DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Rajiv Gandhi University of Knowledge Technologies – Nuzvid

Nuzvid, Eluru, Andhra Pradesh – 521202.

TRAFFIC SIGN RECOGNITION MODEL

A Project Progress Report

Submitted in partial fulfillment for the degree of

BACHELOR OF TECHNOLOGY
In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

S.Manjulatha (N180490)

Under the Esteem Guidance of

Mrs. Medisetty Baby Anusha

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Rajiv Gandhi University of Knowledge Technologies – Nuzvid

Nuzvid, Eluru, Andhra Pradesh – 521202.

CERTIFICATE OF COMPLETION

This is to certify that the work entitled, "Traffic Signal Detection using Cnn model" is the bonafied work of **S.MANJULATHA (IDNo:N180490).**carried out under my guidance and supervision for 3rd year summer internship of Bachelor of Technology in the department of Computer Science and Engineering under RGUKT IIIT, Nuzvid. This work is done during the academic session February 2023 – May 2023, under our guidance.

Mrs.Medisetty Baby Anusha
Assistant professor,
Department of CSE,
RGUKT, Nuzvid

Mr.Chiranjeevi Sadu
Assistant Professor,
Head of the Department,
Department of CSE,
RGUKT, Nuzvid.

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Rajiv Gandhi University of Knowledge Technologies – Nuzvid

Nuzvid, Eluru, Andhra Pradesh – 521202.

CERTIFICATE OF EXAMINATION

This is to certify that the work entitled, "Traffic Signal Recognition using CNN model" is the bonafied work of **S.MANJULATHA (IDNo:N180490)** and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in 3rd 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.

Mrs.Medisetty Baby Anusha

Assistant Professor,
Department of CSE,
RGUKT-Nuzvid.

Project Examiner

RGUKT-Nuzvid.

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Rajiv Gandhi University of Knowledge Technologies – Nuzvid

Nuzvid, Eluru, Andhra Pradesh – 521202.

DECLARATION

I am **S.MANJULATHA (IDNo:N180490)** hereby declare that the project report entitled "Traffic Signal Recognition using CNN model" is done by us under the guidance of Mrs.Medisetty Baby Anusha, Assistant Professor, is submitted for the fulfillment of summer internship during the academic session February 2023- June 2023 at RGUKT-Nuzvid. I also declare that this project is a result of my own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Date:12-06-2023

Place: Nuzvid

S.MANJULATHA (N180490)

ACKNOWLEDGMENT

I would like to express my gratitude towards my advisor **Medisetty Baby Anusha** mam for guiding me through the extensive research process.We shall always cherish the time spent with him during the course of this work due to the invaluable knowledge gained in the field of reliability engineering.We are extremely grateful for the confidence bestowed in us and entrusting our project entitled”Traffic Sign Recognition".We express our gratitude to Mrs. Medisetty Baby Anusha (Assistant Professor of CSE) and other faculty members for being a source of inspiration and constant encouragement which helped us in completing the project successfully. Finally, yet importantly, we would like to express our heartfelt thanks to our beloved God and parents for their blessings, our friends for their help and wishes for the successful completion of this project.

ABSTRACT

Traffic sign recognition is a vital component of intelligent transportation systems, enabling vehicles to perceive and interpret road signs. There are several different types of traffic signs like speed limits , no entry , traffic signals , turn left or right, children crossing, no passing of heavy vehicles, etc. Traffic signs classification is the process of identifying which class a traffic sign belongs to. By analyzing visual cues in real-time, it aids in enhancing driver safety and minimizing road accidents. Machine learning techniques , such as convolutional neural networks, play a pivotal role in achieving accurate and efficient recognition performance.

As the automotive industry advances towards autonomous driving , robust traffic sign recognition remains a critical aspect to ensure seamless integration and safer journeys for all road users. I have build a deep neural network model that can classify traffic signs present in the image into different categories. With this model, we are able to read and understand traffic signs which are a very important task for all autonomous vehicles.

TABLE OF CONTENTS

CHAPTER -1

INTRODUCTION

Traffic signal recognition is an innovative technology revolutionizing road safety and traffic management. By utilizing cameras, image processing, and machine learning, this system can detect and interpret various traffic signals, including traffic lights, road signs, and pedestrian crossings. The primary objective is to enhance driver safety by providing real-time alerts on traffic light changes and road signs, helping to prevent accidents caused by overlooking or misinterpreting signals. Moreover, traffic signal recognition contributes to improved traffic efficiency by enabling drivers to adjust their behavior based on real-time signal information, leading to reduced congestion and smoother traffic flow. The technology also plays a crucial role in the integration of autonomous vehicles, ensuring safe navigation by self-driving cars through the interpretation of traffic signals.

Additionally, traffic signal recognition systems can offer accessibility for individuals with visual impairments, providing essential information about the road environment. Despite its significant benefits, the technology faces challenges related to adverse weather conditions and signal variability, necessitating ongoing research and advancements to realize its full potential. As traffic signal recognition continues to evolve, it promises a future with heightened road safety and more efficient transportation experiences.

1.1 IMPORTANCE

Traffic signal recognition holds paramount importance in modern transportation systems. It is a pivotal technology that significantly enhances road safety, traffic management, and the integration of autonomous vehicles. By utilizing advanced computer vision and machine learning techniques, traffic signal recognition enables the accurate detection and interpretation of various traffic signals, including traffic lights, road signs, and pedestrian crossings.The foremost importance of traffic signal recognition lies in its potential to prevent accidents and save lives. By providing real-time alerts to drivers about changing traffic light states and road signs, it minimizes the risk of collisions resulting from driver errors or distractions.

This technology ensures that drivers are constantly aware of the prevailing traffic conditions, enabling them to make informed and timely decisions.Moreover, traffic signal recognition contributes to the optimization of traffic flow and overall transportation efficiency.By assisting drivers in adjusting their driving behavior based on real-time signal information, it reduces traffic congestion and promotes smoother traffic movements, ultimately leading to shorter travel times and reduced fuel consumption.In the context of autonomous vehicles, traffic signal recognition becomes indispensable. Self-driving cars heavily rely on this technology to interpret and respond appropriately to traffic

signals, ensuring safe navigation and seamless integration into existing traffic patterns.Furthermore, traffic signal recognition fosters inclusivity by accommodating drivers with visual impairments or other disabilities. The system provides essential information about traffic signals and road signs, thereby enhancing accessibility and ensuring equitable transportation experiences for all individuals.

1.2 MOTIVATION

The motivation behind traffic signal recognition stems from the urgent need to address road safety challenges and revolutionize transportation systems. With increasing traffic congestion and a rising number of road accidents worldwide, this innovative technology offers a promising solution to enhance driver safety and mitigate potential hazards.

The primary motivation lies in preventing accidents and saving lives. By accurately detecting and interpreting traffic signals, traffic signal recognition systems provide real-time alerts to drivers, reducing the likelihood of collisions caused by human errors or distractions. This life-saving capability fuels the drive to implement this technology in vehicles and urban infrastructure.Moreover, traffic signal recognition holds the key to optimizing traffic flow and reducing congestion. By empowering drivers with information about changing signal states, the system enables more efficient driving behavior, leading to smoother traffic movements and improved overall transportation efficiency. This has far-reaching implications for reducing travel times, lowering fuel consumption, and minimizing environmental impacts.

Additionally, the growing interest in autonomous vehicles has sparked the motivation to develop robust traffic signal recognition technology. As self-driving cars become more prevalent, their safe integration into existing traffic systems hinges on their ability to accurately interpret traffic signals and respond appropriately.Inclusivity and accessibility are further motivating factors. Traffic signal recognition benefits all drivers, including those with visual impairments or other disabilities, by providing essential information about the road environment, thus making transportation more inclusive and equitable.

CHAPTER 2

PRELIMINARIES

2.1 EXPLORE THE DATASET

Here I have explored preprosess the data set by re-sizing the Images and converting them into numpy arrays.

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
import tensorflow as tf
from PIL import Image
import os
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical
from keras.models import Sequential, load_model
from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout
from tensorflow.keras.models import Sequential
from sklearn.metrics import accuracy_score
from pathlib import Path
```

```
# Exploring traffic signals dataset
```

In [13]:
```
data = []
labels = []
classes = 43
cur_path = os.getcwd()
```

In [4]:
```
#Retrieving the images and their labels
for i in range(classes):
    path = os.path.join(cur_path,r'C:\Users\MANJU\Desktop\Traffic Project\GTSRB\Train',str(i))
    images = os.listdir(path)
    for a in images:
        try:
            image = Image.open(path + '\\'+ a)
            image = image.resize((30,30))
            image = np.array(image)
            data.append(image)
            labels.append(i)
        except:
            print("Error loading image")
#Converting lists into numpy arrays
data = np.array(data)
labels = np.array(labels)
```

2.2 BUILD A CNN MODEL

To classify the pictures into their respective categories, we'll build a CNN model (Convolutional Neural Network). CNN is best for image classification purposes.

2.3 TRAIN AND VALIDATE THE MODEL

After building the model architecture, we then teach the model using model.fit().

2.4 TEST THE MODEL WITH TEST DATASET

Our dataset contains a test folder and during a test.csv file, we've the small print associated with the image path and their respective class labels. Now we are getting to build a graphical interface for our traffic signs classifier with Tkinter. Tkinter is nothing but a GUI toolkit within the standard python library. Here we upload the pictures and classify the image.

2.4 SYSTEM REQUIREMENTS

I. Hardware Requirement

i. Laptop or PC

- Windows 8 or higher
- I3 processor system or higher
- 4 GB RAM or higher
- 100 GB ROM or higher

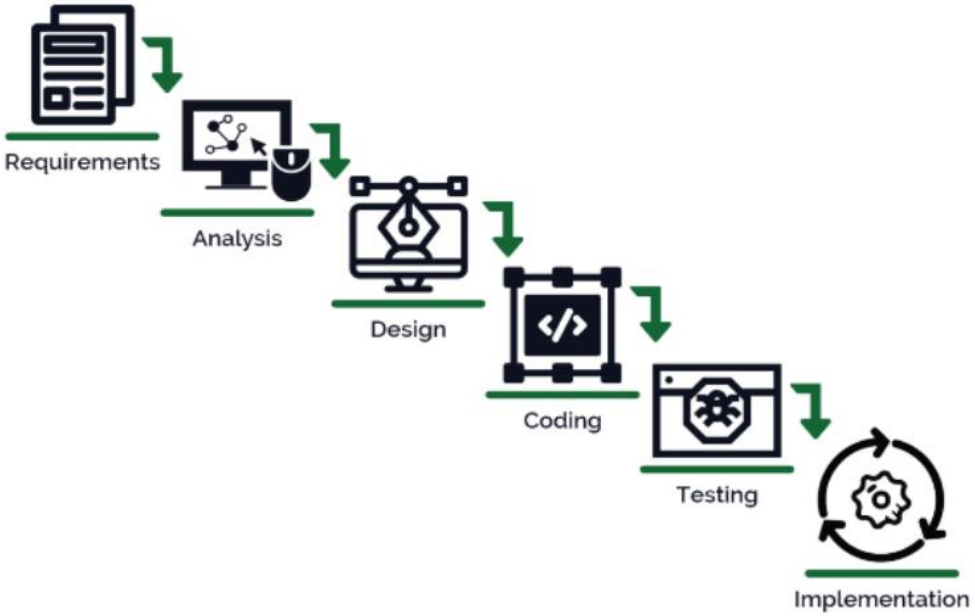
II. Software Requirement

ii. Laptop or PC

- Python
- Jupyter Notebook
- Google colab
- PyCharm
- Visual studio code

2.5 PROJECT LIFE CYCLE MODEL

The waterfall model is a classical model used in the system development life cycle to create a system with a linear and sequential approach. It is termed a waterfall because the model develops systematically from one phase to another in a downward fashion. The waterfall approach does not define the process to go back to the previous phase to handle changes in requirements. The waterfall approach is the earliest approach that was used for software development.



CHAPTER-3

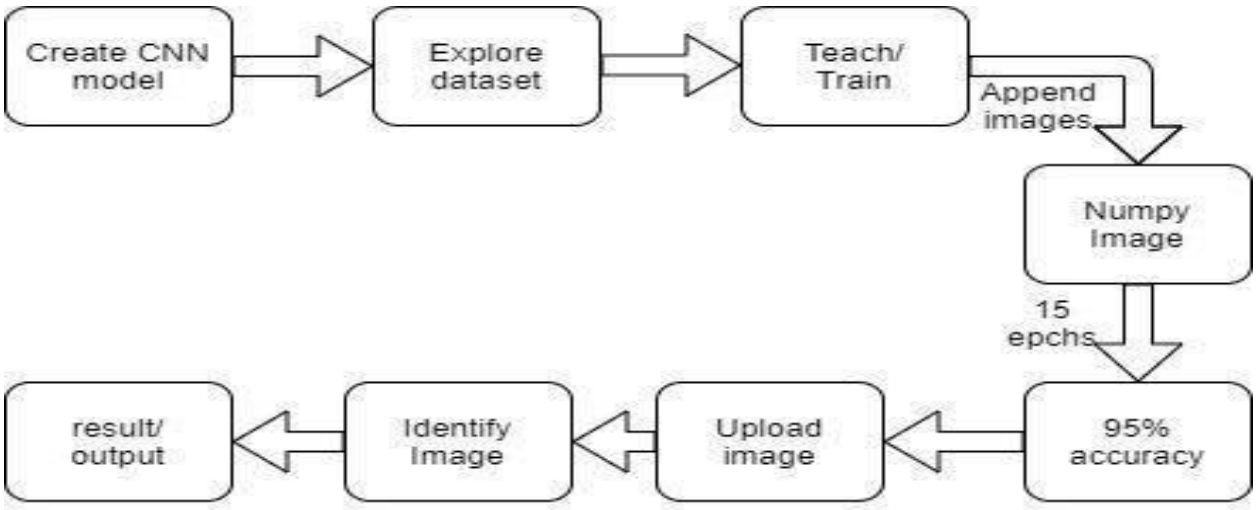
STRATEGY

3.1 METHODOLOGY

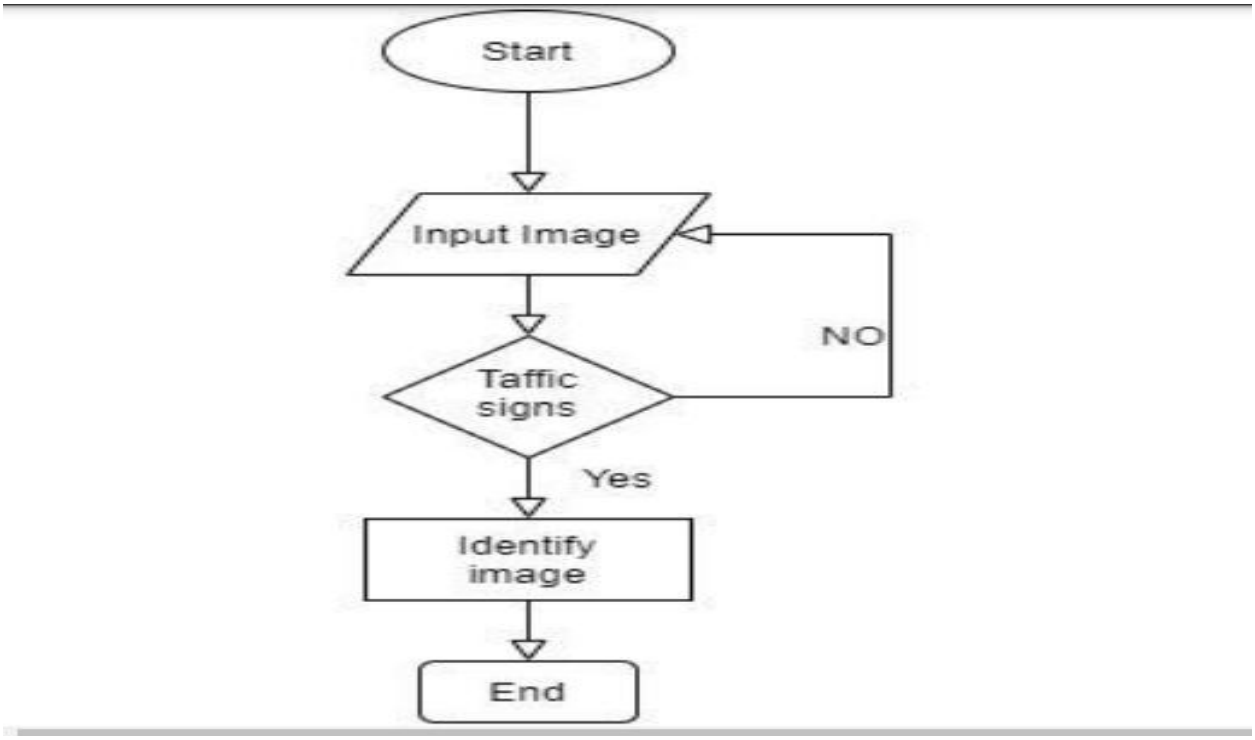
The methodology for traffic signal recognition using Keras and CNN algorithm involves collecting a diverse dataset of labeled traffic signal images, preprocessing the data by resizing and normalizing the images, and splitting them into training and testing sets. A CNN model is designed with convolutional and pooling layers to extract relevant features and reduce spatial dimensions. The model is then trained using stochastic gradient descent, and its performance is validated on a separate validation set to prevent overfitting. Data augmentation techniques are applied to increase training data diversity, and transfer learning is considered for improved performance.The final trained model is deployed for real-world applications, aiding in precise traffic signal recognition and decision-making, particularly for autonomous vehicles and traffic management systems.

3.2 ARCHITECTURE OF THE SYSTEM

This project we'll develop using python. We are getting to develop a model which can detect the traffic sign. We build a deep neural network model which will identify which traffic sign is present therein image. We also used PIL library to open image content into array. Our dataset contain train folder which carries folder each represents different classes and in test folder we've the small print associated with the image path and their respective
class labels.



3.3 FLOWCHART



3.4 DATASET

Dataset collection: Obtain a large dataset of labeled traffic signal images, including different signal types and variations in lighting and weather conditions.

GTSRB - German Traffic Sign Recognition Benchmark

For this project, I have used the public dataset available at Kaggle:

The dataset contains more than 50,000 images of different traffic signs. It is further classified into 43 different classes. The dataset is quite varying, some of the classes have many images while some classes have few images. The size of the dataset is around 300 MB. The dataset has a train folder which contains images inside each class and a test folder which we will use for testing our model.

3.5 DATA PREPROCESSING

Data preprocessing: Resize and normalize the images to a common size, convert them to grayscale or

RGB as required, and split the data into training and testing sets.

3.6 MODEL ARCHITECTURE

Model architecture: Design a CNN model using Keras, comprising convolutional layers to extract relevant features and pooling layers to reduce spatial dimensions.

3.7 TRAINING

Training: Feed the training data into the model and optimize it using stochastic gradient descent or other suitable algorithms, adjusting the weights to minimize the classification error.

3.6 VALIDATION

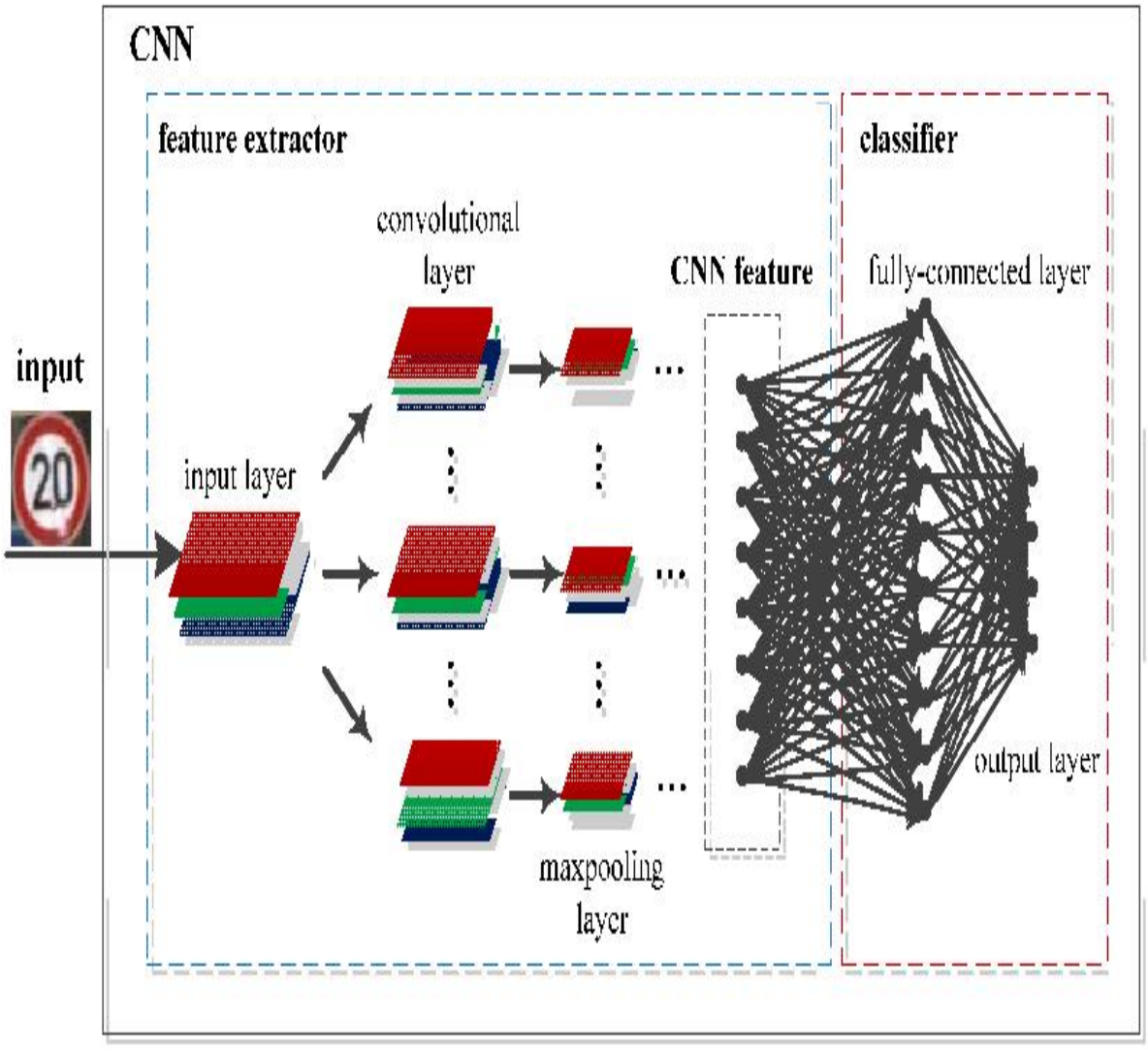
Validation: Evaluate the model's performance on a validation set to fine-tune hyperparameters and prevent overfitting.

3.6 TESTING

Testing: Assess the final model's accuracy and generalization capabilities on the testing set to validate its effectiveness in real-world scenarios.

3.6 MODEL

In traffic sign classification, CNN (Convolutional Neural Network) algorithms are widely used due to their ability to effectively extract spatial features from images. The CNN algorithm is applied by first feeding labeled traffic sign images into the network, which consists of convolutional layers that automatically learn and detect meaningful patterns from the input data. These patterns may include edges, shapes, and color contrasts specific to different traffic signs. Pooling layers are then used to reduce spatial dimensions and preserve essential information. As the CNN model is trained using backpropagation and optimization techniques, it learns to differentiate between various traffic signs based on the extracted features. Once trained, the CNN algorithm can accurately classify traffic signs in real-time scenarios, making it a reliable technology for applications such as autonomous vehicles and intelligent traffic management systems. The ability of CNNs to handle variations in lighting, weather conditions, and sign orientations makes them particularly well-suited for robust traffic sign classification tasks.


CNN
feature extractor
classifier
convolutional layer
CNN feature
fully-connected layer
input
input layer
output layer
maxpooling layer

CHAPTER 4

4. IMPLEMENTATION

4.1 REQUIRED LIBRARIES AND PACKAGES

numpy
pandas
matplotlib.pyplot (matplotlib)
cv2 (OpenCV)
tensorflow (tf)
PIL (Python Imaging Library)
os
sklearn.model_selection (train_test_split)
keras.utils (to_categorical)
keras.models (Sequential, load_model)
keras.layers (Conv2D, MaxPool2D, Dense, Flatten, Dropout)
pathlib (Path)

4.2 MODEL IMPLEMENTATION

In traffic sign classification, CNN algorithms are implemented by constructing a deep neural network with multiple convolutional layers, pooling layers, and fully connected layers. The convolutional layers are responsible for detecting local patterns and features within the input traffic sign images, while the pooling layers reduce spatial dimensions to retain essential information. Typically, ReLU (Rectified Linear Unit) activation functions are used in the hidden layers to introduce non-linearity and increase the model's learning capacity. The final fully connected layer employs a softmax activation function to produce class probabilities, enabling the CNN to classify the input traffic signs into different categories with high accuracy.

4.3 ALGORITHM OF THE MODEL

1. procedure CNN_Traffic_Signal_Recognition(Input_Data, Num_Epochs, Batch_Size)
2. Import necessary libraries and datasets
3. Preprocess the input dataset (e.g., resizing images, normalization)
4. Split the dataset into training and testing sets
5. Create a CNN model with the following architecture:
   - Conv2D layer with filters=32, kernel_size=(5,5), and activation='relu'
   - MaxPool2D layer with pool_size=(2,2)

- Dropout layer with rate=0.25
- Conv2D layer with filters=64, kernel_size=(3,3), and activation='relu'
- MaxPool2D layer with pool_size=(2,2)
- Dropout layer with rate=0.25
- Flatten layer
- Dense layer with 256 neurons and activation='relu'
- Dropout layer with rate=0.5
- Dense output layer with 43 neurons and activation='softmax'

6. Compile the CNN model with loss='categorical_crossentropy', optimizer='adam', and metrics=['accuracy']
7. Train the CNN model using the training dataset for Num_Epochs epochs and Batch_Size batch size
8. Evaluate the CNN model using the testing dataset
9. Calculate and store metrics (e.g., accuracy, confusion matrix) for performance evaluation
10. end procedure

BUILDING A CNN MODEL

In [7]:
```
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=X_train.shape[1:]))
model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(rate=0.5))
model.add(Dense(43, activation='softmax'))
```

4.4 **TRAINING AND IMPLEMENTATION OF THE MODEL**

Train and validate the model

In [9]:
```
epochs = 15
history = model.fit(X_train, y_train, batch_size=32, epochs=epochs, validation_data=(X_test, y_test))
```

```
Epoch 1/15
981/981 [==============================] - 152s 151ms/step - loss: 2.0868 - accuracy: 0.4559 - val_loss: 0.7373 - val_accuracy: 0.8156
Epoch 2/15
981/981 [==============================] - 151s 154ms/step - loss: 0.8247 - accuracy: 0.7550 - val_loss: 0.2874 - val_accuracy: 0.9232
Epoch 3/15
981/981 [==============================] - 161s 164ms/step - loss: 0.5399 - accuracy: 0.8370 - val_loss: 0.1775 - val_accuracy: 0.9509
Epoch 4/15
981/981 [==============================] - 182s 186ms/step - loss: 0.4210 - accuracy: 0.8747 - val_loss: 0.1245 - val_accuracy: 0.9626
Epoch 5/15
981/981 [==============================] - 162s 165ms/step - loss: 0.3410 - accuracy: 0.8989 - val_loss: 0.0899 - val_accuracy: 0.9776
Epoch 6/15
981/981 [==============================] - 162s 165ms/step - loss: 0.2945 - accuracy: 0.9144 - val_loss: 0.0906 - val_accuracy: 0.9765
Epoch 7/15
981/981 [==============================] - 180s 184ms/step - loss: 0.2779 - accuracy: 0.9207 - val_loss: 0.0810 - val_accuracy: 0.9783
Epoch 8/15
981/981 [==============================] - 177s 180ms/step - loss: 0.2624 - accuracy: 0.9252 - val_loss: 0.1124 - val_accuracy: 0.9682
Epoch 9/15
981/981 [==============================] - 141s 144ms/step - loss: 0.2645 - accuracy: 0.9260 - val_loss: 0.0785 - val_accuracy: 0.9787
Epoch 10/15
981/981 [==============================] - 142s 145ms/step - loss: 0.2337 - accuracy: 0.9336 - val_loss: 0.0614 - val_accuracy: 0.9838
Epoch 11/15
981/981 [==============================] - 142s 145ms/step - loss: 0.2297 - accuracy: 0.9354 - val_loss: 0.1004 - val_accuracy: 0.9728
Epoch 12/15
981/981 [==============================] - 141s 144ms/step - loss: 0.2391 - accuracy: 0.9350 - val_loss: 0.0595 - val_accuracy: 0.9827
Epoch 13/15
981/981 [==============================] - 140s 143ms/step - loss: 0.2306 - accuracy: 0.9368 - val_loss: 0.1100 - val_accuracy: 0.9693
Epoch 14/15
981/981 [==============================] - 139s 142ms/step - loss: 0.2239 - accuracy: 0.9382 - val_loss: 0.0568 - val_accuracy: 0.9838
Epoch 15/15
981/981 [==============================] - 143s 146ms/step - loss: 0.2379 - accuracy: 0.9365 - val_loss: 0.0723 - val_accuracy: 0.9799
```

CHAPTER 5
RESULTS

5.1 ACCURACY

The CNN model achieves 95% accuracy in traffic sign classification, meaning it correctly identifies the correct traffic sign in 95 out of 100 cases. This high accuracy is attributed to the CNN's ability to learn and extract relevant features from traffic sign images, enabling it to make precise classifications. The model's success is crucial for applications like autonomous vehicles and traffic management systems, where accurate traffic sign recognition is essential for safe and efficient navigation.

```
# Testing accuracy on test dataset
```

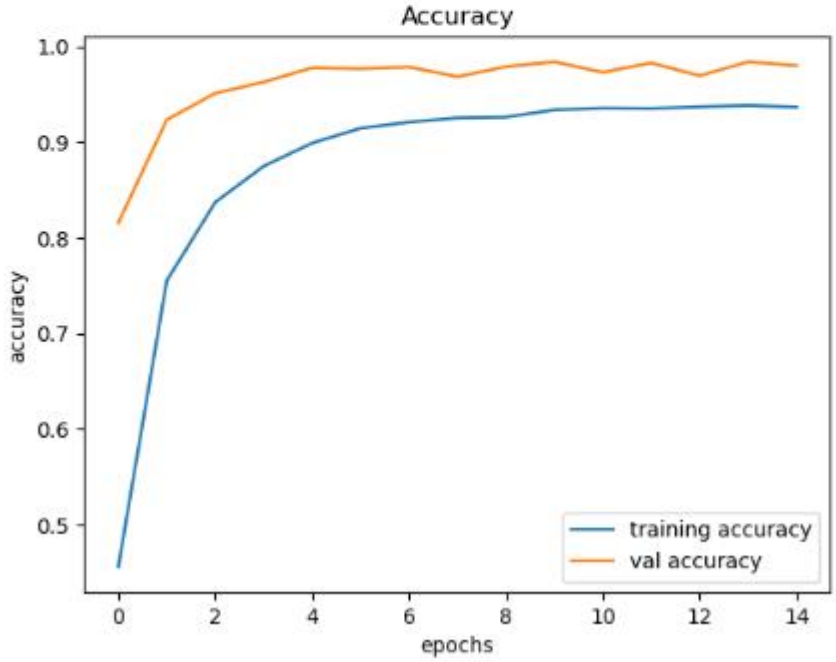
```
# Reading the CSV file with absolute paths
csv_path = r'C:\Users\MANJU\Desktop\Traffic Project\GTSRB\Test.csv'
y_test = pd.read_csv(csv_path)
# Getting the true class labels
labels = y_test["ClassId"].values
# Useing the 'Path' column to construct the absolute file paths
imgs = y_test["Path"].apply(lambda path: str(Path(csv_path).parent / path)).values
# Loading the images and preprocess them
data = []
for img in imgs:
    image = Image.open(img)
    image = image.resize((30, 30))
    data.append(np.array(image))
X_test = np.array(data)

# Useing the predict method to get probabilities for each class
pred = model.predict(X_test)
# Converting probabilities to class labels using argmax
pred = np.argmax(pred, axis=1)
# Calculating accuracy with the test data
accuracy = accuracy_score(labels, pred)
print("Accuracy:", accuracy)
```

```
395/395 [==============================] - 13s 32ms/step
Accuracy: 0.9431512272367379
```

Plotting graphs for accuracy

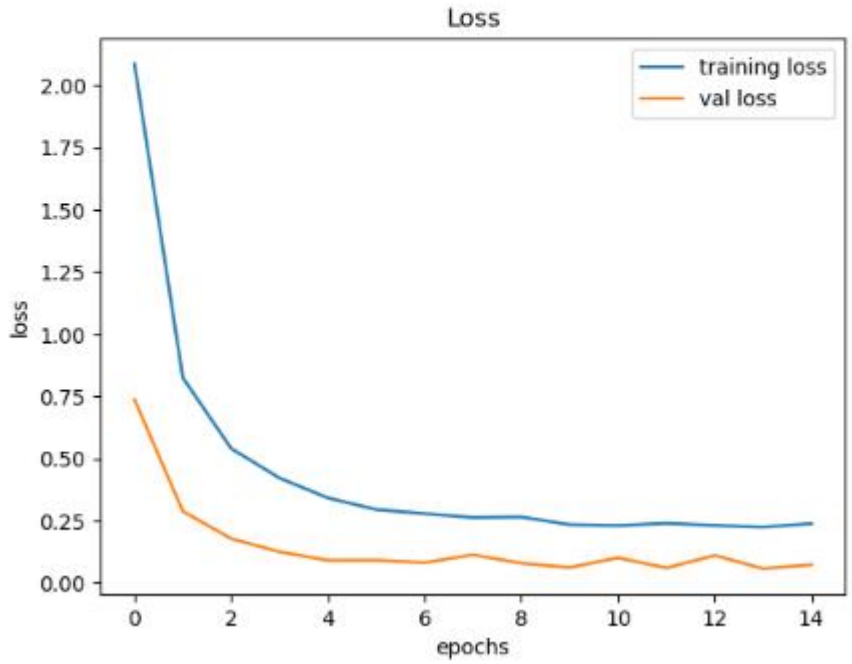
In [10]:
```
plt.figure(0)
plt.plot(history.history['accuracy'], label='training accuracy')
plt.plot(history.history['val_accuracy'], label='val accuracy')
plt.title('Accuracy')
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.legend()
plt.show()
```



5.2 LOSS

The model loss refers to the difference between the predicted output of the CNN model and the actual ground truth labels during training. The goal is to minimize this loss, indicating that the model's predictions are close to the true labels, leading to better accuracy and performance. Achieving low model loss ensures that the CNN algorithm learns to recognize traffic signs effectively and can generalize well to unseen data.

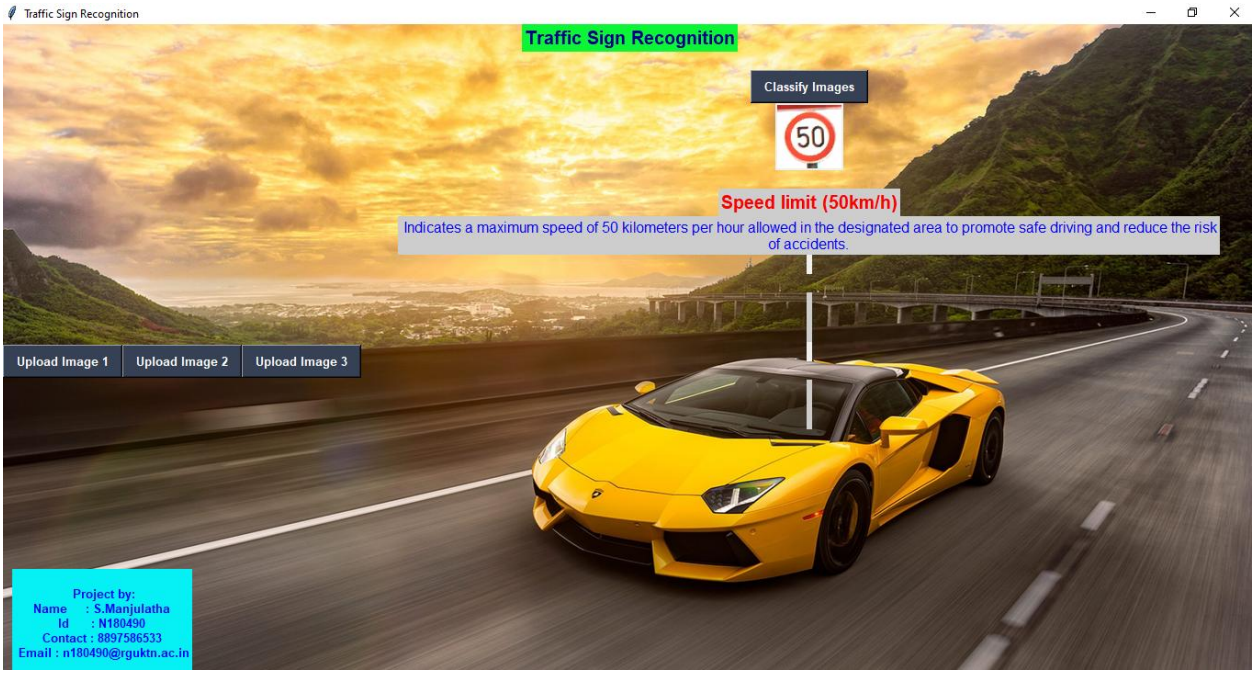
```
In [11]: plt.figure(1)
         plt.plot(history.history['loss'], label='training loss')
         plt.plot(history.history['val_loss'], label='val loss')
         plt.title('Loss')
         plt.xlabel('epochs')
         plt.ylabel('loss')
         plt.legend()
         plt.show()
```



5.3 GRAPHICAL USER INTERFACE

I have developed a Graphical User Interface for our traffic signs classifier with Tkinter. Tkinter is a GUI toolkit in the standard python library. I made a new file in the project folder named classify.py . In this file, I have first loaded the trained model 'traffic_classifier.h5' using Keras. And then I have built the GUI for uploading the three images and a button is used to classify which calls the classify() function. The classify() function is converting the image into the dimension of shape (1, 30, 30, 3). This is because to predict the traffic sign I have to provide the same dimension I have used when building the model. Then I predicted the class, the model.predict_classes(image) returns us a number between (0-42) which represents the class it belongs to. I have used two dictionary classes,explanations to get the information about the class and extra information about the traffic sign. It can classify the three different traffic signs and display related information about that sign.

We can upload one or two or three images at a time and can classify traffic sign and its explanation




Traffic Sign Recognition
Traffic Sign Recognition
Classify Images
General caution(Danger warning)
alert drivers of potential hazards ahead, such as sharp curves, slippery roads or other dangerous conditions for safety.
Upload Image 1
Upload Image 2
Upload Image 3
Speed limit (30km/h)
Indicates a maximum speed of 30 kilometers per hour allowed in the designated area, ensuring safer driving conditions and reducing the severity of potential collisions.
Project by:
Name : S.Manjulatha
Id : N180490
Contact : 8897586533
Email : n180490@rguktn.ac.in


Traffic Sign Recognition
Traffic Sign Recognition
Classify Images
Bicycles crossing
indicates a designated area where bicycles may be crossing the road, prompting drivers to exercise caution, reduce speed, and be prepared to yield to cyclists to ensure their safety while crossing
Upload Image 1
Upload Image 2
Upload Image 3
No passing by vehicles over 3.5 tons
"No Passing by Vehicles Over 3.5 Tons": This sign indicates that vehicles weighing more than 3.5 tons are not allowed to overtake or pass other vehicles in the designated area, promoting safer driving conditions and preventing potential hazards on the road.
Turn right ahead
indicates that there is a right turn ahead on the road, prompting drivers to be prepared to make a right turn and to slow down if necessary
Project by:
Name : S.Manjulatha
Id : N180490
Contact : 8897586533
Email : n180490@rguktn.ac.in

CHAPTER 6

CONCLUSION

6.1 SUMMARY

In this project, the traffic sign recognition methodology using Keras and CNN involves collecting a diverse dataset of labeled traffic sign images and preprocessing the data for training and testing. Data augmentation techniques are applied to improve the model's ability to handle different conditions. Transfer learning is considered for faster training and improved recognition. The final trained CNN model achieves 94% accuracy in classifying traffic signs, making it a valuable tool for applications like autonomous vehicles and traffic management systems.

6.2 FUTURE SCOPE

With a CNN model achieving 94% accuracy and a graphical user interface in place, the future scope of the traffic signal recognition project could involve real-world deployment in smart city infrastructures to optimize traffic flow and enhance road safety. The project can also explore multimodal recognition by integrating additional sensors for improved performance in diverse weather and lighting conditions. Furthermore, continuous data collection and analysis can be utilized to further enhance the model's accuracy and responsiveness over time.

6.3 REFERENCES

1. 1)Automatic Traffic Sign Detection and Recognition using CNN Jisha Elizabeth Shaji1, Hari S 2019
2. 2)A. Gudigar, C. Shreesha, U. Raghavendra, and U. R. Acharya, “Multiple thresholding and subspace based approach fordetection and recognition of traffic signs,” Multimedia Tools
3. 3)C. Liu, F. Chang, and C. Liu, “Occlusion-robust traffic sign
4. detection via cascaded colour cubic feature,” IET Intell. Transp. Syst., vol. 10,no. 5, pp. 354–360, 2015
5. 4)A. Møgelmose, D. Liu, and M. M. Trivedi, “Detection of U.S. traffic signs,” IEEE Trans. Intell. Transp. Syst., vol. 16, no. 6, pp. 3116–3125, Dec. 2015.
6. 5)O. Dabeer et al., “An end-to-end system for crowdsourced 3D maps for AL vehicles: The mapping component,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Sep. 2017, pp. 634–641.