WWW.IJRPR.COM

**International Journal of Research Publication and Reviews**

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

# Indian Traffic Sign Detection and Recognition System Using CNN

***[1] Gautam Mishra, [2] Dr. Tejna Khosla***

[1 2] Department of Information Technology Maharaja Agrasen Institute of Technology New Delhi , India

[1] gautam20mishra@gmail.com

[2] Tejnakhosla@mait.ac.in

**ABSTRACT :**

A key component of the Intelligent Transportation System is the Indian Traffic Sign Detection and Recognition system, which helps drivers drive more cautiously and professionally by using traffic signs. The construction and operation of an Indian traffic sign detection and recognition system utilizing Convolutional Neural Networks (CNNs) for precise traffic sign detection and classification from photos are demonstrated in this research. Image pre-processing, CNN model, and post-processing are the three primary components of the system, which produced a high classification accuracy of 82.21%. The CNN model classifies input photos into several traffic sign categories using multiple convolutional and fully connected layers, whereas the pre-processing step entails scaling, normalization, and augmentation. A standard data set is also taken into consideration to assess the system's performance, and the robustness of the system is cross-examined for illumination, scale, rotation, and similar color and shape fluctuations. Experiments show that the system is highly successful and reliable in a variety of situations, highlighting its potential to lower traffic incidents and improve traffic management in general. CNN-enabled visual and aural recognition combined.

Keywords: deep learning, cars, visual recognition, road signals, CNN, accident detection, and intelligent transportation systems

## 1 Introduction

In recent years, there has been an increase in interest in the detection and recognition of traffic signs. For the intelligent vehicle, it is even regarded as one of the most crucial jobs. Both autonomous driving cars and sophisticated driver support systems depend on the detection and recognition of traffic signs. Roadside traffic signs serve a variety of purposes, including informing drivers of the state of the road ahead, providing guidance to drivers at important intersections, and ensuring that traffic moves smoothly. The general strategy for traffic sign detection systems focuses on both identification and detection. Additionally, a traffic sign detection model should possess extensive information about the area of interest.

The built model should be able to recognize the region of attention based on the generated model and the extensive information found in a road scene. Although they have been mentioned since 1990, automatic traffic sign detection and recognition has been the focus of numerous investigations. The distinctive color and shape of traffic signs set them apart from their surroundings. This data can be used for identification and detection.

Numerous factors, including changes in lighting, scale, motion blur, geometric distortions, weather, and complicated backgrounds, might cause different people to recognize traffic signs. The segmentation process, which is essential to high-level traffic sign detection and recognition, is impacted by the issues mentioned above.

In addition to shape, color is a crucial factor that distinguishes traffic signs from other nations. These traffic signs remind commuters of the laws they must abide by and provide them directions on how to move through traffic. Consequently, based on the information they offer, Indian traffic signs are divided into three groups. Triangular, circular, inverted triangular, octagonal, and rectangular shapes with red, blue, and green hues are the most common geometric shapes found on Indian traffic signs. These color and shape data were utilized by numerous researchers to identify and detect Indian traffic signs.

In order to locate the traffic sign in an obtained image, its distinctive color and shape information are highly adjustable. Sign detection is a constant challenge in India because of environmental factors like mist, fog, pollution, weather, and deteriorated sign boards. Therefore, it is crucial that the built-in color and form models for traffic sign detection be resilient to changes in the environment.

**Neural networks that are convolutional:**

Applications in computer vision have been transformed by a class of neural networks called convolutional neural networks, or CNNs. CNNs are made to interpret and categorize visual input, and their architecture and operation are modeled after that of the visual cortex in animals. They are made up of several layers of neurons that extract information and generate predictions by applying various operations to the input data.

## 2 Literature review

A significant use of computer vision, traffic sign recognition has been the subject of numerous studies in recent years. Convolutional neural networks (CNNs), which have demonstrated outstanding performance in a variety of visual recognition tasks, are among the most promising methods for traffic

sign recognition. In order to automatically learn and extract features from picture data, CNNs are a type of deep learning model. They are composed of several convolutional layers that apply a set of learnt filters to the input image. The feature maps are then down-sampled by pooling layers.
The final classification decision is then made by running the resultant features through one or more fully linked layers. CNNs have been employed in a number of studies to recognize traffic signs, and on a variety of benchmark datasets, the results have been state-of-the-art. A multi-stage CNN design, comprising a convolutional network, a spatial pyramid pooling layer, and a multi-layer perceptron for classification, was employed by Sermanet et al. in one such study for traffic sign identification. The writers assessed their strategy using the German. The accuracy was 98.47% on the Traffic Sign Recognition Benchmark (GTSRB) dataset. In a related study, Zhang et al. presented Traffic Sign Net, a novel CNN architecture for traffic sign identification that includes a dynamic network reconfiguration module to adapt the network design for various traffic sign scales. Using the China Traffic Sign Recognition (CTSR) dataset, the authors tested their method and obtained a 98.51% accuracy rate. For traffic sign recognition, Li et al. [12] suggested a hybrid CNN and hand-crafted feature-based method in which the CNN learns features from the input photos and the hand-crafted features are utilized to enhance the CNN features. The accuracy of 99.5%, the current state-of-the-art result on the GTSRB dataset, was attained by the authors when they tested their method on this dataset.

In order to enhance the generalization performance of CNN models for traffic sign identification, future research could concentrate on creating more reliable and effective CNN architectures as well as investigating the application of synthetic data creation techniques.

The fields of autonomous driving and intelligent transportation systems both greatly benefit from research on Indian traffic sign detection and recognition systems. The main contributions are as follows:

***1. Dealing with Regional Issues:***

Indian traffic signs frequently have different designs and meanings than those in Western nations due to cultural and regional differences. Models are trained on pertinent data thanks to research specifically designed for Indian circumstances, which increases accuracy and dependability in the local environment.

Environmental Factors: Specialized strategies are required to address the particular difficulties presented by India's varied weather patterns, road kinds, and traffic patterns. The goal of research is to create systems that function well in a variety of lighting and weather scenarios, which is essential for real-time applications.

***2. Advancements in Deep Learning Techniques***

Novel Model Architectures:
To improve performance in detecting and identifying traffic signals, recent research has developed sophisticated architectures such as the Mask R-CNN. By pushing the limits of current deep learning techniques, these developments add to the corpus of knowledge in computer vision.

Data Augmentation and Real-Time Processing:
Researchers have created methods for data augmentation that improve model resilience by using large datasets (e.g., 6480 photos with 7056 instances). This approach is crucial for achieving high accuracy rates in real-time applications, which is vital for autonomous cars.

***3. Enhancing Road Safety***

Impact on Traffic Management:
By giving drivers and self-driving cars timely information, efficient traffic sign detecting systems can drastically lower the number of accidents. High precision rates (e.g., 97.08% accuracy) are indicated by research findings, and these can be directly translated into better road safety measures.

Integration with Intelligent Transportation Systems (ITS):
These research' results encourage the creation of all-encompassing ITS, of which automatic traffic sign recognition is a key element, improving overall traffic control and safety procedures.

***4. Contributions to Autonomous Vehicle Development***

Basis for Autonomous Navigation:
By accurately identifying and reacting to traffic signs, the research offers fundamental algorithms and frameworks that may be incorporated into autonomous cars to allow for safe and effective navigation.

**Real-Time Decision Making:**

These systems aid with autonomous cars' real-time decision-making processes by enhancing the speed and precision of traffic sign detection, which is essential for their safe operation in changing situations.

***5. Future Research Directions***

Increasing Dataset Diversity: To improve model training and generalization capabilities across different locations, ongoing research highlights the need for additional diverse datasets that cover a range of Indian traffic circumstances.

Investigation of Hybrid Models: To increase detection rates and decrease false positives, future research may investigate hybrid strategies that integrate CNNs with additional machine learning techniques. This will increase system reliability in practical applications.

## 3 Methodology

**Description of the Dataset :**
A dataset is needed for the model's training in order to detect the Indian traffic sign. Thus, the Indian Traffic Sign Dataset (ITSD) serves as a standard and dataset for assessing algorithms for traffic sign recognition. The dataset includes a CSV file and images of 59 different types of traffic signs. 11,859 photographs of traffic signs from actual scenes are included, including 59 distinct kinds of traffic signs with varying sizes, shapes, colors, and directions, such as yield signs, stop signs, speed limits, and more. It is a useful tool for recognition since it offers a difficult and varied dataset for assessing and contrasting various methods and algorithms.
1. A traffic sign detection and recognition system was developed using an Indian traffic sign dataset.

2. Any traffic sign project based on image classification can use the dataset.

3. The dataset's photos have dimensions of 32 by 32 3.

4. A government website image of each class was used to construct the dataset, and additional image scaling was done.

5.Preprocessing, grayscale, and standardized lighting are applied to the images.

6. In order to identify the sign provided to the system, it also includes a CSV file with the class name and their IDs.

***Training and Testing Traffic sign model***

In computer vision and transportation systems, training and testing a traffic sign model is an essential activity. The dataset required to be used for model training, and the training images needed to be processed in a way that made them suitable for training. Therefore, depending on the needs, there are Python libraries that may be used to process and convert images. Accurately identifying traffic signs is the model's goal since it can greatly improve road safety. Traffic sign identification is a complex operation that requires a robust feature extraction and classification system. Sign recognition is one of the many computer vision applications where convolutional neural networks (CNNs) have shown remarkable performance.

***Numerical operations (matrices, arrays, etc.) are performed using numpy.***

***Plotting graphs (training loss, accuracy, etc.) is done with matplotlib.pyplot.***

***Sequential: Specifies a CNN model's linear layer stack.***

***CNN model layers include Dense, Dropout, Flatten, Conv2D, and MaxPooling2D.***

***Adam: The CNN is trained using this optimizer.to_categorical: transforms class labels into a format that is one-hot encoded.***

***cv2: The OpenCV image processing library.***

***The dataset is divided into training, validation, and test sets using the train_test_split function.***

***(Not used here) pickle usually employed for data loading and saving.***

***OS: Used to manage files and directories.***

***Pandas: For working with CSV files.***

***ImageDataGenerator: A tool for augmenting data.***

***path: The directory containing the photos of traffic signs.***

***labelFile: The location of the labeled CSV file.***

***batch_size_val: The quantity of photos handled during a single training phase.***

***epochs_val: The number of times the model runs the complete dataset.***

***pictureThe image's dimensions are 32 x 32 pixels with three color channels.***

***testRatio: Testing takes up 20% of the data.***

***verificationFor validation, 20% of the training data is further divided.***

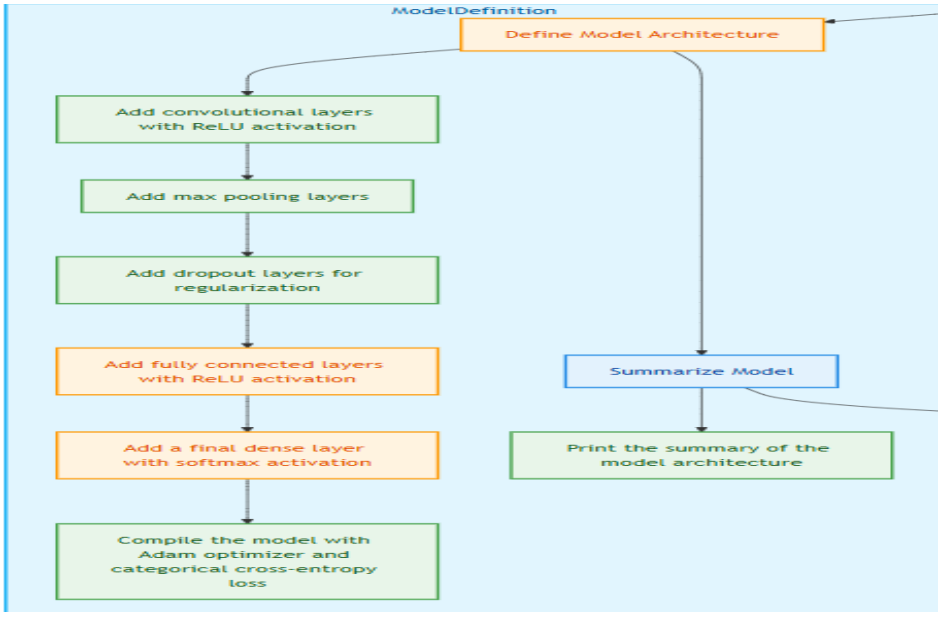



Fig 1. Flow of Model Definition

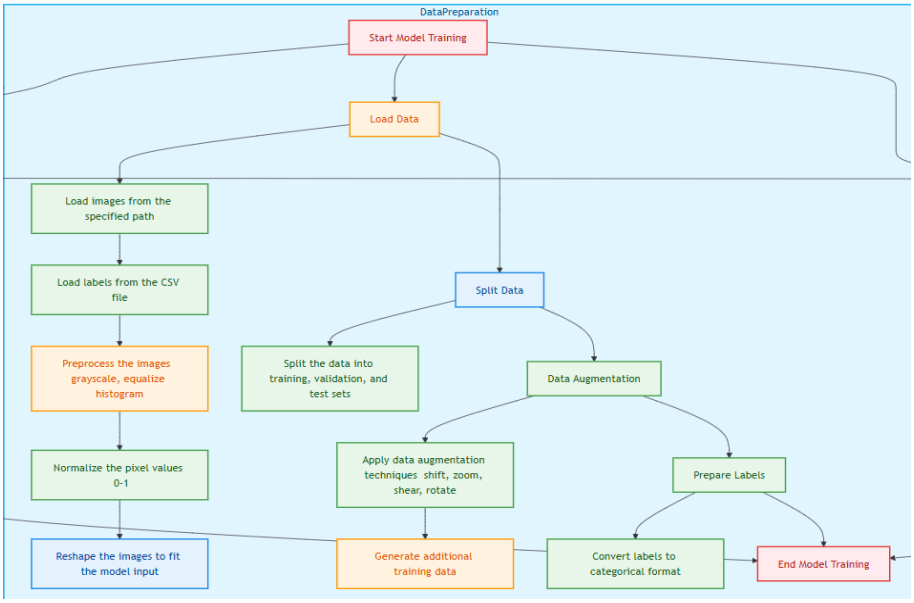



Fig 2. Flow of Data Preparation

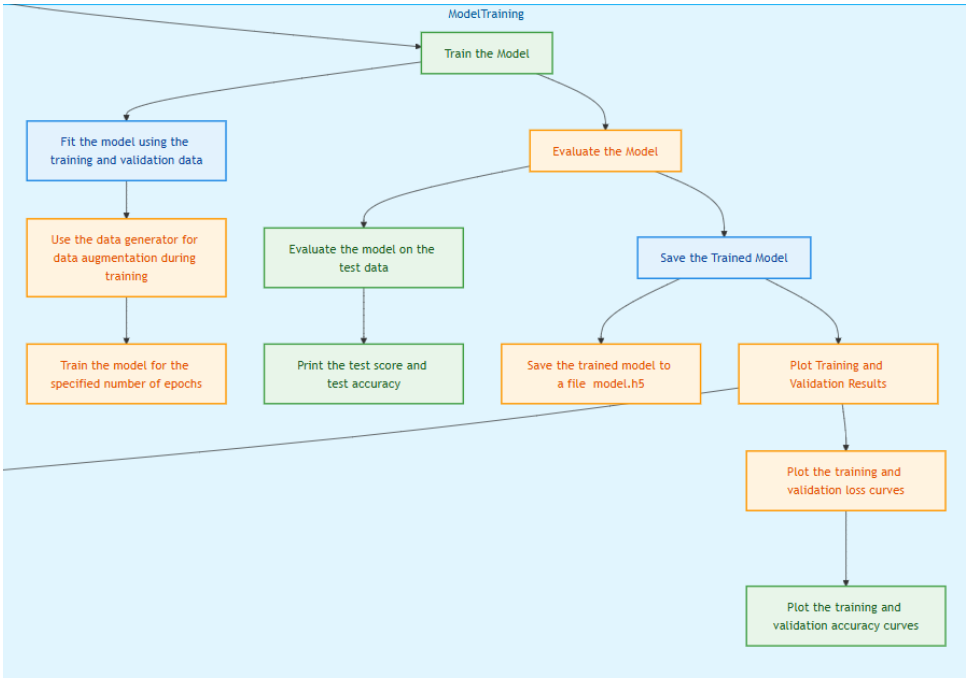



Fig 3. Flow of Model Training

***Flask Setup:-***

The flask server is first initialized, ready to load the trained model, and the upload system is configured. Make sure the upload directory is present before loading the model from a file. The unsupported markdown then appears: Codespan recommends that a specific filename or location be included here. The system checks to see if the uploads directory is there before allowing user-uploaded files. The uploads directory is created dynamically to store files if it is absent. The directory name is once more suggested to be a placeholder by "Unsupported markdown: codespan". The upload folder is then saved in the Flask configuration settings after the step has configured Flask to recognize the newly formed directory. The Flask setup is finished once the model has been loaded and the upload folder has been configured.

***Image Preprocessing:-***

The pipeline for preprocessing is started. This guarantees that every image is in the right format for training the model. The image is then loaded by the system from a specified file path. Usually, OpenCV (cv2) is used to load images in the BGR (Blue-Green-Red) format. OpenCV uses this format by default, however other libraries, such as PIL, use RGB. The image is reduced to grayscale since traffic sign identification does not always require color information.As a result, there are now only one channel (Grayscale) instead of three (BGR). The BGR picture is transformed into a single-channel grayscale image using the OpenCV function cv2.COLOR_BGR2GRAY. This preserves key characteristics while lowering computational complexity. maintains homogeneity by standardizing all photos to 32x32 pixels. This guarantees that the dimensions of every input image for training are the same. To guarantee constant input size, OpenCV's cv2.resize(image, (32,32)) function is used. increases the contrast of an image to help distinguish features. This is helpful for enhancing visibility in photos with inadequate illumination. makes use of the cv2.equalizeHist(image) function in OpenCV. By more equally dispersing pixel intensity levels, this enhances contrast. changes pixel values from 0 to 255 to a range of 0 to 1. As a result, training the model becomes faster and more stable. The picture is resized to fit the model's specifications.

***Model prediction:-***

An input image is classified into a particular traffic sign category by a trained deep learning model during the crucial Model Prediction Process. This process's steps guarantee that the image is appropriately processed before being fed into the model and that the predictions are correctly interpreted.

***Flask Routes:-***

Flask is a lightweight Python web framework for building online apps. The Flask routes in an application that manages file uploads, processes predictions using a trained model, and provides the user with results are depicted in the flowchart.

## 4 Results

### *Model Summary*

The sequential convolutional neural network (CNN) model was created specifically for the detection and recognition of Indian traffic signs. Several convolutional layers (Conv2D), max-pooling layers, dropout layers, and fully connected (dense) layers make up the network.

• Convolutional Layers: Use filters (kernels) to extract features. In order to reduce spatial dimensions, the next two Conv2D layers contain 30 filters, whereas the previous two have 60 filters.

• Max-Pooling Layers: These layers downsample feature maps to minimize computation and preserve crucial information.

• Dropout Layers: These layers randomly set neurons to zero in order to help prevent overfitting.

• Flatten Layer: For the completely connected layers, this layer transforms 2D feature maps into a 1D vector.

• Dense Layers: Use 500 neurons for the final classification and 58 neurons for the output layer, which is the same number of classifications of traffic signs.

### *Training Performance*

• After ten epochs of training, the model's accuracy and loss decreased.

• The last epoch's training accuracy increased from 11.75% to 65.4%.

• The validation accuracy showed good generalization, rising from 31.17% to 83.63%.

### *Loss and Accuracy Graphs*

• Loss Graph: Indicates successful learning with a declining trend.

• Accuracy Graph: Dropout regularization may be the reason why validation accuracy is higher than training accuracy.

### *Test Performance*

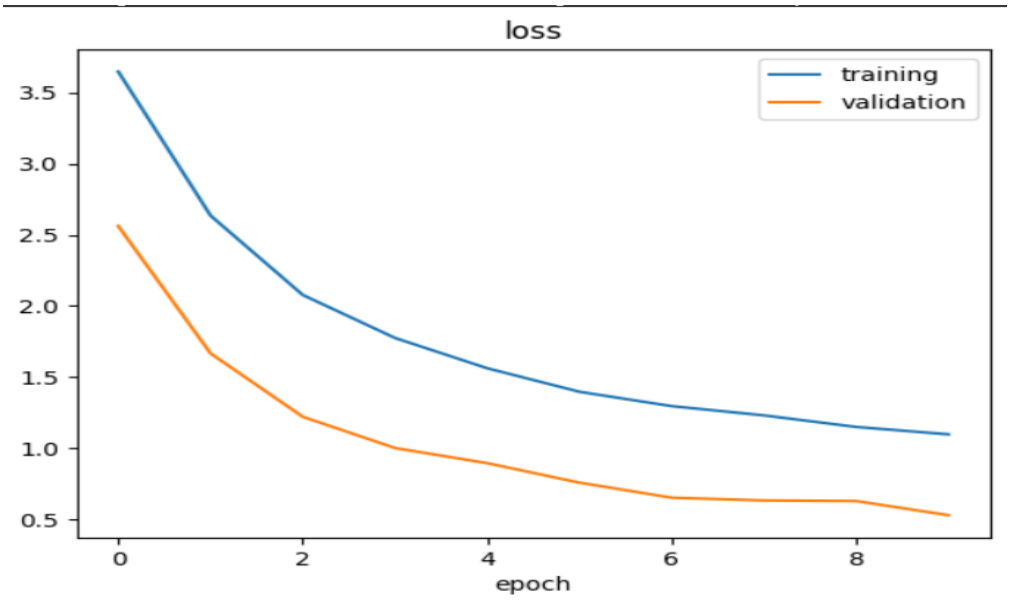
• 82.21% test accuracy indicates that the model has good generalization.

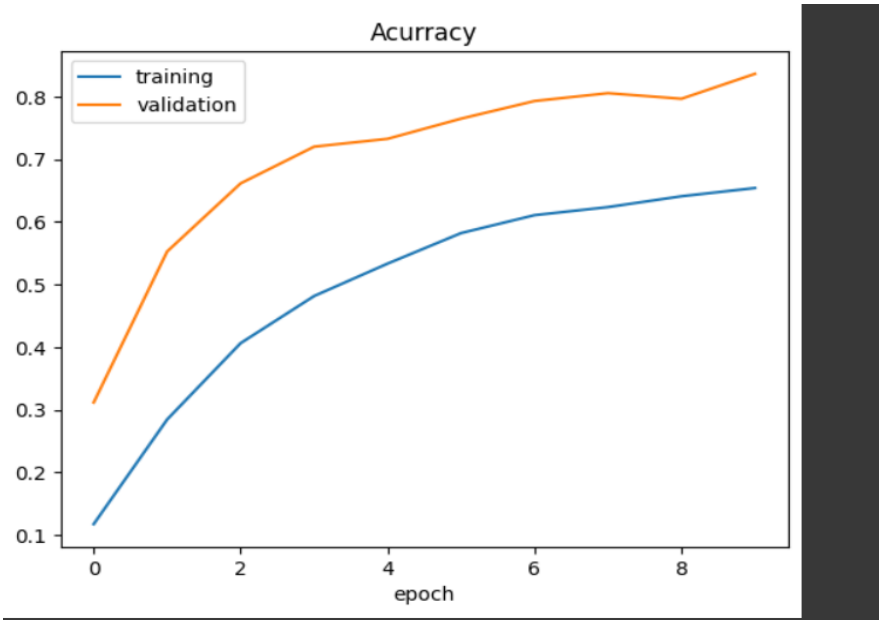
• The test loss was a comparatively modest 0.53.

```
Total Classes Detected: 58
Importing Classes.....
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57
Data Shapes
Train(8940, 32, 32, 3) (8940,)
Validation(2236, 32, 32, 3) (2236,)
Test(2795, 32, 32, 3) (2795,)
data shape (59, 2) <class 'pandas.core.frame.DataFrame'>
WARNING:absl:`lr` is deprecated in TF-Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.Adam.
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 60)        1560

 conv2d_1 (Conv2D)           (None, 24, 24, 60)        90060

 max_pooling2d (MaxPooling2  (None, 12, 12, 60)        0
 D)

 conv2d_2 (Conv2D)           (None, 10, 10, 30)        16230

 conv2d_3 (Conv2D)           (None, 8, 8, 30)          8130

 max_pooling2d_1 (MaxPoolin  (None, 4, 4, 30)          0
 g2D)

 dropout (Dropout)           (None, 4, 4, 30)          0

 flatten (Flatten)           (None, 480)               0
```

```
flatten (Flatten)            (None, 480)               0

dense (Dense)                (None, 500)               240500

dropout_1 (Dropout)          (None, 500)               0

dense_1 (Dense)              (None, 58)                29058

=================================================================
Total params: 385538 (1.47 MB)
Trainable params: 385538 (1.47 MB)
Non-trainable params: 0 (0.00 Byte)
```

```
<ipython-input-28-07e239ed873b>:123: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `
  history = model.fit_generator(dataGenerator.flow(X_train, y_train, batch_size=32), steps_per_epoch=len(X_train) // 32, epochs=epochs_val,
None
Epoch 1/10
279/279 [==============================] - 95s 336ms/step - loss: 3.6462 - accuracy: 0.1175 - val_loss: 2.5614 - val_accuracy: 0.3117
Epoch 2/10
279/279 [==============================] - 94s 338ms/step - loss: 2.6351 - accuracy: 0.2846 - val_loss: 1.6671 - val_accuracy: 0.5528
Epoch 3/10
279/279 [==============================] - 93s 332ms/step - loss: 2.0775 - accuracy: 0.4064 - val_loss: 1.2218 - val_accuracy: 0.6614
Epoch 4/10
279/279 [==============================] - 97s 347ms/step - loss: 1.7753 - accuracy: 0.4816 - val_loss: 1.0021 - val_accuracy: 0.7200
Epoch 5/10
279/279 [==============================] - 93s 332ms/step - loss: 1.5625 - accuracy: 0.5333 - val_loss: 0.8958 - val_accuracy: 0.7326
Epoch 6/10
279/279 [==============================] - 92s 331ms/step - loss: 1.3972 - accuracy: 0.5821 - val_loss: 0.7588 - val_accuracy: 0.7648
Epoch 7/10
279/279 [==============================] - 97s 348ms/step - loss: 1.2964 - accuracy: 0.6107 - val_loss: 0.6532 - val_accuracy: 0.7929
Epoch 8/10
279/279 [==============================] - 92s 330ms/step - loss: 1.2314 - accuracy: 0.6236 - val_loss: 0.6336 - val_accuracy: 0.8055
Epoch 9/10
279/279 [==============================] - 92s 331ms/step - loss: 1.1502 - accuracy: 0.6409 - val_loss: 0.6296 - val_accuracy: 0.7965
Epoch 10/10
279/279 [==============================] - 96s 344ms/step - loss: 1.0984 - accuracy: 0.6540 - val_loss: 0.5304 - val_accuracy: 0.8363
```




```
Test Score: 0.5364290475845337
Test Accuracy: 0.8221824765205383
/usr/local/lib/python3.11/dist-packages/tf_keras/src/engine/training.py:3098:
  saving_api.save_model(
```

## 5 Conclusion

When it comes to accurately recognizing Indian traffic signs, the CNN model is doing well. Performance can be further increased by making certain adjustments, such as expanding training data or adjusting hyperparameters.

In this study, a traffic sign recognition system built with Convolutional Neural Networks (CNN) is presented as a viable way to improve road safety and decrease accidents caused by driver mistake.

With a high classification accuracy of 83.63% on a validation set and 82.21% on a real-world dataset, the system proved its resilience to changing weather and lighting circumstances as well as its capacity to correctly identify traffic signs at various angles and distances. The three primary parts of the system—the CNN model, post-processing, and picture pre-processing—have all been thoroughly explained.

## 6 Reference

1. Handmann, U., Kalinke, T., Tzomakas, C., Werner, M., & Seelen, W.: A driver assistance image processing system. Comput. Image Vis. 18(5), 367–376 (2000)

2. Timofte, R., et al.: Using 3D tracking and traffic sign detection to improve driver assistance. Comp. Vis. and its App. Emerg. Top. 1–22 (2011)

3. Swathi, M., et al.: A review of automatic traffic sign recognition and detection. Algorithms, Methodology, Models, and Applications in Emerging Technologies: An International Conference (ICAMMAET), pp. 1–17 (2017)

4. Background on traffic sign identification and recognition by Escalera, S., et al. Systems for recognizing traffic signs, pages. 5–13. Springer, 2011

5. Yakimov et al.: Using tracking and prediction to detect traffic signs. International Conference on Telecommunications and E-Business Springer, Colmar, 2015, 454–467

6. Brkic et al.: A summary of techniques for detecting traffic signs. Faculty of Electrical Engineering and Computing, Department of Electronics, Microelectronics, Computer and Intelligent Systems, 1–9 (2010)

7. Saadna, Y., and Behloul, A.: A summary of techniques for detecting and classifying traffic signs. Inf. Retr. Int. J. Multimed. 6(3), 193–210 (2017)

8. H. Kamada et al.: A small navigation system that uses fuzzy control and image processing, IEEE South East Conference Proceedings, 337–342 (1990)

9. R. Janssen et al.: Traffic sign recognition using a hybrid technique. IEEE International Conference on Intelligible Vehicles, Proceedings, 390–397 (1993)