrics | Prediction outcomes |
| “Traffic-Sign Detection and Classification in the Wild” | (Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, 2016) | CNN | Accuracy | 84% |
| “A Survey on Road Traffic Sign Recognition System using Convolution Neural Network” | (Hatolkar et al., 2018) | CNN | Accuracy | NA |
| “Efficient Traffic-Sign Recognition with Scale-aware CNN” | (Yang et al., 2017) | CNN | Precision | 99.88% |
| “Knowledge-based Recurrent Attentive Neural Network for Traffic Sign Detection” | (Yi et al., 2018) | “Knowledge-based Recurrent Attentive Neural Network” | “Precision” | 97% |
| “Cascaded Segmentation-Detection Networks for Text-Based Traffic Sign Detection” | (Zhu et al., 2018) | CNN | Accuracy | 95% |
| “TextBoxes: A Fast Text Detector with a Single Deep Neural Network Minghui” | (Liao et al., 2017) | CNN | Accuracy | 90% |
| “Fast Branch Convolution  Traffic Sign Recognition  Neural Network for Traffic Sign Recognition” | (Hu et al., 2017) | CNN | Accuracy | 99.5% |
| “Total Recall: Understanding Traffic Signs using Deep Hierarchical Convolutional Neural Networks” | (Saha et al., 2019) | CNN | Accuracy | 99.33% |
| “Lightweight deep network for traffic sign classification Jianming” | (Zhang et al., 2020) | Lightweight deep network | Accuracy | 99.61% |
| “RIECNN: real-time image enhanced CNN for traffic sign recognition” | (Abdel-Salam et al., 2022) | RIECNN | Accuracy | GTSRB=99.75%  BTSC=99.25%  rMASTIF=99.55% |

Table 1 The existing CNN model on traffic signs datasets

# Methodology

The classification of traffic signs is one of those topics that are hardly addressed. The overwhelming majority of technologies now available are intended to identify only. To detect something, one must first extract its unique characteristics and then determine its position based on those characteristics. The categorization of images into various categories is what we refer to when we use the term classification. The most often used dataset for this purpose is GTSRB, which has 43 classes. This data is used by the proposed method to train a prediction model. It is effective when used for classifying images. In recent years, Convolutional Neural Networks have been used for object recognition because to its high accuracy and low cost of computing.

Diagram

Description automatically generated

Figure 1 Flowchart of methodology of this research

## Dataset

Before proceeding with detection or classification, it is essential to have access to a large dataset. This dataset is used to train a prediction model, which is subsequently used to generate predictions the test dataset. The sample datasets are shown in the following table.

|  |  |
| --- | --- |
| Dataset | Information |
| “ |  |
| “ |  |
| “ |  |
| “ |  |

Table 2 The existing traffic signs datasets

A group of red and white signs

Description automatically generated with medium confidence

Figure 2 Warning type traffic signs

A group of red and white signs

Description automatically generated with low confidence

Figure 3 Regulatory type traffic signs

A picture containing text, clipart

Description automatically generated

Figure 4 Direction type traffic signs

The German Traffic Sign Recognition Benchmark (GTSRB) dataset is the most prominent of these. The reason for its broad acceptance is:

1. It has an abundance of images.
2. Traffic signs come in a variety of designs, sizes, and colours, which all contribute to the accuracy of the model.

The GTSRB dataset is used by the proposed technique as a foundation for both detection and classification. In the dataset, there are around 50,000 images of different traffic signs. There are 43 other subclasses that it may be put into. Some categories in the collection include a large number of images, whilst others contain just a few. The size of the dataset is around at 314.36 MB, so it will not take long to download. There are two primary parts in this document: a "train" section and a "test" section. The "train" section has many classes, and each "class" section contains several images. Specifically, the Kaggle public dataset will be used for this assignment.

<https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign>



Figure 5 Sample of traffic signs

Deep Learning

In recent years, the performance of various fields, including object recognition and decision-making, has been significantly boosted due to the widespread acceptance and implementation of deep learning. Deep neural network techniques may automatically learn task-specific, hierarchical key properties from labelled input, unlike colour and shape-based systems. On numerous traffic-sign identification and classification datasets, notably the German dataset, methods based on local characteristics have lately showed promise. However, such solutions do not function well in real-world settings, especially when there are several variants of traffic signs (Yi et al., 2018).

## Conventional Neural Network method

The efficacy of Deep Learning over the last two decades has been primarily related to its ability to analyse enormous datasets. The interest in hidden layers has surpassed that of traditional approaches, especially in pattern recognition. Convolutional neural networks are among the most often used deep neural networks. The majority of its use was in the postal sector, where it was used to interpret different types of barcodes. For successful training, deep learning models need large amounts of data and powerful processors. Due to this severe constraint, CNNs were exclusively utilised in the postal industry at the time and never entered the field of machine learning.

Convolutional neural networks (CNN/ConvNet) are a kind of deep neural network often employed for image analysis in the area of deep learning. When the majority of people consider neural networks, matrix multiplications spring to mind. The technique of Convolution is used. Now in mathematics, convolution is an operation performed on two functions that results in a third function showing how the second set of functions impacted the shape of the first two functions.

Chart

Description automatically generated

Figure 6 Architecture of CNN model

CNN consist of many layers of synthetic neurons. In a rudimentary approximation to their biological counterparts, artificial neurons are mathematical functions that calculate the weighted sum of several inputs and provide an activation value. As a picture is input into a ConvNet, activation functions are formed and transmitted from one layer to the next. First-layer feature extraction often focuses on low-level properties “such as horizontal and diagonal edges. This output is received by the succeeding layer, which detects complex properties such as corners and combinational edges”. Complex items, such as objects, faces, and more, may be identified as we go further into the network.

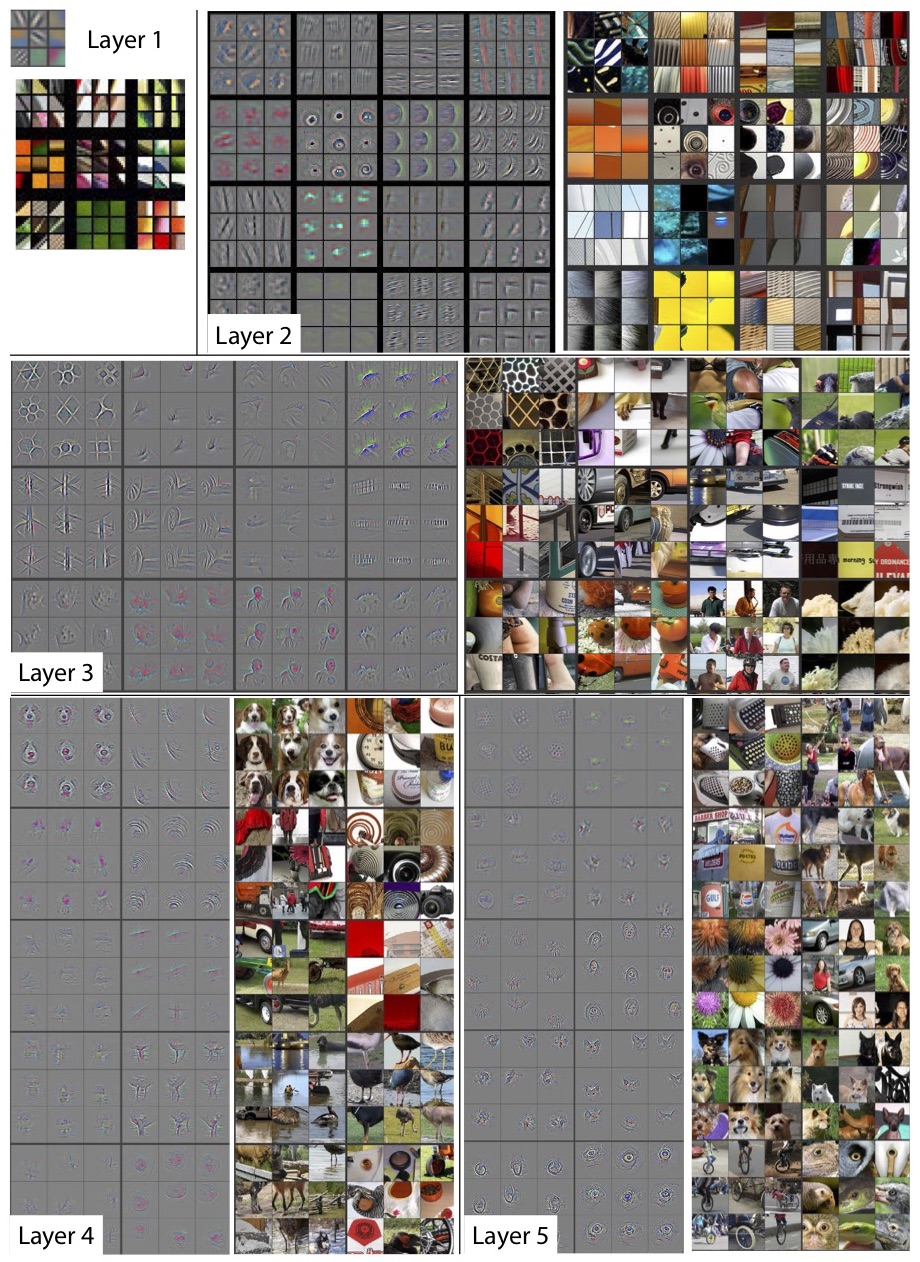
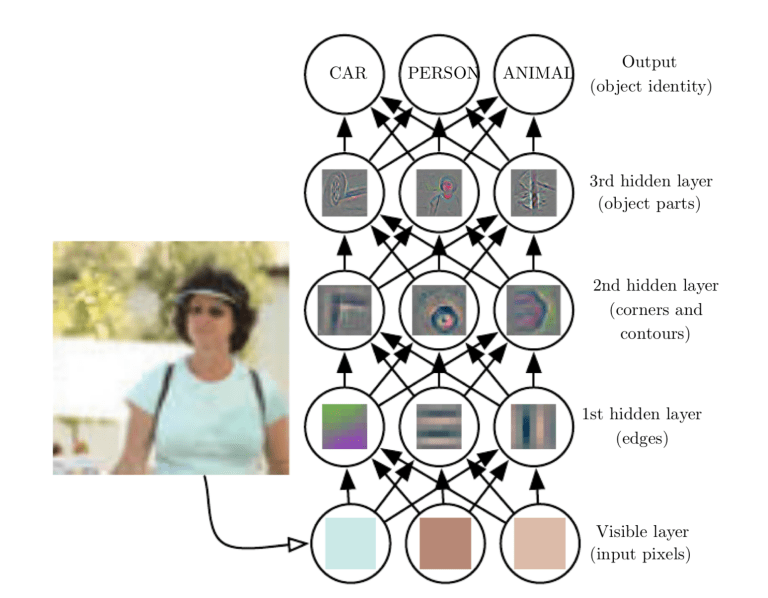


Figure 7Input images are processed by a neural network, with each layer identifying and highlighting unique characteristics.

The classification layer uses the activation map of the final convolution layer to provide confidence scores (0-1) that reflect the degree to which an image belongs to a certain "class." The output of the last layer of a ConvNet trained to recognise, for example, cats, dogs, and horses is the likelihood that the input image contains one of those animals.



“Figure 8 The top layer of the CNN classifies the image based on the features collected by convolutional layers.”

## Evaluation metric

Metrics used to evaluate efficacy provide insight on a model's functionality. Assessment measures need the ability to discern between model results. The accuracy of a model may be quantified by the number of accurate predictions it has made. The precision formula is as follows:

A picture containing text

Description automatically generated

A confusion matrix indicates the proportions of right predictions, wrong forecasts, and false negatives that a model produces. By extracting the appropriate values from the confusion matrix, we may determine the correctness of a model. The proportion of right responses may be calculated by applying the aforementioned procedure to the confusion matrix shown below.

Chart, treemap chart

Description automatically generated

Figure 9 Confusion matrix

## Data mining tool

The programming environment used for the thesis is Python. Data pre-processing libraries such as “tenserflow, keras, sklearn, matplotlib, pandas and pil will be used to support data pre-processing and build classification model. The model development is performed by utilizing Google Collab.”

# Implementation

Now according to a knowledge of how CNN functions, it is time to design the model. Researcher begin by studying the principles of a CNN, after which she creates the model and select the parameters properly. Researcher uses a small number of epochs (epoch=30) and a batch size of 62 to create the model since these two variables are known to vary from study to study.

## Pre-processing

Before developing a model, data processing is an essential prerequisite. This is the most important stage; even if you have the finest model, if you do not have adequate data, your results will not be satisfactory. So, researcher retrieved the data from the subdirectory 'train,' there are about 43 subfolders (0–42) that each represent a unique class and stored them in a data frame. Then she separated the data into two data frames, a training set and a test set.

researcher began by importing several library packages to load the data. As she cycles through each class, she add images and her related labels to a list of data and labels in accordance with the OS module. Second, she read the contents of images into an array using the PIL package. After that, she transformed all of the data and labels lists into numpy arrays and put them into the model that was built atop the images and annotations.

Graphical user interface, text, application

Description automatically generated

Figure 10 Code snippet of loading dataset

The dataset is then separated into training, test, and validation sets. During the development of the model, it is exposed to training dataset data. The validation dataset may be used for general model assessment and hyper-parameter adjustment. Various key parameters, including as the number of epochs and the activation function, may be modified to fine-tune and improve the accuracy of the learning process. Only once the model has been trained will it be evaluated on the test dataset. It is a method for determining if the model can accurately predict outcomes. For the training, testing, and validation data sets, histograms are also provided, where the X label indicates the "Class ID”, and the Y label reflects the total number of images in each class. A visual representation of the data is provided by the graph.

The train\_test\_split() method of the sklearn package is used to separate data between the training and testing stages. Using the to\_categorical function from the keras.utils package, the labels from y train and t test are converted to a one-hot encoding. The train and test images along with the labels are loaded and stored in variables X\_train, X\_test, y\_train, y\_test, respectively. Researcher must develop lists as a last option to keep track of each image and its associated labels. Before feeding this list to the model, she must thus convert it to a NumPy array. (13432, 30-30-3), where 13432 is the total number of images, 30-30-3 is the pixel width and height, and 3 is the RGB value (availability of coloured data).

Text

Description automatically generated

Figure 11 Code snippet of splitting the data



Figure 12 Code snippet of determine y\_train and y\_tets

Text, letter

Description automatically generated

Figure 13 Result of dataset

## Model Building

According to previous sections in this study, researcher categorised the images based on the CNN model into “their respective categories and construct a CNN model (Convolutional Neural Network).” CNN is the most effective in image classification. Based on this model the parameters are:

* 2 Conv2D layer which includes filter=32, kernel\_size=(5,5), activation= “relu”
* MaxPool2D layer includes pool\_size=(2,2)
* Dropout layer includes (rate=0.25)
* 2Conv2D layer includes filter=64, kernel\_size=(3,3),activation= “relu”
* MaxPool2D layer includes pool\_size=(2,2)
* Dropout layer includes rate=0.25
* Flatten layer to squeeze the layers into 1 dimension
* Dense Fully connected layer includes 256 nodes, activation= “relu”
* Dense layer includes 43 nodes, activation = “softmax”

The first step is to construct the framework for the CNN model (as seen in the fig. 2). The procedure is as follows:

“1) The following layers are required: two convolution layers, one pooling layer, one dropout layer, one flattening layer, one dense layer, one dropout layer, one more dense layer, and one more dense layer.

“2) There is a parameter for the total number of filters to utilise in the convolutional layer. The original image is subjected to convolution, and a feature map is generated.”

“3) To generate a rectified feature map, ReLU maximises negative values to zero without harming positive values. On the corrected feature map, the Pooling layer performs a down-sampling operation (Max Pooling or average pooling) to decrease the image's dimensionality.”

“4) Utilizing the flattening layer, the input feature map is reduced to a one-dimensional array.”

“5) The dropout layer may be used to avoid overfitting during training by setting a portion of the input neurons to 0. The dense layer, on the other hand, receives all outputs from the layer above it and sends them to all of its neurons, where they are multiplied by a matrix (the row vector of the output from the layer above should be identical to the column vector of the dense layer) to produce an m-dimensional vector.”

“6) Once all of the layers have been included, the model must be assembled using the categorical crossentropy loss function and the Adam optimizer (the last step in model construction after the loss function and optimization procedures have been designed and implemented). The proposed system is a multiclass classification problem in which many classes are analysed, yet each image corresponds to a single class; hence, a loss function must be defined.”

“7) Next, the model is trained using the training dataset by feeding it the preprocessed images.

“8) Finally, the predictions on the test data are done using the trained model and the traffic sign name along with the class Id is shown as an output.”

The model is constructed using the efficient Adam optimizer, with "categorical crossentropy" acting as the loss due to the existence of multiple classes.

Text

Description automatically generated

Figure 14 Code snippet of Model

## Model Visualization:

For deep learning, the Keras API was designed as a Python library. Researcher utilise this robust API with an attractive user interface to solve machine learning problems. The Keras plot library is a Tensorflow framework extension. It is designed to test new products in a certain manner. It will give her with the necessary basics and abstractions to package the machine learning solution. The model of the keras plot is represented by the graph's layers. Using the keras function plot model, the relevant graph may be generated. She may characterise it using the keras model and the keras model sequential function. Importing the plot model library is required. Here is the complete architecture of her CNN model with the following parameters:

A picture containing diagram

Description automatically generated

Figure 15 Code snippet of plot of model

Table

Description automatically generatedTable

Description automatically generated

Figure 16 plot of model Figure 17 Summary of mode

Model.fit() “is a reliable method for training models that may be used after the model architecture has been properly constructed.” Using 62 batch sizes, high accuracy was attained on training sets, and stability was reached after 30 iterations.



Figure 18 Code snippet of training model

Text, table

Description automatically generated

Table

Description automatically generated

Figure 19 Result of model

A picture containing text

Description automatically generated

Figure 20 code snippet of summary of model

A screenshot of a computer

Description automatically generated with low confidence

Figure 21 result of summary

## Accuracy Plot

Matplotlib is a toolkit for viewing and analysing model data.

Text

Description automatically generated with medium confidence

Figure 22 Code snippet of getting plot of accuracy and loss

# Results

Figure 23 and 24 displays the prediction accuracy on the training dataset. This is known as training accuracy. Likewise, the accuracy of the validation dataset symbolises the individual or entity being verified. This "hyperparameter" specifies the number of times the training dataset is analysed by the learning algorithm. Determines the number of times the model will be trained by repeatedly analysing the same data set. After around 30 epochs, the lines of training accuracy and validation accuracy begin to converge, as seen in Figures 22 and 23. Here, training accuracy and validation accuracy are at their best. If these lines begin to diverge in a predictable fashion, researcher may use this visualisation to decide the ideal time to stop training. This illustrates that just 30 epochs are required to extract features from the model.

Chart

Description automatically generated with low confidenceChart, histogram

Description automatically generated

Figure 23 Plot of accuracy Figure 24 Plot of loss

# Future research

The ability to read and follow traffic signs is advantageous for all drivers. With the use of traffic signs, motorists may respect all traffic regulations and safeguard pedestrians. Any real-time system is susceptible to environmental which may have an effect on detection and thus classification. To overcome these issues, further research and development are necessary. Chance also plays a role in the interpretation of certain traffic lights. One-hot encoding and enhancement are two ways that may be used for this goal. As part of the augmentation procedure, images might be moved, magnified, and rotated (if required).

This solution allows the driver to have quick access to the on-screen alert. Thus, it is unnecessary to physically search for a traffic sign board, assess its kind, and take the proper action. Therefore, Traffic Sign Classification has a wide range of applications in the development of more intelligent vehicles, such as self-driving cars, where the system can automatically detect, classify, and display a traffic sign.

# Conclusion

The suggested method efficiently gathers images, is simple to implement, and provides reasonably precise classification "on the GTSRB dataset and the newly generated one" (consisting of really existing images of all classes). Convolutional Neural Networks (CNN) are used in the recommended method for training the model. As part of the image preparation phase, histogram equalisation is conducted to increase accuracy and loss plot. As observed in the results section, the ultimate accuracy of the train dataset is excellent. The "Traffic Sign Categorization and Detection System" aids drivers largely by making their lives simpler. The categorising of traffic signs offers advantages, but it also has disadvantages. Occasionally, the traffic signs may be hidden or difficult to read. There is a need for more research in this area since events involving drivers who lose control of their cars pose a danger not only to themselves but also to other motorists and pedestrians.

# References

Abdel-Salam, R., Mostafa, R., & Abdel-Gawad, A. H. (2022). RIECNN: real-time image enhanced CNN for traffic sign recognition. *Neural Computing and Applications*, *34*(8), 6085–6096. https://doi.org/10.1007/s00521-021-06762-5

Alghmgham, D. A., Latif, G., Alghazo, J., & Alzubaidi, L. (2019). Autonomous Traffic Sign (ATSR) Detection and Recognition using Deep CNN. *Procedia Computer Science*, *163*, 266–274. https://doi.org/10.1016/j.procs.2019.12.108

Hatolkar, Y., Agarwal, P., & Patil, S. (2018). A Survey on Road Traffic Sign Recognition System using Convolution Neural Network. *International Journal of Current Engineering and Technology*, *8*(01), 104–108. https://doi.org/10.14741/ijcet/v.8.1.21

Hu, W., Zhuo, Q., Zhang, C., & Li, J. (2017). Fast Branch Convolutional Neural Network for Traffic Sign Recognition. *IEEE Intelligent Transportation Systems Magazine*, *9*(3), 114–126. https://doi.org/10.1109/MITS.2017.2709780

Li, J., & Wang, Z. (2019). Real-Time Traffic Sign Recognition Based on Efficient CNNs in the Wild. *IEEE Transactions on Intelligent Transportation Systems*, *20*(3), 975–984. https://doi.org/10.1109/TITS.2018.2843815

Liao, M., Shi, B., Bai, X., Wang, X., & Liu, W. (2017). TextBoxes: A fast text detector with a single deep neural network. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 4161–4167. https://doi.org/10.1609/aaai.v31i1.11196

Radu, M. D., Costea, I. M., & Stan, V. A. (2020). Automatic Traffic Sign Recognition Artificial Inteligence - Deep Learning Algorithm. *Proceedings of the 12th International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2020*. https://doi.org/10.1109/ECAI50035.2020.9223186

Saha, S., Kamran, S. A., & Sabbir, A. S. (2019). Total Recall: Understanding Traffic Signs Using Deep Convolutional Neural Network. *2018 21st International Conference of Computer and Information Technology, ICCIT 2018*, 2–7. https://doi.org/10.1109/ICCITECHN.2018.8631925

Yang, Y., Liu, S., Ma, W., Wang, Q., & Liu, Z. (2017). Efficient traffic sign recognition with scale-aware CNN. *British Machine Vision Conference 2017, BMVC 2017*. https://doi.org/10.5244/c.31.168

Yi, K., Jian, Z., Chen, S., & Zheng, N. (2018). *Feature Selective Small Object Detection via Knowledge-based Recurrent Attentive Neural Network*. http://arxiv.org/abs/1803.05263

Zhang, J., Wang, W., Lu, C., Wang, J., & Sangaiah, A. K. (2020). Lightweight deep network for traffic sign classification. *Annales Des Telecommunications/Annals of Telecommunications*, *75*(7–8), 369–379. https://doi.org/10.1007/s12243-019-00731-9

Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-Sign Detection and Classification in the Wild. *Thermal Engineering*, *48*(10), 835–841.

Zhu, Y., Liao, M., Yang, M., & Liu, W. (2018). Cascaded Segmentation-Detection Networks for Text-Based Traffic Sign Detection. *IEEE Transactions on Intelligent Transportation Systems*, *19*(1), 209–219. https://doi.org/10.1109/TITS.2017.2768827