Sign Sense System Using

Convolutional Neural Network Algorithm

A major project review submitted to

Rajiv Gandhi University of Knowledge Technologies

SRIKAKULAM

In partial fulfillment of the requirements for the

Award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

4th year B. Tech 2nd semester

Karothu.Lavanya (S180198)

Meesala.Manikanta (S180981)

G.Jyosthna (S180845)

Under the Esteemed Guidance of

Mr.T.Anil Kumar M. Tech, (Ph.D)

 Assistant Professor

Department of Computer Science and Engineering,

**RGUKT-SRIKAKULAM, ETCHERLA.**



**RAJIV GANDHI UNIVERSITY OF KNOWLEDGE**

**TECHNOLOGIES**

**------------------------------------------------------------------------------------**

We declared that this thesis work titled “Sign Sense” Traffic Sign Recognition System ,by using Traffic signs Detecting the sign images using webcam is carried out by me during the year 2023-24 in partial fulfillment of the requirements for the Major Project under the supervisor and guidance of Mr.T Anil Kumar, Assistant professor department of Computer Science and Engineering.

We further declare that this dissertation has not been submitted elsewhere for any Degree. The matter embodied in this dissertation report has not been submitted elsewhere for any other degree. Furthermore, the technical details furnished in various chapters of this are purely relevant to the above project and there is no deviation from the theoretical point of view for design, development and implementation

Date:

Place: Srikakulam

**Karothu Lavanya– S180198**

**Meesala Manikanta-S180981**

**G.Jyosthna - S180845**



**RAJIV GANDHI UNIVERSITY OF KNOWLEDGE**

**TECHNOLOGIES**

**----------------------------------------------------------------------------------**

**CERTIFICATION OF COMPLETION**

This is to certify that the thesis work titled “**Sign Sense System Using Convolutional Neural** **Network Algorithm**”,was successfully completed by **K.LAVANYA (S180198),** **M.MANIKANTA (S180981),G JYOSTHNA(S180845)**. In partial fulfilment of the requirements for the Major Project in Computer Science and Engineering of **Rajiv Gandhi** **University of Knowledge Technologies** under my guidance and output of the work carried out is satisfactory.

----------------------------------------------- --------------------------------------------

**Mr.T.ANIL Kumar M.Tech,(Ph.D)** **Mrs.Ch.Lakshmi Bala M.Tech,(Ph.D)**

Assistant Professor, Assistant Professor,

Project Guide, Head of the Department,

Department of CSE, Department of CSE,

RGUKT-SKLM. RGUKT- SKLM.

**CERTIFICATE OF EXAMINATION**

This is to certify that the work entitled, “**SIGN SENSE SYSTEM USING CONVOLUTIONAL NEURAL NETWORK ALGORITHM”** is work of **KAROTHU LAVANYA(S180198), MEESALA MANIKANTA(S180918), G JYOSTHNA(S180845) ”** and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in final year of **Bachelor of Technology** for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusion drawn, as recorded in this thesis. It only signifies the acceptance of this thesis for the purpose for which it has been submitted.

**PROJECT GUIDE PROJECT COORDINATOR HEAD OF THE DEPARTMENT**

Mr.T.Anil Kumar Mr.S.Sateesh kumar Mrs.Lakshmi Bala

Assistant Professor Project co-ordinator Head of Department

Department of CSE Department of CSE Department of CSE

RGUKT-Srikakulam RGUKT-Srikakulam RGUKT-Srikakulam

|  |
| --- |
|  |

**ACKNOWLEDGEMENT**

We would like to articulate my profound gratitude and indebtedness to my project guide Professor. **Mr. T Anil Kumar, Assistant Professor**, who has always been a constant motivation and guiding factor throughout the project time. It has been a great pleasure for me to get an opportunity to work under his guidance and complete the thesis work successfully.

We wish to extend my sincere thanks to our head of the department **Mrs. Ch. Lakshmi Bala**, **Assistant professor,Head of the computer science and Engineering Department**,for his constant encouragement throughout the project.

We am also grateful to other members of the department without their support my work would have been carried out so successfully.

We thank one and all who have rendered help to me directly or indirectly in the completion of my thesis work.

**Project Associate**

**Karothu Lavanya– S180198**

**Meesala Manikanta– S180981**

**G.Jyosthna - S180845**

**ABSTRACT**

In today's world, almost everything we do has been simplified by Automated tasks. In an attempt to focus on the road while driving, drivers often miss out on signs on the side of the road, which could be dangerous for them and for the people around them. This problem can be avoided if there was an efficient way to notify the driver without having them to shift their focus. Traffic Sign Detection and Recognition plays an important role here by detecting and recognizing a sign, thus notifying the driver of any upcoming signs. This not only ensures road safety, but also allows the driver to be at little more ease while driving on tricky or new roads. Another commonly faced problem is not being able to understand the meaning of the sign. With the help of this Advanced Driver Assistance Systems application, drivers will no longer face the problem of understanding what the sign says. In this paper, we propose a method for Traffic Sign Detection and Recognition using image processing for the detection of a sign and an ensemble of Convolutional Neural Networks (CNN) for the recognition of the sign. CNNs have a high recognition rate, thus making it desirable to use for implementing various computer vision tasks. Tensor Flow is used for the implementation of the CNN.

**Keywords** – Convolution neural network, Feature extraction, Road accidents, Traffic sign recognition

Contents

Contents

[**CHAPTER-1** 1](#_Toc163596397)

[1.1. Introduction 1](#_Toc163596398)

[1.2 Problem Statement 2](#_Toc163596399)

[1.3 Introduction to Convolutional Neural Networks (CNN) 3](#_Toc163596400)

[1.4 Aims and Objectives of the Research 4](#_Toc163596401)

[1.5 Goals 4](#_Toc163596402)

[1.6 Scope: 5](#_Toc163596403)

[1.7 Applications: 6](#_Toc163596404)

[1.8 Limitations: 6](#_Toc163596405)

[**CHAPTER-2** 6](#_Toc163596406)

[Literature Survey 6](#_Toc163596407)

[2.1 Collect Information 6](#_Toc163596408)

[2.2 Study: 7](#_Toc163596409)

[2.3 Benefits 8](#_Toc163596410)

[2.4 Summary: 8](#_Toc163596411)

[**CHAPTER-3** 9](#_Toc163596412)

[Analysis 9](#_Toc163596413)

[3.1 Existing system 9](#_Toc163596414)

[3.2 Disadvantages 9](#_Toc163596415)

[3.3 Overview of the Proposed Approach 10](#_Toc163596416)

[3.4 Advantages of proposed system 11](#_Toc163596417)

[3.5 System Requirements 11](#_Toc163596418)

[**CHAPTER-4** 12](#_Toc163596419)

[System Design 12](#_Toc163596420)

[4.1 Design of the system 12](#_Toc163596421)

[4.1.4 DFD Diagram: 16](#_Toc163596422)

[**CHAPTER-5** 18](#_Toc163596423)

[System implementation using CNN 18](#_Toc163596424)

[5.1 Flow Chart For Traffic Sign Detection Model 18](#_Toc163596425)

[5.2 Code For Training the model 20](#_Toc163596426)

[5.3 Testing code: 25](#_Toc163596427)

[**CHAPTER 6** 30](#_Toc163596428)

[System Testing 30](#_Toc163596429)

[**CHAPTER 7** 33](#_Toc163596430)

[7. Conclusion 33](#_Toc163596431)

[**CHAPTER 8** 34](#_Toc163596432)

[8. Future Enhancement 34](#_Toc163596433)

[**CHAPTER 9** 35](#_Toc163596434)

[REFERENCES 35](#_Toc163596435)

# **CHAPTER-****1**

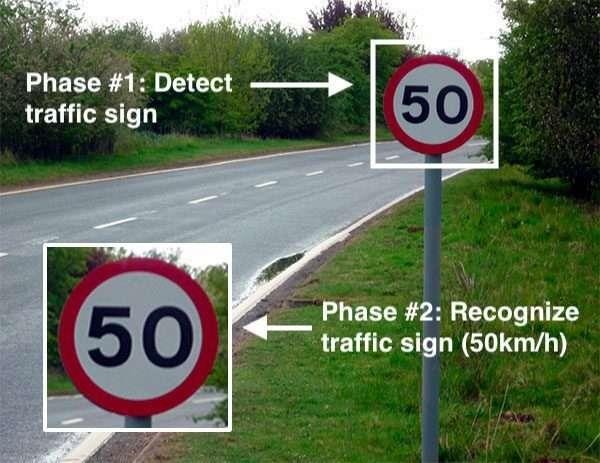
### 1.1. Introduction

In this era of Artificial Intelligence, humans are becoming more dependent on technology. With the enhanced technology, multinational companies like Google, Tesla, Uber, Ford, Audi, Toyota, Mercedes-Benz, and many more are working on automating vehicles. They are trying to make more accurate autonomous or driverless vehicles. You all might know about self-driving cars, where the vehicle itself behaves like a driver and does not need any human

guidance to run on the road. This is not wrong to think about the safety aspects—a chance of significant accidents from machines. But no machines are more accurate than humans. Researchers are running many algorithms to ensure 100% road safety and accuracy. One such algorithm is Traffic Sign Recognition that we talk about in this project.

When you go on the road, you see various traffic signs like traffic signals, turn left or right, speed limits, no passing of heavy vehicles, no entry, children crossing, etc., that you need to follow for a safe drive. Likewise, autonomous vehicles also have to interpret these signs and make decisions to achieve accuracy. The methodology of recognizing which class a traffic sign belongs to is called Traffic signs classification.

In this Deep Learning project, we will build a model for the classification of traffic signs available in the image into many categories using a convolutional neural network(CNN) and Keras library.



### 1.2 Problem Statement

The performance of the traffic sign recognition systems is improved by using a powerful neural network approach called Convolutional Neural Networks (CNN) which acts as a powerful tool to classify and recognize the traffic signs. In this paper, the machine learning model is designed and trained to classify and recognize the traffic signs from the dataset provided by the German Traffic Sign Benchmark (GTSB). The coding of this system is done in python language.



### 1.3 Introduction to Convolutional Neural Networks (CNN)

CNN is one of the neural network models for deep learning, which has three specific characteristics, firstly it takes locally connected neurons, secondly, it calculates shared weight and finally gives the spatial or temporal sub-sampling. Generally, CNN is compromised of two main parts (see Fig.1.2.1). The first contains alternating Convolutional and second is the maxpooling layers. The input of each layer is just the output of its previous layer. As a result, this forms a hierarchical feature extractor that maps the original input images into feature vectors. Then the second part classifies the extracted features vectors, that is, the fully-connected layers, which is a typical feed-forward neural network.

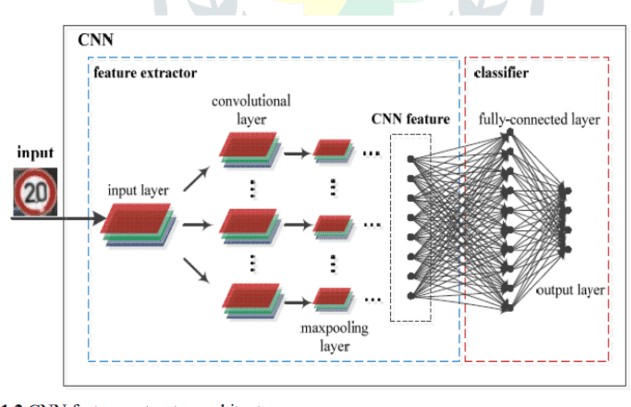


Fig. 1.2 CNN feature extractor architecture.

### 1.4 Aims and Objectives of the Research

The overall aim is to develop a system that can be used for traffic sign inventory. This system can assist local or national authorities in the task of maintaining and updating their road and traffic signs by automatically detecting and classifying one or more traffic signs from a complex scene (like the one shown in Figure 1.2) when captured by a camera from a vehicle. The main strategy is to find the right combination of colours in the scene so that one colour is located inside the convex hull of another colour and combine this with the right shape. If a candidate is found, the system tries to classify the object according to the rimpictogram combination and give the result of this classification.

#### The objectives are thus:

1.To understand the properties of road and traffic signs and their implications for image processing for the recognition task.

2.To understand colour, colour spaces and colour space conversion.

3.To develop robust colour segmentation algorithms that can be used in a wide range of environmental conditions.

4.To develop a recogniser that is invariant to in-plane transformations such as translation, rotation, and scaling based on invariant shape measures.

5.To identify the most appropriate approach for feature extraction from road signs.

6.To develop an appropriate road sign classification algorithm.

7.To evaluate the performance of the aforementioned methods for robustness under different conditions of weather, lighting geometry, and sign.

### 1.5 Goals

A traffic sign recognition system using Convolutional Neural Networks (CNNs) aims to automatically detect and classify traffic signs from images or video feeds captured by cameras mounted on vehicles or at various locations on the road. The primary goals of such a system include:

* **Detection:** The system should accurately locate and identify traffic signs within images or video frames, regardless of factors such as lighting conditions, weather, and occlusions.
* **Classification:** Once detected, the system should correctly classify the type of traffic sign present in the image. This includes recognizing signs for speed limits, stop signs, yield signs, directional arrows, pedestrian crossings, etc.
* **Robustness:**The system should be robust to variations in the appearance of traffic signs due to factors such as occlusion, rotation, scale, weather conditions, variations in lighting, and degradation of sign quality over time.
* **Generalization:** The system should generalize well across different datasets and environments, meaning it should perform effectively on traffic signs from various regions, countries, and regulatory standards.
* **Accuracy:**The system should achieve high accuracy in both detection and classification tasks to minimize false positives and negatives, ensuring reliable performance in real-world scenarios.
* **Scalability:** The system should be scalable to handle large volumes of data and potentially support multiple cameras or sensors simultaneously.

By achieving these goals, a traffic sign recognition system using CNNs can significantly contribute to improving road safety, facilitating autonomous driving, enhancing traffic management, and providing valuable insights for transportation planners and policymakers.

### 1.6 Scope:

The scope of a traffic sign recognition system using Convolutional Neural Networks (CNNs) encompasses various aspects related to its design, development, deployment, and application. Here's a breakdown of the key areas within the scope of such a system.

* **Image Acquisition:** This involves capturing images or video frames using cameras mounted on vehicles or at fixed locations along roads.
* **Preprocessing:** Steps may include image resizing, normalization, noise reduction, and augmentation techniques to enhance the quality and variability of the input data.
* **Dataset collection:** Gathering a comprehensive dataset of annotated images containing various types of traffic signs is crucial for training and evaluating the CNN models.
* **Model Architecture-design:** Designing the CNN architecture involves selecting appropriate network structures, such as convolutional layers, pooling layers, and fully connected layers, to effectively learn spatial hierarchies and features from input images.
* **Training:** Training the CNN models involves feeding labeled images into the network and adjusting the model parameters to minimize prediction errors.
* **Evaluation and validation:** Assessing the performance of trained models involves evaluating their accuracy, precision, recall, and other metrics on validation datasets.
* **Deployment:** Deploying the trained CNN models into real-world applications involves integrating them into software systems or embedded devices.

By addressing these aspects within its scope, a traffic sign recognition system using CNNs can contribute to enhancing road safety, optimizing traffic flow, and enabling more efficient and intelligent transportation systems.

### 1.7 Applications:

A traffic sign recognition system using Convolutional Neural Networks (CNNs) has a wide range of applications across various domains, including transportation, automotive, urban planning, and public safety. Some of the key applications include:

* **Advance driver assistance system:** Integrating traffic sign recognition into ADAS enhances vehicle safety by providing real-time alerts to drivers about speed limits, stop signs, yield signs, and other relevant traffic regulations.
* **Autonomous vehcles:**Traffic sign recognition is crucial for autonomous vehicles to navigate safely and follow traffic laws..
* **Smart cities and urban planning:** Implementing traffic sign recognition systems in smart city initiatives can help optimize traffic flow, reduce congestion, and enhance urban mobility. By analyzing traffic sign data, city planners can make informed decisions about road infrastructure, traffic signal timings, and public transportation routes.
* **Traffic and monitoring management:** Traffic sign recognition can be used for monitoring traffic conditions in real-time and managing traffic flow more efficiently.

### Limitations:

* **Variability in real world conditions**: Traffic signs can exhibit considerable variability in appearance due to factors such as lighting conditions, weather conditions (e.g., rain, snow), occlusions .
* **Data imbalance:**Datasets for training traffic sign recognition models may suffer from class imbalance, where certain types of signs are overrepresented, while others are underrepresented..
* **Limited to recognization of uncommon signs:**CNN-based models may struggle to recognize uncommon or rare traffic signs that are not well represented in the training data.
* **Performance degradation of challenging conditions:**CNN models may experience performance degradation in challenging conditions such as low lighting, heavy rain, fog, or adverse weather conditions..

# **CHAPTER-2**

## Literature Survey

### 2.1 Collect Information

We have taken the information from the other sources like IEEE papers, and you tube reference videos and respected faculty members and my friends.



### 2.2 Study:

Key Features in traffic sign recognition:

A literature survey on traffic sign recognition project would involve reviewing existing research on the topic, including methods, techniques, and algorithms used in traffic sign recognition. Some important areas to consider in the survey may include

* Object detection and recognition techniques: This involves studying different techniques used for object detection and recognition in traffic sign recognition, such as Haar cascades, HOG features, and deep learning techniques.
* Feature extraction techniques: Different feature extraction techniques such as color-based features, shape-based features, texture-based features, and combination features can be reviewed.
* Classification algorithms: This involves studying different classification algorithms such as SVM, KNN, Naive Bayes, and neural network-based approaches, which are commonly used in traffic sign recognition.
* Datasets: A survey of traffic sign recognition would require the evaluation of various datasets used in traffic sign recognition, such as GTSRB, BelgiumTS, and LISA.
* Challenges: Understanding the challenges faced in traffic sign recognition such as variations in lighting conditions, occlusion, and background complexity.
* Recent research: The survey should also include recent research in the field such as the use of deep learning, data augmentation, and transfer learning in traffic sign recognition.
* Performance evaluation metrics: It is important to study different evaluation metrics used to measure the performance of traffic sign recognition algorithms, such as accuracy, precision, recall, F1 score, and mean average precision (mAP).

### 2.3 Benefits

There are several benefits of traffic sign recognition projects, some of which include:

* Improved road safety
* Increased efficiency:
* Enhanced driver experience
* Cost savings
* Accessibility

### 2.4 Summary:

* + - Traffic sign recognition (TSR) project involves developing a system that can detect, recognize, and interpret traffic signs present on the road. The goal of TSR is to improve road safety, increase traffic efficiency, and enhance the driver experience.
    - TSR systems use computer vision techniques, such as image processing, object detection, and machine learning algorithms, to detect and recognize traffic signs accurately. The system can detect various types of traffic signs, such as speed limits, stop signs, pedestrian crossings, and no entry signs.
    - The system works by capturing real-time images or videos of the road and processing them using computer vision algorithms. The algorithms identify the traffic signs present in the image, extract relevant features, and classify them into different categories.
    - TSR systems have several benefits, including improved road safety, increased efficiency, enhanced driver experience, cost savings, and increased accessibility for drivers with disabilities.
    - Overall, traffic sign recognition is an important area of research, and the development of accurate and reliable TSR systems has the potential to make a significant impact on road safety and traffic management.

# **CHAPTER-3**

## Analysis

### 

### 3.1 Existing system

The existing system of traffic sign recognition (TSR) includes various techniques and algorithms to detect and recognize traffic signs accurately. Here are some of the common techniques used in the existing TSR systems:

* Image processing techniques: Image processing techniques such as filtering, edge detection, and segmentation are used to enhance the quality of the image and extract relevant features.
* Feature extraction techniques: Different feature extraction techniques such as color-based features, shape-based features, and texture-based features are used to extract relevant features from the image.
* Classification algorithms: Various classification algorithms such as SVM, KNN, Naive Bayes, and neural network-based approaches are used to classify traffic signs into different categories.
* Datasets: Several datasets such as GTSRB, Belgium TS, and LISA are used to train and test the performance of TSR systems.
* Deep learning techniques: In recent years, deep learning techniques such as Convolutional Neural Networks (CNN) have gained popularity in TSR systems due to their ability to learn complex features automatically.

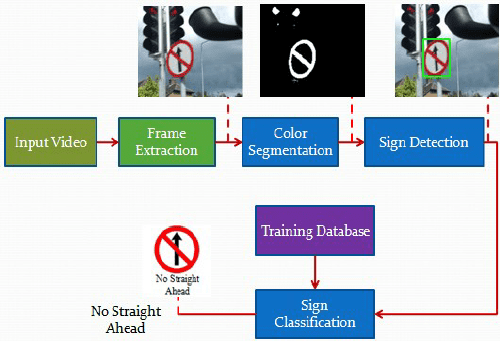
### 3.2 Disadvantages

While the existing TSR systems have achieved high accuracy in detecting and recognizing traffic signs, there are still some limitations and challenges. Some of these limitations and challenges include:

* Variations in lighting conditions: Variations in lighting conditions can affect the performance of TSR systems, as the system may not be able to detect and recognize traffic signs accurately.
  + - Occlusion: Occlusion can occur when traffic signs are partially or fully blocked, which can make it challenging for the system to detect and recognize them accurately.
    - Background complexity: Complex backgrounds can also affect the performance of TSR systems, as the system may not be able to differentiate between the traffic sign and the background.
    - Lack of standardization: Different countries may have different traffic sign designs and colors, which can make it challenging for the system to recognize them accurately.

### 3.3 Overview of the Proposed Approach

The framework we proposed is categorized into three stages: Detection and feature extraction and recognition. The detection stage is just used to find a road sign. At the point when a vehicle is travelling at a specific speed, the camera catches the road sign in nature, and our calculation verifies whether a sign is available in that outline or not available in that perimeter. Distinguishing the traffic sign depends on shape and color. In the feature extraction stage, the proposed calculation characterizes the distinguished road sign. This is accomplished with the assistance of "Convolutional Neural Network" algorithm which classifies the image into sub classes.



### 3.4 Advantages of proposed system

The advantages of a proposed traffic sign recognition (TSR) project system can vary depending on the specific approach and techniques used. However, some potential advantages of a well-designed TSR system may include:

* + - Improved accuracy: A well-designed TSR system can improve the accuracy of traffic sign detection and recognition, which can help to reduce the risk of accidents caused by driver error or failure to notice traffic signs.
    - Faster response time: A TSR system can provide real-time information on traffic signs, allowing drivers to respond quickly and appropriately to changing road conditions.
    - Increased efficiency: By improving traffic flow and reducing congestion, a TSR system can increase the efficiency of transportation networks, resulting in reduced travel times and fuel consumption.
    - Reduced costs: The implementation of a TSR system can potentially reduce the costs associated with accidents and congestion, including medical bills, vehicle repairs, and environmental damage.
    - Scalability: A well-designed TSR system can be easily scaled to work in different geographic regions, as the system can be trained to recognize different types of traffic signs in different countries or regions.

### 3.5 System Requirements

Software Requirements:

* Jupyter note book.
* System camera
* Windows 10 Hardware Requirements:
* RAM: 4GB above
* Hard disk: 500 GB above

# **CHAPTER-4**

## System Design

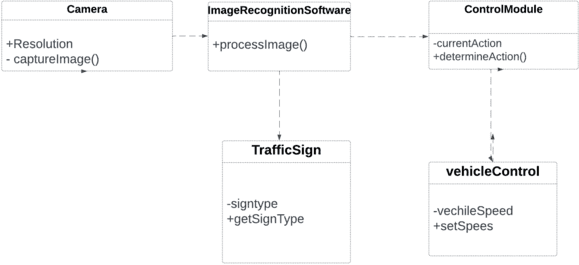
### 

### 4.1 Design of the system

Unified Modelling Language (UML) was created in 1995 by using merging diagramming conventions used by three application development methodologies: OMT by James Rumbaugh, Objector y by Invar Jacobson and the Brooch procedure by using Grady Brooch. Previous to this time, these three amigos, together with a few dozen other practitioners had promoted competing methodologies for systematic program development, each and every with its possess system of diagramming conventions. The methodologies adopted a sort of cookbook sort of pushing a application task via a succession of life cycle stages, culminating with a delivered and documented software.One purpose of UML was once to slash the proliferation of diagramming techniques by way of standardizing on a original modelling language, as a result facilitating verbal exchange between builders. It performed that goal in 1997 when the (international) Object administration team (OMG) adopted it as a commonplace. Some critics don’t forget that UML is a bloated diagramming language written by means of a committee. That said, I do not forget it to be the nice manner to be had today for documenting object-oriented program progress. It has been and is fitting more and more utilized in industry and academia. Rational Rose is a pc Aided program Engineering (CASE) software developed by way of the Rational organization underneath the course of Brooch, Jacobson and Rumbaugh to support application progress using UML. Rational Rose is always complex due to its mission of wholly supporting UML. Furthermore, Rational Rose has countless language extensions to Ada, C++, VB, Java, J2EE, and many others. Rational Rose supports ahead and reverse engineering to and from these langue ages. However, Rational Rose does now not aid some usual design tactics as knowledge drift diagrams and CRC cards, due to the fact that these will not be a part of UML. Considering that Rational Rose has so many capabilities it is a daunting task to master it. Happily, loads can be executed making use of only a small subset of these capabilities. These notes are designed to introduce beginner builders into making productive use of the sort.

#### 4.1.1 Class diagram:

Class diagram in the Unified Modelling Language (UML), is a kind of static structure diagram hat describes the constitution of a process through showing the system's classes, their attributes, and the relationships between the class. The motive of a class diagram is to depict the classes within a model. In an object-oriented software, classes have attributes (member variables), operations (member capabilities) and relation.



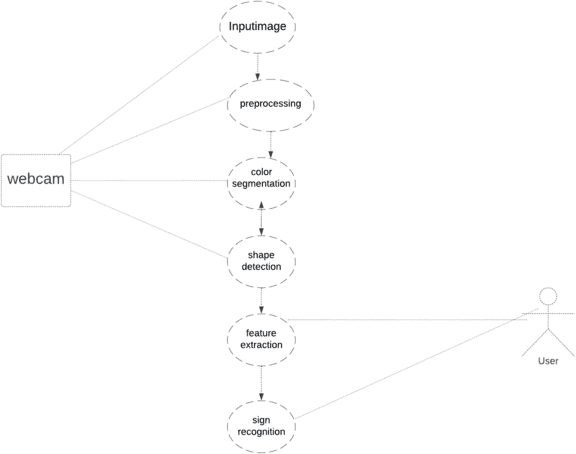
In this class diagram, there are four classes: Camera, Software, ControlModule, TrafficSign, and VehicleControl.

* + - * The Camera class represents the camera that captures an image of the road. It has a resolution attribute and a captureImage() method.
      * The ImageRecognitionSoftware class represents the software that processes the image captured by the camera. It has a processImage() method.
      * The ControlModule class represents the module that determines the appropriate action based on the identified traffic sign. It has a currentAction attribute and a determineAction() method.
      * The TrafficSign class represents the traffic sign that is identified by the system. It has a signType attribute and a getSignType() method.
      * The VehicleControl class represents the vehicle's control system. It has a vehicleSpeed attribute and a setSpeed() method.

Overall, this class diagram represents the key components and their relationships in a traffic sign recognition system.

#### 4.1.2 Use Case Diagram:

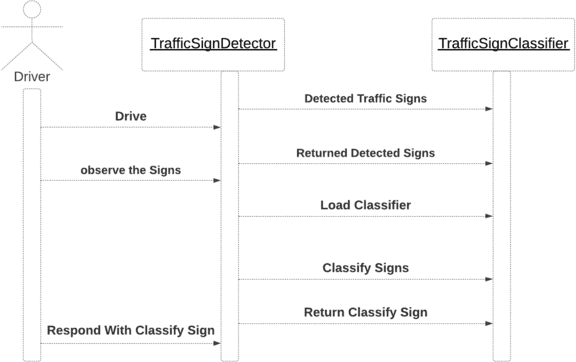
It is a visually representation what happens when actor interacts with system. A use case diagram captures the functional aspects of a system.The system is shown as a rectangle with name of the system inside ,the actor are shown as stick figures, the use case are shown as solid bordered ovals 14odellin with name of the use case and relationships are lines or arrows between actor and use cases. Symbols used in Use case are as follows-



* + - * In this use case diagram, we have three actors: the Driver, the Traffic Sign Detector, and the Traffic Sign Classifier
      * The Driver actor drives a vehicle and observes traffic signs. The use case for the Driver is to drive and observe signs.
      * The Traffic Sign Detector actor detects traffic signs in an image captured by a camera or retrieved from a database. The use case for the Traffic Sign Detector is to detect traffic signs and return the detected signs.
      * The Traffic Sign Classifier actor classifies the detected traffic signs using a machine learning model. The use case for the Traffic Sign Classifier is to load the classifier and classify the detected signs.
      * The use case diagram shows the interactions between the actors and the system, which provides a high-level overview of the functionality of the traffic sign recognition project.

#### 4.1.3 Sequence diagram

A sequence diagram in Unified Modelling Language (UML) is one variety of interaction diagram that suggests how methods operate with one other and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are quite often referred to as event-hint diagrams, event situations, and timing diagrams. A sequence diagram suggests, as parallel vertical traces (lifelines), special systems or objects that are residing at the same time, and, as horizontal arrows, the messages exchanged between them, within the order the place they occur.



* In this sequence diagram, we have three actors: the Driver, the Traffic Sign Detector, and the Traffic Sign Classifier.
* The sequence diagram starts with the Driver driving and observing traffic signs. The Driver then sends a request to the Traffic Sign Detector to detect traffic signs in the observed image. The Traffic Sign Detector analyzes the image and detects the traffic signs. It then returns the detected signs to the Driver.
* The Driver then sends a request to the Traffic Sign Classifier to classify the detected signs. The Traffic Sign Classifier loads the classifier and classifies the signs using the machine learning model. It then returns the classified signs to the Driver.
* Finally, the Driver responds with the classified signs, which completes the sequence diagram.
* This sequence diagram provides a detailed view of the interactions between the actors and the system, which helps to understand the flow of data and processing in the traffic sign recognition system

### 4.1.4 DFD Diagram:

A data flow diagram or bubble chart (DFD) is a graphical representation of the “flow” of data .through an information system, 16odelling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

The primitive symbols used for constructing DFD’s are: Symbols used in DFD

A circle represents a process

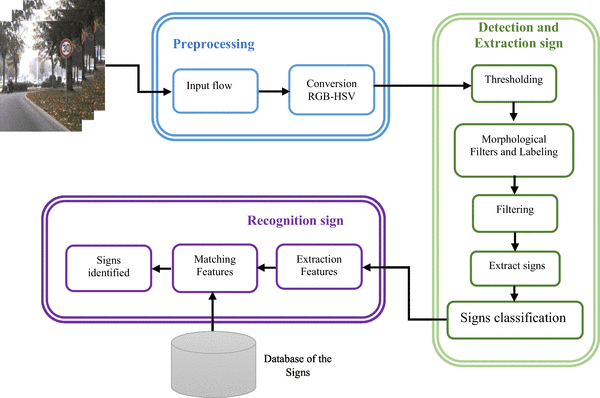
A rectangle represents external entity

A square defines a source or destination of the system data.

An arrow identifies dataflow.

Double line with one end closed indicates data store

Data Flow Diagram Of Traffic Sign Recognition System:



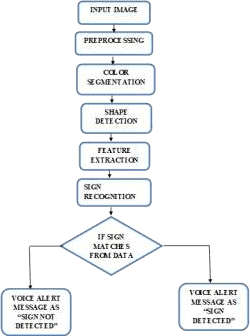
# **CHAPTER-5**

## System implementation using CNN

### 

### 5.1 Flow Chart For Traffic Sign Detection Model

The flow chart below gives details about various steps involved in designing traffic sign detection model.



STEP 1. INPUT IMAGE

Camera which is placed inside the vehicle records a video which is nothing but a series of photos. Usually there are 24 frames per second. These photos go through the processors for the detection of sign. The camera quality should be very high that it should clearly shows the traffic sign from the minimum distance required. Generally, a camera of more than 8MP is required for this process.

STEP 2. PRE-PROCESSING

The image or videos which is scanned is pre-processed through convolutional neural network. The image which has higher resolution is scaled down to small resolution and RGB image is converted into greyscale format so that the image can be easily processed by the “Convolution Neural Networks”. The Neural networks are very similar to the neurons in our brain it senses the image and gives the necessary information to the processor. Below is the figure which shows left or straight sign, it is converted into greyscale and is fed into CNN. It is then sampled and divided into small pieces and the obtained output is compared with the data set which we gave and the corresponding voice output is given. Neural networks are similar to the neurons in our brain it senses the image and gives the necessary information to the processor. The steps which are required for pre-processing in Convolution neural network is lower when compared to other classification algorithms.

STEP 3. (RGB) BASED DETECTION

In a sign the most significant thing is the color. Once the red colour is seen it is understood that it is a traffic sign on the road. This same idea is used for our detection process. Based on whatever the frames captured our algorithm is designed to check for a sign depending on the red colour. From the image captured if a part of it is similar to the threshold values of red colour then it is passed for the further steps to recognize if it is a sign or not. Once the threshold of the red is checked the main sign in the red part is to be recognized.

STEP 4. DETECTION BASED ON SHAPE OF THE SIGN

Based on the previous detection method we calculate the number of edges which is implemented using the algorithm of Douglas-Peucker. We majorly concentrate on two shapes which are circle and triangle because they are most repeated shapes of traffic sign. Once using Douglas-Peucker algorithm the number of edges and area of interest is found. Now if the edges found is equal or greater than six and if the main part satisfies minimum condition then it is considered as a triangle. And if edges are equal and greater than six and moreover if they satisfy minimum condition then the major part of the image is recognized as circle. After the shapes are recognized the next major step is in detecting the box of the bounding. The bounding box is important because the Region of Interest (ROI) is separated from the environment by the bounding box. Usually the box touches circle or the triangle of the main region. In a triangular sign, it consists of two triangles they are outer triangle and inner triangle. The outer triangle just touches the box of bounding and the one which does not touch is the inner triangle.

### 5.2 Code For Training the model

Importing libraies:

import matplotlib.pyplot as plt

from keras.models import Sequential from keras.layers import Dense

from keras.optimizers import Adam

from keras.utils.np\_utils import to\_categorical from keras.layers import Dropout, Flatten

from keras.layers.convolutional import Conv2D, MaxPooling2D import cv2

from sklearn.model\_selection import train\_test\_split import pickle

import os

import pandas as pd import random

from keras.preprocessing.image import ImageDataGenerator import numpy as np

Passing Parameters:

path = r"C:\Users\Y Ramu\Downloads\TRAFFIC\TRAFFIC\myData" labelFile =r'"C:\Users\Y

Ramu\Downloads\TRAFFIC\TRAFFIC\labels.csv"' #file with all names of classes

batch\_size\_val=50 # how many to process together steps\_per\_epoch\_val=2000

epochs\_val=10 imageDimesions = (32,32,3)

testRatio = 0.2 # if 1000 images split will 200 for testing validationRatio = 0.2 # if 1000 images 20% of remaining 800 will be 160 for validation

Importing of the Images:

count = 0 images = [] classNo = []

myList = os.listdir(path)

print("Total Classes Detected:",len(myList)) noOfClasses=len(myList)

print("Importing Classes ")

for x in range (0,len(myList)):

myPicList = os.listdir(path+"/"+str(count)) for y in myPicList:

curImg = cv2.imread(path+"/"+str(count)+"/"+y) images.append(curImg)

classNo.append(count) print(count, end =" ") count +=1

print(" ")

images = np.array(images) ­­classNo = np.array(classNo) output:­­­



­­­­­ Split Data:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(images, classNo, test\_size=testRatio)

X\_train, X\_validation, y\_train, y\_validation = train\_test\_split(X\_train, y\_train, test\_size=validationRatio)

# X\_train = ARRAY OF IMAGES TO TRAIN # y\_train = CORRESPONDING CLASS ID

############################### TO CHECK IF NUMBER OF IMAGES MATCHES TO NUMBER OF LABELS FOR EACH DATA SET

print("Data Shapes")

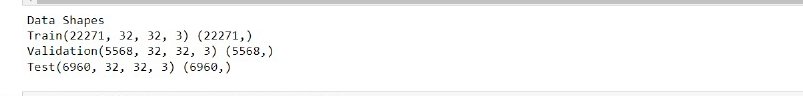
print("Train",end = "");print(X\_train.shape,y\_train.shape) print("Validation",end =

"");print(X\_validation.shape,y\_validation.shape)

print("Test",end = "");print(X\_test.shape,y\_test.shape) assert(X\_train.shape[0]==y\_train.shape[0]), "The number of images in not equal to the number of lables in training set" assert(X\_validation.shape[0]==y\_validation.shape[0]), "The number of images in not equal to the number of lables in validation set" assert(X\_test.shape[0]==y\_test.shape[0]), "The number of images in not equal to the number of lables in test set" assert(X\_train.shape[1:]==(imageDimesions))," The dimesions of the Training images are wrong " assert(X\_validation.shape[1:]==(imageDimesions))," The dimesionas of the Validation images are wrong "

equal to the number of lables in test set" assert(X\_train.shape[1:]==(imageDimesions))," The dimesions of the Training images are wrong " assert(X\_validation.shape[1:]==(imageDimesions))," The dimesionas of the Validation images are wrong "

assert(X\_test.shape[1:]==(imageDimesions))," The dimesionas of the Test images are wrong"



############################### READ CSV FILE

data=pd.read\_csv(r'C:\Users\LAVANYA\Downloads\TRAFFIC\kkk\labels.csv')

print("data shape ",data.shape,type(data))



DISPLAY SOME SAMPLES IMAGES OF ALL THE CLASSES:

num\_of\_samples = [] cols = 5

num\_classes = noOfClasses

fig, axs = plt.subplots(nrows=num\_classes, ncols=cols, figsize=(5, 300)) fig.tight\_layout()

for i in range(cols):

for j,row in data.iterrows(): x\_selected = X\_train[y\_train == j]

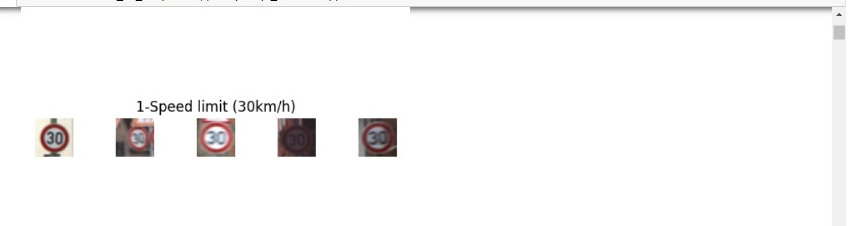
axs[j][i].imshow(x\_selected[random.randint(0, len(x\_selected)- 1),

:, :], cmap=plt.get\_cmap("gray"))

axs[j][i].axis("off") if i == 2:

axs[j][i].set\_title(str(j)+ "-"+row["Name"]) num\_of\_samples.append(len(x\_selected))

Output:

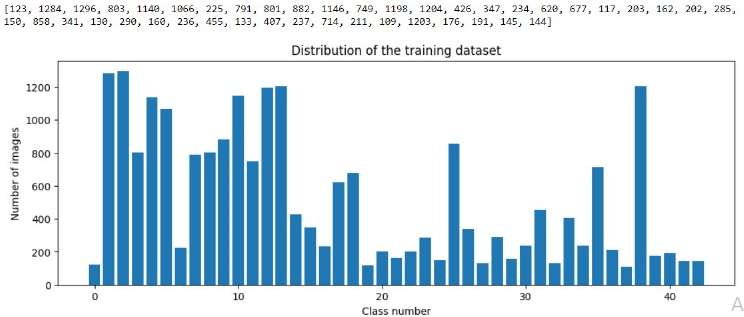


**DISPLAY A BAR CHART SHOWING NO OF SAMPLES FOR EACH CATEGORY:**

print(num\_of\_samples) plt.figure(figsize=(12, 4))

plt.bar(range(0, num\_classes), num\_of\_samples) plt.title("Distribution of the training dataset") plt.xlabel("Class number")

plt.ylabel("Number of images") plt.show()



**PREPROCESSING THE IMAGES:**

def grayscale(img):

img = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) return img

def equalize(img):

img =cv2.equalizeHist(img) return img

def preprocessing(img):

img = grayscale(img) # CONVERT TO GRAYSCALE

img = equalize(img) # STANDARDIZE THE LIGHTING IN AN IMAGE

img = img/255 # TO NORMALIZE VALUES BETWEEN 0 AND 1 INSTEAD OF 0 TO 255

return img

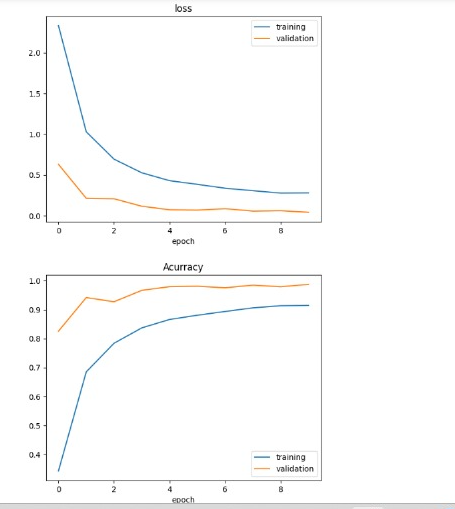
X\_train=np.array(list(map(preprocessing,X\_train))) # TO IRETATE AND PREPROCESS ALL IMAGES

X\_validation=np.array(list(map(preprocessing,X\_validation))) X\_test=np.array(list(map(preprocessing,X\_test))) cv2.imshow("GrayScale Images",X\_train[random.randint(0,len(X\_train)1)])

ADD A DEPTH OF 1:

X\_train=X\_train.reshape(X\_train.shape[0],X\_train.shape[1],X\_train.shap e[2],1)

X\_validation=X\_validation.reshape(X\_validation.shape[0],X\_validation. shape[1],X\_validation.shape[2],1) X\_test=X\_test.reshape(X\_test.shape[0],X\_test.shape[1],X\_test.shape[2], 1)



### 5.3 Testing code:

import numpy as np import cv2

import pickle

frameWidth= 640 # CAMERA RESOLUTION

frameHeight = 480

brightness = 180

threshold = 0.75 # PROBABLITY THRESHOLD font = cv2.FONT\_HERSHEY\_SIMPLEX

# SETUP THE VIDEO CAMERA

cap = cv2.VideoCapture(0)

cap.set(3, frameWidth) cap.set(4, frameHeight) cap.set(10, brightness)

# IMPORT THE TRANNIED MODEL

#import pickle

pickle\_in = open("model\_trained.p","rb")

model = pickle.load(pickle\_in)

def grayscale(img):

img = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) return img

def equalize(img):

img =cv2.equalizeHist(img) return img

def preprocessing(img): img = grayscale(img) img = equalize(img) img = img/255

return img

def getCalssName(classNo):

if classNo == 0: return 'Speed Limit 20 km/h' elif classNo == 1: return 'Speed Limit 30 km/h' elif classNo == 2: return 'Speed Limit 50 km/h' elif classNo == 3: return 'Speed Limit 60 km/h' elif classNo == 4: return 'Speed Limit 70 km/h' elif classNo == 5: return 'Speed Limit 80 km/h'

elif classNo == 6: return 'End of Speed Limit 80 km/h' elif classNo == 7: return 'Speed Limit 100 km/h'

elif classNo == 8: return 'Speed Limit 120 km/h' elif classNo == 9: return 'No passing'

elif classNo == 10: return 'No passing for vechiles over 3.5 metric tons'

elif classNo == 11: return 'Right-of-way at the next intersection' elif classNo == 12: return 'Priority road'

elif classNo == 13: return 'Yield'

elif classNo == 14: return 'Stop'

elif classNo == 15: return 'No vechiles'

elif classNo == 16: return 'Vechiles over 3.5 metric tons prohibited'

elif classNo == 17: return 'No entry’

elif classNo == 18: return 'General caution'

elif classNo == 19: return 'Dangerous curve to the left' elif classNo == 20: return 'Dangerous curve to the right' elif classNo == 21: return 'Double curve'

elif classNo == 22: return 'Bumpy road' elif classNo == 23: return 'Slippery road'

elif classNo == 24: return 'Road narrows on the right' elif classNo == 25: return 'Road work'

elif classNo == 26: return 'Traffic signals' elif classNo == 27: return 'Pedestrians'

elif classNo == 28: return 'Children crossing' elif classNo == 29: return 'Bicycles crossing' elif classNo == 30: return 'Beware of ice/snow'

elif classNo == 31: return 'Wild animals crossing'

elif classNo == 32: return 'End of all speed and passing limits' elif classNo == 33: return 'Turn right ahead'

elif classNo == 34: return 'Turn left ahead' elif classNo == 35: return 'Ahead only'

elif classNo == 36: return 'Go straight or right' elif classNo == 37: return 'Go straight or left' elif classNo == 38: return 'Keep right'

elif classNo == 39: return 'Keep left'

elif classNo == 40: return 'Roundabout mandatory' elif classNo == 41: return 'End of no passing'

elif classNo == 42: return 'End of no passing by vechiles over 3.5 metric tons'

while True:

# READ IMAGE

success, imgOrignal = cap.read()

# PROCESS IMAGE

img = np.asarray(imgOrignal) img = cv2.resize(img, (32, 32)) img = preprocessing(img)

cv2.imshow("Processed Image", img) img = img.reshape(1, 32, 32, 1)

cv2.putText(imgOrignal, "CLASS: " , (20, 35), font, 0.75, (0, 0, 255),

2, cv2.LINE\_AA)

cv2.putText(imgOrignal, "PROBABILITY: ", (20, 75), font, 0.75, (0,

0, 255), 2, cv2.LINE\_AA) # PREDICT IMAGE

predictions = model.predict(img)

classIndex = np.argmax(model.predict(img),axis=1) probabilityValue =np.amax(predictions)

if probabilityValue > threshold: cv2.putText(imgOrignal,str(classIndex)+"

"+str(getCalssName(classIndex)), (120, 35), font, 0.75, (0, 0, 255), 2,

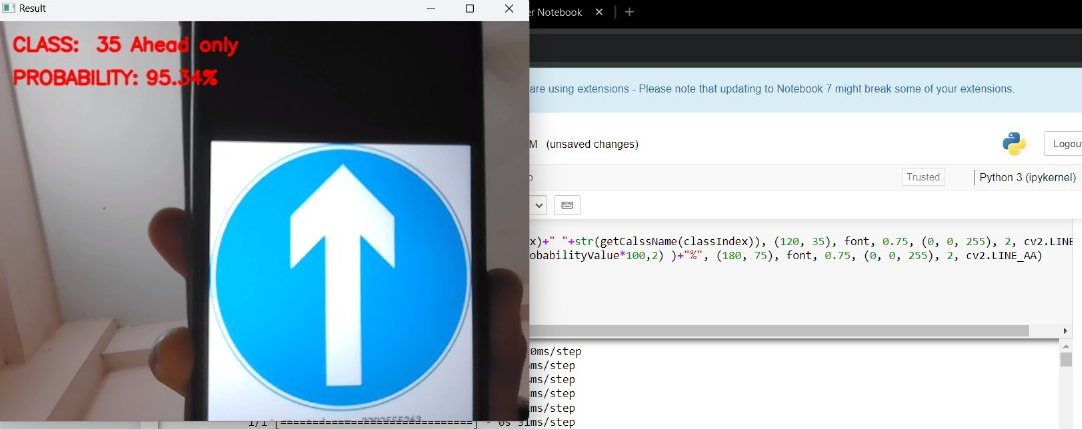
cv2.LINE\_AA)

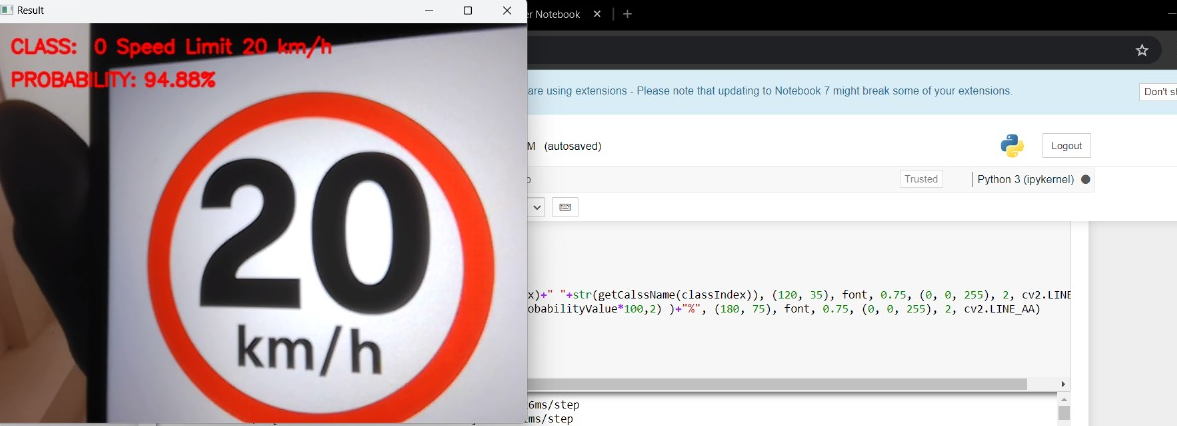
cv2.putText(imgOrignal, str(round(probabilityValue\*100,2) )+"%", (180, 75), font, 0.75, (0, 0, 255), 2, cv2.LINE\_AA)

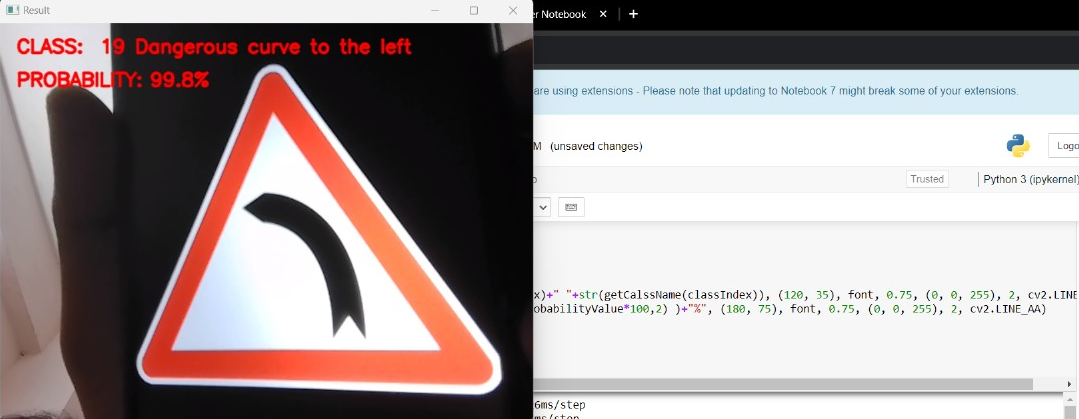
cv2.imshow("Result", imgOrignal)

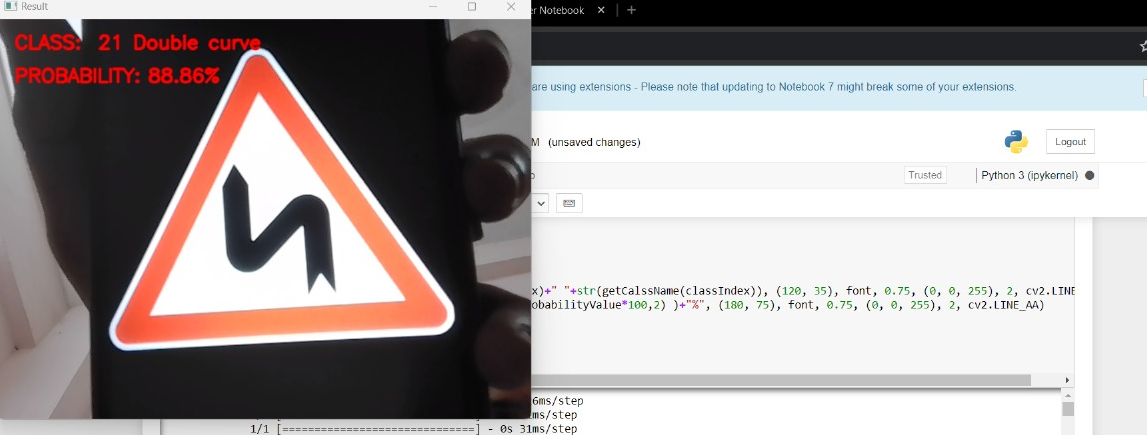
if cv2.waitKey(1) and 0xFF == ord('q'): break

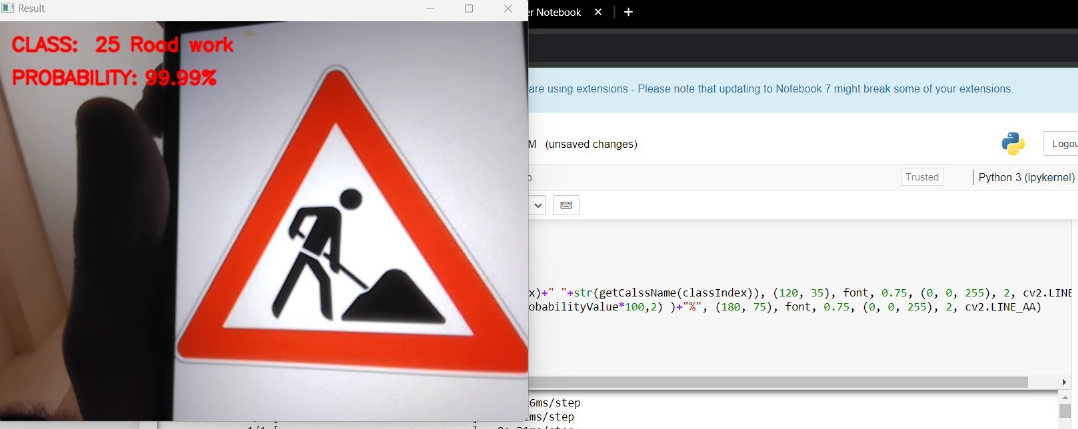
**RESULT:**











# **CHAPTER 6**

## System Testing

6.1 Introduction

The cause of testing is to detect mistakes. Making an attempt out is the technique of looking for to realize each viable fault or weakness in a piece product. It presents a method to determine the performance of add-ons, sub-assemblies, assemblies and/or a completed product. It is the method of excising program with the intent of constructing certain that the application procedure meets its necessities and client expectations and does no longer fail in an unacceptable process.There are rather plenty of forms of scan. Each experiment sort addresses a special trying out requirement.

6.2 Types of Testing

*Unit Testing*

Unit checking out involves the design of scan circumstances that validate that the Internal application good judgment is functioning safely, and that program inputs produce legitimate outputs. All decision branches and interior code float must be validated. It's the checking out of character application items of the application . It is achieved after the completion of an person unit earlier than integration. It is a structural checking out, that relies on competencies of its construction and is invasive. Unit exams participate in common exams at component level and scan a distinct business approach, utility, and/or process configuration. Unit assessments be certain that every specified course of a industry method performs appropriately to the documented requisites and involves clearly outlined inputs and anticipated results.

*Integration Testing:*

Integration Testing are designed to scan built-in program accessories to determine within the occasion that they evidently run as one software. Trying out is occasion driven and is more concerned with the fundamental final result of screens or fields. Integration assessments reveal that despite the fact that the accessories had been for my part pleasure, as proven through effectively unit checking out, the combo of accessories is correct and regular. Integration checking out is chiefly aimed at exposing the issues that come up from the performance of different components.

*Functional Testing:*

Functional Testing checks provide systematic demonstrations that capabilities established are to be had as particular by means of the business and technical specifications, method documentation, and consumer manuals. Functional testing is working on below mentioned data:

Legitimate input: Identified lessons of legitimate input ought to be accredited.

Invalid enter : recognized lessons of unacceptable effort must be rejected.

Capabilities : Recognized features ought to be exercised.

Output : Recognized courses of software outputs have got to be exercised.

Systems/Procedures: performance of the system here was invoked

­­

Individual and team work of useful checks is fascinated by specifications, key capabilities, or special scan instances. Moreover, systematic insurance plan concerning establish business method flows; data fields, predefined processes, and successive strategies have to be regarded for trying out. Before useful trying out is whole, extra checks are recognized and the strong price of present checks be strongminded.

*System Testing:*

Scheme difficult ensure so as to the whole included agenda process meets principles. It exams a pattern to make sure identified and predictable outcome. An illustration of procedure testing is the configuration oriented approach integration scan. System testing is based on approach descriptions and flows, emphasizing pre-driven system links and integration aspects.

*White Box Testing:*

This testing is a trying out wherein where the application tester has competencies of the interior workings, constitution and software language, or at least its cause. It's rationale. It's used to test areas that can't be reached from a black box stage.

Black Box Testing*:*

This is testing the software with none advantage of the inside workings, establishment or words of the unit life form veteran. Black field checks, as most other sorts of

6.3 Levels of Testing

*Unit Testing Strategy:*

Unit checking out is most commonly performed as a part of a mixed code and unit experiment part of the software lifecycle, though it be not exceptional for coding and unit checking out to be performed as two targeted phases.

*Test Strategy And Approach:*

Field testing out can be carried out manually and sensible assessments shall be written in element.

Test Objectives:

Each field must be work correctly.

Each page must be activated through the specified link.

Features to be tested Verify that the entries are of the correct format No duplicate entries should be allowed.

*Integration Testing Strategy:*

Software integration testing is the incremental integration checking out of two otherwise further included software gears on top of a solo stage to fabricate failure induced with the aid of interface defects. The project of the mixing scan is to check that components or program applications,

e.g. Components in a program approach or œ one step up œ software purposes at the company degree œ interact without error.

Test Results: All of the scan circumstances recounted above passed efficiently. No defects encountered.

Acceptance Testing:

User Acceptance testing trying out is a crucial section of any mission and requires enormous participation by the tip user. It additionally ensures that the procedure meets the functional specifications.

Test Results: The entire test cases recounted above passed effectually. No defects Encountered.

# **CHAPTER 7**

## 7. Conclusion

In conclusion, the traffic sign recognition system using CNN project is an effective application of deep learning techniques for traffic sign detection and recognition. The system is designed to automatically recognize traffic signs from images or video feeds using convolutional neural networks (CNN).

The project involves several stages, including image acquisition, preprocessing, data augmentation, feature extraction, and classification. The CNN model is trained using a large dataset of traffic sign images, which enables it to learn the features and characteristics of different traffic signs. The model's accuracy can be improved by optimizing the hyperparameters, using different architectures, and implementing real- time performance optimizations.

The traffic sign recognition system using CNN has several advantages over traditional computer vision techniques, including improved accuracy, robustness to variations in lighting and backgrounds, and the ability to detect and recognize multiple signs simultaneously. The system has the potential to be used in various settings, including autonomous vehicles, smart cities, and traffic monitoring systems.

However, some challenges remain, including the need for a large dataset of annotated images, the need to overcome overfitting, and the need to ensure the system's reliability and robustness in real-world scenarios. Further research and development are required to address these challenges and to ensure the system's effectiveness and scalability.

# 

* + - * .

# **CHAPTER 8**

## 8. Future Enhancement

There are several possible future enhancements for the traffic sign recognition system project:

**Real-time performance optimization:** The system can be further optimized for real-time performance, allowing it to operate efficiently in real-world scenarios.

**Improved accuracy:** The accuracy of the system can be improved by using a larger and more diverse dataset, optimizing the hyperparameters, and exploring new architectures and techniques.

**Multi-task learning:** The system can be extended to recognize multiple tasks simultaneously, such as detecting and recognizing road markings, pedestrian crossings, and traffic lights.

**Object detection:** The system can be enhanced to detect and recognize other objects, such as vehicles, pedestrians, and cyclists, which can improve road safety and traffic management.

**Multi-camera systems:** The system can be extended to work with multi- camera systems, which can provide a more comprehensive view of the road and improve accuracy.

**Cloud-based solution:** The system can be implemented as a cloud-based solution, allowing it to process large amounts of data and provide real- time analysis and insights.

**Edge computing:** The system can be optimized for edge computing, allowing it to operate efficiently on resource-constrained devices, such as smartphones and embedded systems.

**Transfer learning:** Transfer learning can be explored to improve the system's performance by leveraging the knowledge learned from other related tasks or domains.

Overall, the future enhancements of the traffic sign recognition system project can improve the system's accuracy, scalability, and effectiveness in various scenarios.

# **CHAPTER 9**

## REFERENCES

* https://ieeexplore.ieee.org/document/9498270
* https://youtu.be/BvYFGScsE7Y
* https://[www.studocu.com/in/document/visvesvaraya-](http://www.studocu.com/in/document/visvesvaraya-)technological-university/computer-communication- networks/traffic-sign-detection-recognition-system- report/30131569
* https://[www.powershow.com/view/16db73-](http://www.powershow.com/view/16db73-) YmNkM/Traffic\_Sign\_Recognition\_powerpoint\_ppt\_pres entation
* https://[www.slideserve.com/zada/traffic-sign-recognition](http://www.slideserve.com/zada/traffic-sign-recognition)
* https://[www.slideshare.net/anujraghav123/traffic-sign-](http://www.slideshare.net/anujraghav123/traffic-sign-)recognition-183188970
* https://[www.slideshare.net/cseij/trafficsign-recognition-](http://www.slideshare.net/cseij/trafficsign-recognition-)for-an-intelligent-vehicledriver-assistant-system-using- hog
* https://[www.ijert.org/traffic-sign-recognition-using-](http://www.ijert.org/traffic-sign-recognition-using-)machine-learning-a-