BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

Autonomous Institute under VTU, Belagavi | Approved by AICTE,New Delhi Recognized by Govt. of Karnataka

NACC Accredited Institution*
(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belgavi)
"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballar1-583 104 (Karnataka) (India)
Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE - ARTIFICIAL INTELLIGENCE

A Project Report On

"Traffic sign recognition"

Project Associates:

Hemalatha D	**3BR22CA019**

Under the Guidance of

Prof. Pavan Kumar

Mr. Vijay Kumar

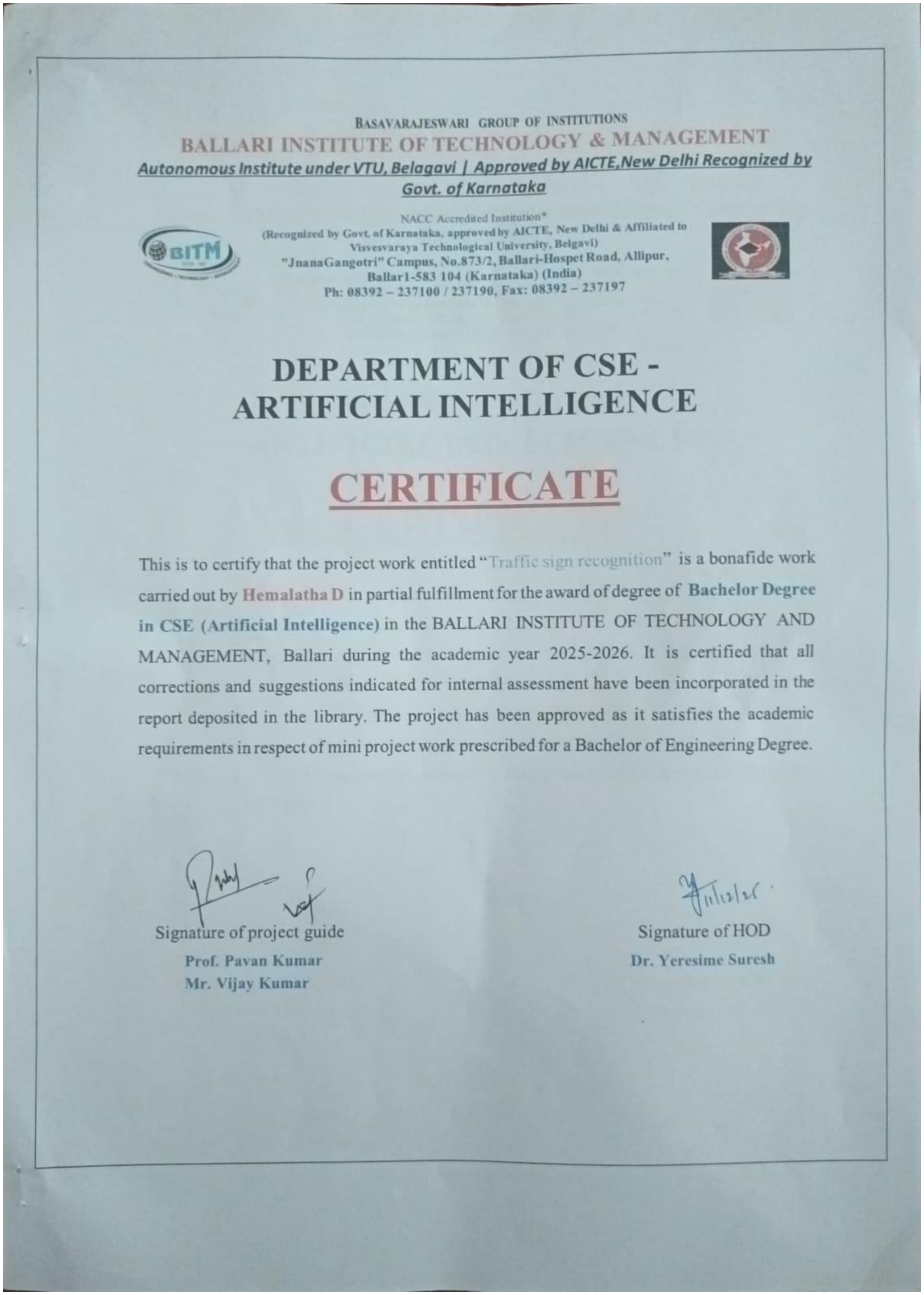
Dept. of CSE-Artificial Intelligence

BITM, Ballari.

Visvesvaraya Technological University

Belagavi, Karnataka

2025-2026



BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

Autonomous Institute under VTU, Belagavi | Approved by AICTE,New Delhi Recognized by Govt. of Karnataka

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belgavi)

"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballar1-583 104 (Karnataka) (India)

Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE - ARTIFICIAL INTELLIGENCE

CERTIFICATE

This is to certify that the project work entitled "Traffic sign recognition" is a bonafide work carried out by **Hemalatha D** in partial fulfillment for the award of degree of **Bachelor Degree in CSE (Artificial Intelligence)** in the BALLARI INSTITUTE OF TECHNOLOGY AND MANAGEMENT, Ballari during the academic year 2025-2026. It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report deposited in the library. The project has been approved as it satisfies the academic requirements in respect of mini project work prescribed for a Bachelor of Engineering Degree.

[illegible] [illegible]

Signature of project guide

Prof. Pavan Kumar

Mr. Vijay Kumar

[illegible]

Signature of HOD

Dr. Yeresime Suresh

Abstract

Traffic sign recognition is an important component of intelligent transportation systems and autonomous vehicles. This project presents a Convolutional Neural Network (CNN)–based traffic sign recognition system using the German Traffic Sign Recognition Benchmark (GTSRB) dataset. Image preprocessing techniques such as resizing, normalization, and one-hot encoding are applied to improve model performance. The CNN model effectively extracts visual features and classifies traffic signs into 43 categories with high accuracy. Performance is evaluated using accuracy, precision, recall, F1-score, and confusion matrix. The results demonstrate the effectiveness of deep learning in accurately recognizing traffic signs for real-world applications.

Acknowledgement

The satisfaction that accompanies the successful completion of the project on *Traffic sign recognition* would be incomplete without acknowledging the people whose noble gestures, affection, guidance, encouragement, and support made this achievement possible. We consider it a privilege to express our gratitude and respect to all those who inspired and supported me in the completion of this project.

I am extremely grateful to our guide, **prof. Pavan Kumar and Mr. Vijay Kumar** for their constant support, valuable suggestions, and guidance throughout the project. Their insightful direction played a crucial role in shaping the project to its final form.

I would also like to extend my sincere thanks to **Dr. Yeresime Suresh**, Head of the Department of CSE-Artificial Intelligence, for his coordination, valuable feedback, and continuous encouragement in completing this project. His contributions were invaluable.

Name	USN
Hemalatha D	3BR22CA019

Table of Contents

List of Figures

Chapter 1
INTRODUCTION

Traffic sign recognition is essential for road safety and intelligent transportation systems. Traffic signs provide important information such as speed limits, warnings, and mandatory instructions that help drivers make safe decisions. Failure to recognize traffic signs can lead to accidents and traffic violations, creating a need for automated recognition systems.

Traditional methods based on human observation and basic image processing are unreliable under real-world conditions such as poor lighting, weather changes, and distractions. To overcome these challenges, deep learning techniques are widely used.

This project develops a CNN-based traffic sign recognition system using the GTSRB dataset, which contains 43 different traffic sign classes. Images are preprocessed and trained using a CNN model to achieve accurate classification. The system demonstrates effective performance and practical applicability through a graphical user interface, showing the importance of deep learning in autonomous driving and smart transportation.

1.1. Problem Statement

Incorrect or missed recognition of traffic signs can lead to road accidents and traffic violations. Manual observation and traditional image processing techniques are unreliable under varying lighting, weather, and road conditions. With the rise of autonomous vehicles and intelligent transportation systems, there is a need for an automated and accurate traffic sign recognition system. This project aims to develop a CNN-based traffic sign detection model that can reliably classify traffic signs from images and improve road safety.

1.2. Scope of the project

The scope of this project is to design and implement a CNN-based traffic sign recognition system capable of classifying traffic signs from image data. The system focuses on image preprocessing, feature extraction, model training, and evaluation using standard performance metrics. Although the project is implemented using a benchmark dataset, the proposed approach can be extended to real-time traffic sign detection systems in autonomous vehicles, driver assistance systems, and intelligent transportation applications.

1.3 Objectives

The objectives of this project are:

- To develop a CNN-based traffic sign recognition model.
- To preprocess and classify traffic sign images effectively.
- To train and evaluate the model using accuracy and loss metrics.
- To analyze performance using confusion matrix and classification measures.
- To design a simple GUI for traffic sign prediction.

Chapter 2

LITERATURE SURVEY

[1] Zhang et al. (2021) proposed a deep learning-based traffic sign recognition system using Convolutional Neural Networks (CNN). Their study demonstrated that CNNs significantly outperform traditional image processing methods, achieving high accuracy even under challenging conditions such as varying illumination and complex backgrounds.

[2] Li et al. (2022) developed an improved CNN model combined with data augmentation techniques for traffic sign classification. The results indicated that data augmentation enhances model robustness and reduces overfitting, leading to improved recognition accuracy on unseen traffic sign images.

[3] Kumar and Singh (2022) investigated traffic sign detection using a CNN-based approach and highlighted the importance of image preprocessing steps such as resizing and normalization. Their findings showed that proper preprocessing plays a key role in improving classification performance.

[4] Ahmed et al. (2023) proposed a deep learning framework for real-time traffic sign recognition using CNN architecture. The study concluded that CNN-based models are effective in extracting spatial features and are suitable for integration into autonomous driving systems.

[5] Patel et al. (2023) presented a traffic sign recognition system using deep neural networks trained on benchmark datasets. Their experimental results revealed that deep learning models achieve better generalization and higher accuracy compared to conventional machine learning methods.

[6] Rao et al. (2024) explored the use of lightweight CNN architectures for traffic sign recognition to reduce computational complexity. Their work emphasized that optimized CNN models can maintain high accuracy while being suitable for real-time and embedded applications.

Chapter 3

SYSTEM REQUIREMENTS

The successful implementation of the Traffic Sign Recognition system requires appropriate hardware and software resources to support model training, testing, and deployment. A system with a reliable processor, sufficient memory, and adequate storage is necessary to handle image data and deep learning computations. Python is used as the programming language along with deep learning libraries such as TensorFlow and Keras to build and train the CNN model. Supporting libraries are used for data processing, evaluation, and visualization, ensuring efficient and accurate system performance.

3.1 Software Requirements

- Operating System: Windows / Linux
- Programming Language: Python 3.x
- Development Environment: VS Code / Jupyter Notebook
- Libraries & Frameworks:
  - TensorFlow
  - Keras
  - NumPy
  - Pandas
  - Matplotlib

3.2 Hardware Requirements

- Processor: Intel i5 or higher
- RAM: Minimum 8 GB (16 GB recommended)
- Storage: 20 GB free disk space
- System Type: 64-bit system
- Optional: GPU (NVIDIA CUDA-enabled) for faster model training.

3.3 Dataset Requirements

- Dataset: German Traffic Sign Recognition Benchmark (GTSRB)
- Image Size: 30 × 30 pixels
- Number of Classes: 43

3.4 Other Requirements

- Internet connection (for dataset download and package installation)
- Basic knowledge of Python and Machine Learning

Chapter 4

DESCRIPTION OF MODULES

4.1 Data Preprocessing Module

The data preprocessing module is responsible for preparing traffic sign images for model training. In this module, images are loaded from the dataset and resized to a fixed dimension of 30×30 pixels to ensure uniformity. Pixel values are normalized to improve learning efficiency. The traffic sign class labels are converted into one-hot encoded vectors to make them suitable for multi-class classification. The dataset is then divided into training and testing sets for model evaluation.

4.2 CNN Model Building Module

This module focuses on constructing the neural network architecture used for traffic sign recognition. A Convolutional Neural Network (CNN) is designed with multiple convolutional layers to extract important image features, followed by pooling layers for dimensionality reduction and dropout layers to prevent overfitting. Fully connected dense layers are added at the end to perform classification. The softmax activation function is used in the output layer to classify images into 43 traffic sign categories.

4.3 Model Training Module

The model training module trains the CNN using the preprocessed training data. During training, the model learns patterns and features from traffic sign images by adjusting its weights using the optimization algorithm. The training process is carried out for a fixed number of epochs and batch size. Training and validation accuracy and loss values are monitored to analyze the learning behavior of the model and ensure effective convergence.

4.4 Model Evaluation Module

In this module, the trained model is evaluated using test data that was not seen during training. The model's performance is measured using metrics such as accuracy, precision, recall, F1-score, and confusion matrix. These metrics help assess the reliability and effectiveness of the model in classifying traffic signs and identifying misclassifications.

4.5 Visualization Module

The visualization module presents the results of the model in graphical form for better analysis and interpretation. Graphs showing training and validation accuracy and loss are generated to understand model performance over epochs. A confusion matrix is displayed to visualize classification results. These visual outputs help in evaluating model consistency and identifying improvement areas.

4.6 Prediction Module

The prediction module is responsible for identifying the traffic sign from a new input image. After the model is trained, it accepts unseen traffic sign images, preprocesses them, and passes them through the trained CNN model. The model predicts the class label with the highest probability, and the corresponding traffic sign name is displayed to the user. This module demonstrates the practical applicability of the system in real-time scenarios.

4.7 Data Splitting Module

The data splitting module divides the dataset into training and testing subsets. Typically, 80% of the data is used for training the model, while the remaining 20% is reserved for testing. This separation ensures that the model is evaluated on unseen data, helping to assess its generalization capability and prevent overfitting.

4.8 Feature Scaling Module

The feature scaling module improves the learning efficiency of the model by normalizing the image pixel values. Pixel intensities are scaled to a standard range, usually between 0 and 1. This helps the CNN converge faster during training and ensures numerical stability while processing image data.

4.9 Output Interpretation Module

The output interpretation module presents the prediction results in a user-understandable format. It maps the predicted class index to the corresponding traffic sign label and displays it through text or GUI output. Additionally, performance metrics such as accuracy, precision, recall, F1-score, and confusion matrix are used to interpret the overall effectiveness and reliability of the system.

Chapter 5

IMPLEMENTATION

The Traffic Sign Recognition system is implemented using Python and deep learning libraries such as TensorFlow and Keras. The implementation follows a modular approach, ensuring clarity, scalability, and ease of maintenance. The system begins by loading the traffic sign dataset and performing necessary preprocessing steps, which include resizing images to a fixed size, normalizing pixel values, and converting class labels into one-hot encoded format.

After preprocessing, the dataset is divided into training and testing sets. The Convolutional Neural Network (CNN) model is then constructed using multiple convolutional layers, pooling layers, dropout layers, and fully connected dense layers. The ReLU activation function is used in hidden layers to speed up learning, while the Softmax activation function is applied in the output layer to perform multi-class classification.

The model is trained using the training dataset over a fixed number of epochs with an appropriate batch size. During training, the optimizer updates the weights to minimize classification error. Validation accuracy and loss are monitored at each epoch to observe the learning behavior and avoid overfitting. Once the training is complete, the trained model is saved for future use and evaluation.

The trained model is then evaluated using the test dataset. Performance metrics such as accuracy, precision, recall, F1-score, and confusion matrix are calculated to assess the effectiveness of the system. Graphical visualizations of training and validation accuracy and loss are generated to support performance analysis.

Finally, a graphical user interface (GUI) is implemented using Tkinter. The GUI allows users to upload traffic sign images and receive real-time predictions from the trained model. This implementation demonstrates the practical usability of the system for real-world traffic sign recognition applications.

Chapter 6

SYSTEM ARCHITECTURE

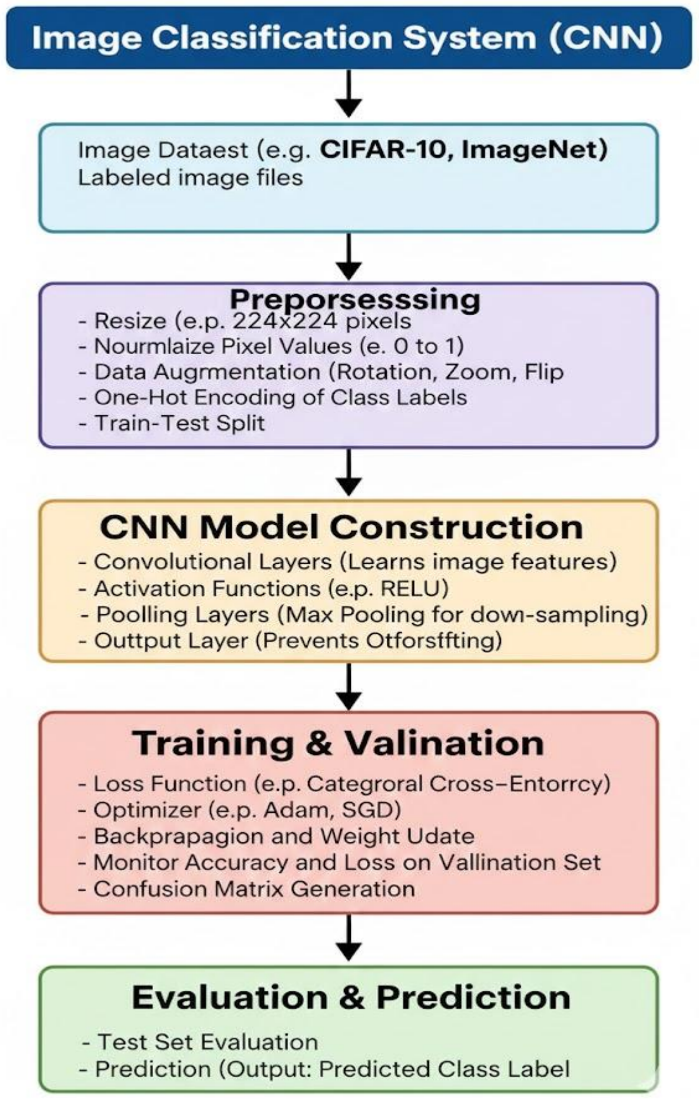


Image Classification System (CNN)
Image Dataest (e.g. CIFAR-10, ImageNet)
Labeled image files
Preporsesssing
- Resize (e.p. 224x224 pixels
- Nourmlaize Pixel Values (e. 0 to 1)
- Data Augrmentation (Rotation, Zoom, Flip
- One-Hot Encoding of Class Labels
- Train-Test Split
CNN Model Construction
- Convolutional Layers (Learns image features)
- Activation Functions (e.p. RELU)
- Poolling Layers (Max Pooling for down-sampling)
- Outtput Layer (Prevents Otforsffting)
Training & Valination
- Loss Function (e.p. Categroral Cross–Entorrcy)
- Optimizer (e.p. Adam, SGD)
- Backprapagion and Weight Udate
- Monitor Accuracy and Loss on Vallination Set
- Confusion Matrix Generation
Evaluation & Prediction
- Test Set Evaluation
- Prediction (Output: Predicted Class Label

Fig 6.1 System Flow digram

Input

The system takes traffic sign images as input. These images may be loaded from the training dataset or provided by the user through a graphical user interface. The input images represent different traffic sign categories such as speed limits, warnings, and mandatory signs.

Preprocessing

In this stage, the input images are prepared for model training and prediction. Images are resized to a fixed size (30×30 pixels) to maintain uniformity. Pixel values are normalized to improve learning efficiency, and class labels are converted into one-hot encoded format for multi-class classification**.**

CNN Model Construction

A Convolutional Neural Network (CNN), which is a type of Artificial Neural Network for image data, is constructed. The model consists of convolutional layers for feature extraction, pooling layers for dimensionality reduction, dropout layers to prevent overfitting, and dense layers for classification. A softmax layer is used at the output to classify traffic signs into multiple categories.

Training

The CNN model is trained using the preprocessed training dataset. During training, the model learns important patterns and features from traffic sign images by minimizing loss using an optimization algorithm. Training and validation accuracy and loss are monitored to evaluate learning performance.

Visualization and Prediction

Graphs showing accuracy and loss are generated to analyze model performance. A confusion matrix and evaluation metrics are used to assess classification quality. Finally, the trained model predicts the class of new traffic sign images, and the predicted traffic sign is displayed to the user through a graphical interface.

Chapter 7

CODE IMPLEMENTATION

Input: GTSRB (German Traffic Sign Recognition Benchmark) Dataset

Output: Predicted Traffic Sign Class (1–43) and performance metrics

1. Start

2. Load Dataset
   2.1 Load traffic sign images from the GTSRB dataset folders.
   2.2 Extract images and corresponding class labels.
   2.3 Store image data in feature set X and labels in target vector y.

3. Preprocess Data
   3.1 Resize all images to 30 × 30 pixels.
   3.2 Normalize pixel values between 0 and 1.
   3.3 Convert class labels into one-hot encoded format.
   3.4 Split the dataset into training and testing sets using:
   • test_size = 0.2
   • random_state = 42

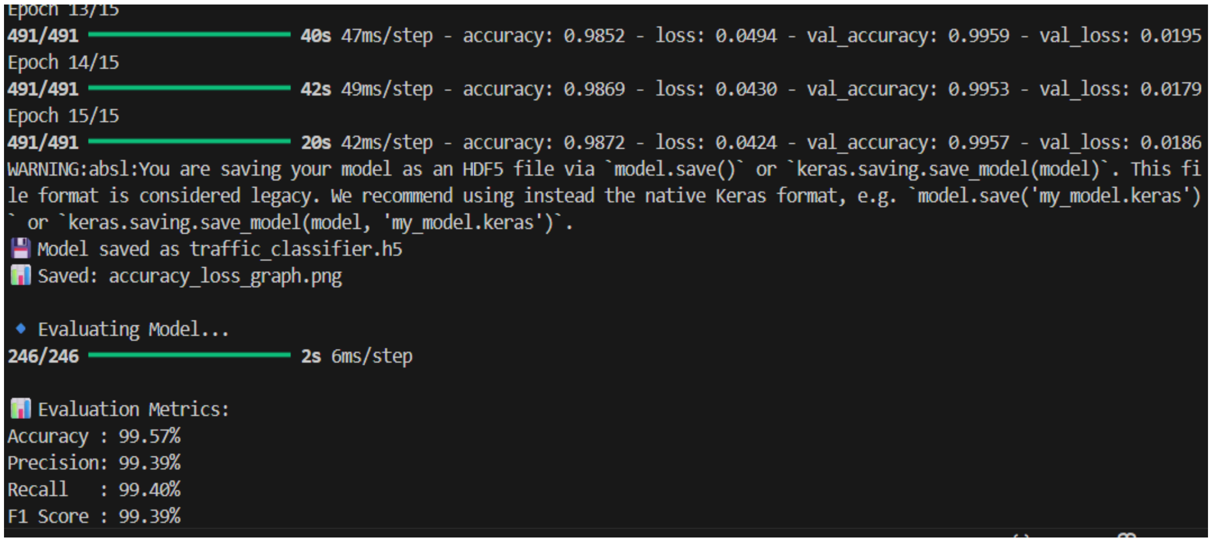
4. Build CNN Model
   4.1 Initialize a Sequential CNN model.
   4.2 Add convolutional layers with ReLU activation.
   4.3 Add max-pooling layers for dimensionality reduction.
   4.4 Add dropout layers to prevent overfitting.
   4.5 Add fully connected dense layers.
   4.6 Add output layer with Softmax activation for multi-class classification.

5. Compile Model
   5.1 Set optimizer as Adam.
   5.2 Set loss function as Categorical Cross-Entropy.
   5.3 Set evaluation metric as Accuracy.

```
6. Train Model
    6.1 Train the CNN model using training data with:
        • Epochs = 15
        • Batch size = 64
        • Validation data = test set
    6.2 Store training history including accuracy and loss values.

7. Test Model
    7.1 Predict class probabilities for test images.
    7.2 Convert predicted probabilities into class labels using argmax.

8. Evaluate Performance
    8.1 Calculate test accuracy using accuracy_score.
    8.2 Generate classification report consisting of precision, recall, and F1-score.
    8.3 Compute confusion matrix.

9. Visualize Results
    9.1 Plot training and validation accuracy curves.
    9.2 Plot training and validation loss curves.
    9.3 Display confusion matrix as a heatmap.

10. Prediction
    10.1 Load the trained CNN model.
    10.2 Accept a new traffic sign image as input.
    10.3 Predict and display the traffic sign class.

11. End
```

Chapter 8

RESULT



```
Epoch 13/15
491/491 ━━━━━━━━━━━━━━━━━━━━ 40s 47ms/step - accuracy: 0.9852 - loss: 0.0494 - val_accuracy: 0.9959 - val_loss: 0.0195
Epoch 14/15
491/491 ━━━━━━━━━━━━━━━━━━━━ 42s 49ms/step - accuracy: 0.9869 - loss: 0.0430 - val_accuracy: 0.9953 - val_loss: 0.0179
Epoch 15/15
491/491 ━━━━━━━━━━━━━━━━━━━━ 20s 42ms/step - accuracy: 0.9872 - loss: 0.0424 - val_accuracy: 0.9957 - val_loss: 0.0186
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This fi
le format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')
` or `keras.saving.save_model(model, 'my_model.keras')`.
💾 Model saved as traffic_classifier.h5
📊 Saved: accuracy_loss_graph.png

🔹 Evaluating Model...
246/246 ━━━━━━━━━━━━━━━━━━━━ 2s 6ms/step

📊 Evaluation Metrics:
Accuracy : 99.57%
Precision: 99.39%
Recall   : 99.40%
F1 Score : 99.39%
```

Fig 8.1 Result of model training

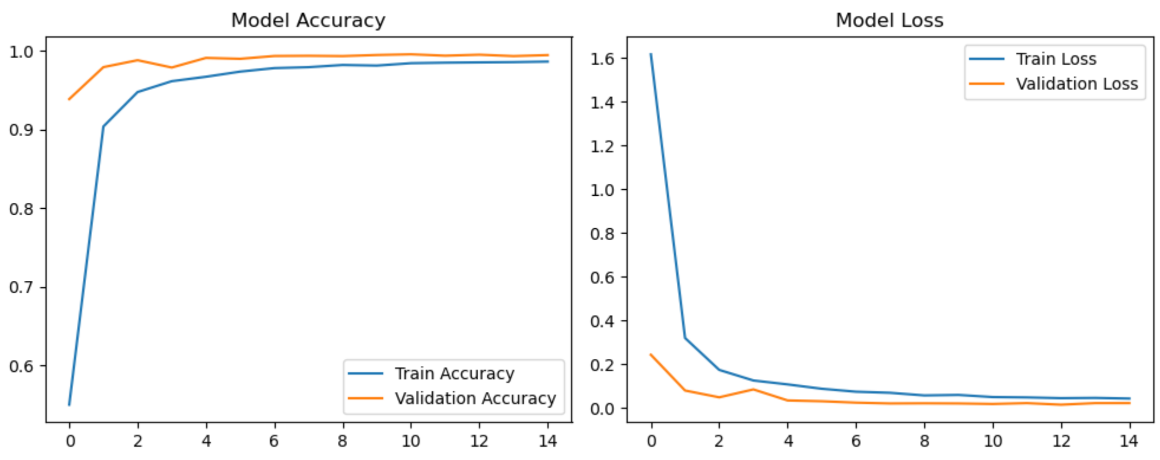


Fig 8.2:Model accuracy and Model loss

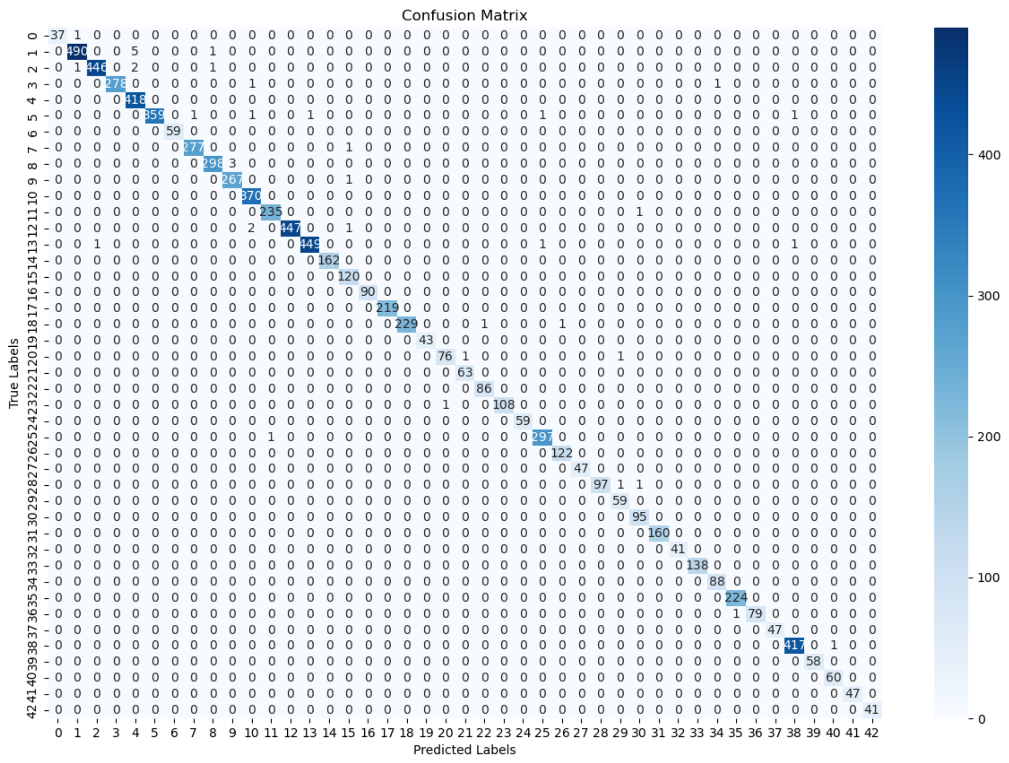


Fig 8.3 Confusion Matrix

Conclusion

This project successfully demonstrates the use of deep learning techniques for traffic sign recognition using a Convolutional Neural Network (CNN). By training the model on the GTSRB dataset, the system is able to accurately identify and classify traffic signs into multiple categories. Image preprocessing techniques such as resizing, normalization, and one-hot encoding contributed to improved learning efficiency and model performance.

The experimental results show high accuracy along with strong precision, recall, and F1-score values, indicating that the proposed system is reliable and effective. Visualizations such as accuracy and loss graphs, as well as the confusion matrix, provide clear insight into the model's performance.

The integration of a graphical user interface further enhances the practicality of the system by allowing users to classify traffic sign images in real time. Overall, this project highlights the importance and potential of CNN-based approaches in intelligent transportation systems and autonomous driving applications, where accurate traffic sign recognition plays a vital role in road safety.

References

[1] **Zhang et al. (2021).** Deep learning–based traffic sign recognition using Convolutional Neural Networks, demonstrating superior performance compared to traditional image processing techniques under varying illumination and complex backgrounds.

[2] **Li et al. (2022).** Improved CNN model for traffic sign classification with data augmentation, highlighting enhanced robustness and reduced overfitting on unseen traffic sign images.

[3] **Kumar and Singh (2022).** CNN-based traffic sign detection emphasizing the importance of image preprocessing techniques such as resizing and normalization for improved classification accuracy.

[4] **Ahmed et al. (2023).** Deep learning framework for real-time traffic sign recognition using CNN architecture, proving its effectiveness for integration into autonomous driving systems.

[5] **Patel et al. (2023).** Traffic sign recognition using deep neural networks trained on benchmark datasets, showing better generalization and higher accuracy than conventional machine learning models.

[6] **Rao et al. (2024).** Lightweight CNN architectures for traffic sign recognition, focusing on reducing computational complexity while maintaining high accuracy for real-time and embedded applications.