TRAFFIC SIGN DETECTION USING NEURAL NETWORKS

PROJECT REPORT

submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

ELECTRONICS AND COMMUNICATION ENGINEERING

of

A.P.J ABDUL KALAM TECHNOLOGICAL UNIVERSITY,TRIVANDRUM,KERALA

by

ROBIN CR (RET17EC126)

Under the guidance of

Mr. AJAI V BABU

Department of Electronics and Communication Engineering

Rajagiri School of Engineering and Technology

Rajagiri Valley, Kakkanad, Kochi, 682039

2021

RSET
RAJAGIRI SCHOOL OF ENGINEERING & TECHNOLOGY
AUTONOMOUS

Rajagiri Valley,Kochi,682039

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

CERTIFICATE

Certified that this document titled ***'Traffic Sign Detection Using Neural Networks'*** *is a bonafide report of the project done by* ***ROBIN CR (RET17EC126)*** *of eighth semester Electronics and Communication Engineering in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Electronics and Communication Engineering of the A.P.J Abdul Kalam Technological University,Trivandrum,Kerala during the academic year 2020-2021.*

Head of Department
Dr. Rithu James
Head of Department
Department of ECE

Project Guide
Mr. Ajai V Babu
Assistant Professor
Department of ECE

Project Co-ordinator
Mr.Naveen N
Assistant Professor
Department of ECE

Place: Kakkanad

Date:15/JULY/2021

ACKNOWLEDGEMENTS

We are grateful to the almighty God for the blessings and for helping us to complete our work successfully.

I render my heartfelt gratitude to our principal, **Prof.Dr. P Sreejith** for providing the necessary infrastructure and labs.

We would like to thank the management and staff of RSET for providing us all facilities to complete the project. We would also like to thank **Dr .Rithu James** , HOD, Electronics and communication Department for her kind support.

In doing our project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. We would like to show our gratitude for giving us a good guideline for assignment through numerous consultations. We would like to express our deepest gratitude to all those who have directly and indirectly guided us in doing this project.

We heartily thank our guide , **Mr. Ajai V Babu** for her guidance and suggestions during this project work .We are highly indebted to our class teacher **Mr.Naveen N** and Asst. class teacher **Ms.Mariya Vincent** for their support and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

We especially thank our classmates and team members itself, who have made valuable comments and suggestions on this proposal . We thank all the people for their help directly and indirectly to complete our project .

ABSTRACT

Road signs give out a number of messages regarding the road and what you as a driver should expect on the road. They keep the traffic flowing freely by helping drivers reach their destinations and letting them know entry, exit and turn points in advance. Pre-informed drivers will naturally avoid committing mistakes or take abrupt turns causing bottlenecks. Road signs, indicating turns, directions and landmarks, also help to save time and fuel by providing information on the route to be taken to reach a particular destination. Road signs are placed in specific areas to ensure the safety of drivers. These markers let drivers know how fast to drive. They also tell drivers when and where to turn or not to turn. In order to be a terrific driver, you need to have an understanding of what the sign mean. Our project implements a procedure to extract the road sign from a natural complex image, processes it and alerts the driver using voice command. It is implemented in such a way that it acts as a boon to drivers to make easy decisions.

List of Figures

Contents

Chapter - 1

Introduction

1.1 Problem Definition

Traffic signs provide valuable information to drivers and other road users. They represent rules that are in place to keep you safe, and help to communicate messages to drivers and pedestrians that can maintain order and reduce accidents. In order to solve the concerns over road and transportation safety, automatic traffic sign detection and recognition (TSDR) system has been introduced.
An automatic TSDR system can detect and recognise traffic signs from and within images captured by cameras or imaging sensors.In adverse traffic conditions, the driver may not notice traffic signs, which may cause accidents. In such scenarios, the TSDR system comes into action.The main objective of the research on TSDR is to improve the robustness and efficiency of the TSDR system. To develop an automatic TSDR system is a tedious job given the continuous changes in the environment and lighting conditions.

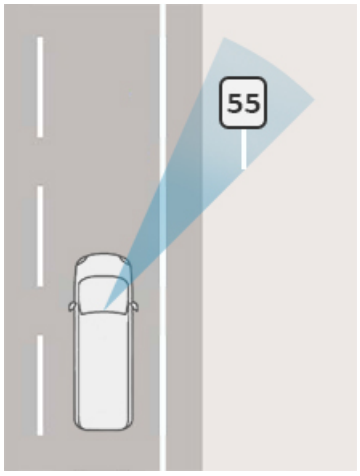

Figure 1.1: *Working of TSDR on road*

Among the other issues that also need to be addressed are partial obscuring, multiple traffic signs appearing at a single time, and blurring and fading of traffic signs, which can also create problem for the detection purpose. For applying the TSDR system in real time environment, a fast algorithm is needed. As well as dealing with these issues, a recognition system should also avoid erroneous recognition of non signs.

1.2 Objective

To detect the traffic sign boards and to provide a voice message according to the sign. Our goal is realized through the following objectives :

- Dataset preparation.
- Design an efficient algorithm for detecting traffic signs on the road.
- Train a model to detect traffic signs, classify them into one or more classes.
- Evaluate the detection accuracy and performance of the trained model.

1.3 Summary

The first chapter discusses the need for the proposed system followed by objectives which the system has to fulfill as a solution to the problem defined.

Chapter - 2

Literature Survey

2.1 Traffic sign recognition by Tesla motors

Tesla motors is accelerating towards a Fully Self Driving (FSD) cars. Tesla's approach to try to achieve FSD is to mimic how a human learns to drive,that is, by training a neural network using the behaviour of hundreds of thousands of Tesla drivers, and (from a sensor perspective), by relying chiefly on visible light cameras supplemented by radar and information from components used for other purposes in the car such as the coarse-grained two-dimensional maps used for navigation; the ultrasonic sensors used for parking, etc.

FSD is an optional upcoming extension of Autopilot to enable fully autonomous driving. TSDR is a part of their Autopilot mode. They could achieve it with the help of Artificial Intelligence(AI) and Internet of Things(IOT).Tesla's self-driving software is being trained on over 20 billion miles driven by Tesla vehicles as of January 2021. In terms of computing hardware, Tesla designed a self-driving computer chip that has been installed in its cars since March 2019.Since Tesla motors is working for autonomous driving they also research in Lane Detection, Pedestrian Detection, Traffic light Detection, Vehicle detection, Vehicle to Vehicle distance estimation etc. Tesla motors have introduced Traffic sign recognition as a software feature initially in 2019. The model could efficiently detects alot of traffic signs such as speed limit signs, stop sign etc.. After feedback from the drivers they have integrated this model with a GPS based navigation, so now speed limits are monitored based on GPS navigation. Now they are researching on integrating this software with mechanical system of the vehicle, for eg. if a pedestrian crossing sign is detected the vehicle will automatically reduce its speed. Tesla is also having a feature that whenever it detects a speed limit sign, it will automatically set speed limit to that number.

Tesla motors encountered lots of challenges as a part of traffic sign recognition feature. Stop sign which is used as general sign by private firms, is being detected by Tesla cars,

which creates a nuisance to the drivers while driving. Traffic sign recognition could not work efficiently when the car moves above a particular speed. Tesla cars need to be connected to a network while driving, if connection is lost, the communication between the sensors is interrupted. So in that case offline mode is activated which is less accurate, in most case the offline model in the car needs to analyse its decision with online server.

2.2 Traffic Sign Recognition by BOSCH

With the advancements in AI and the development of computing capabilities in the 21st century, millions of processes around the globe are being automated like never before. The automobile industry is transforming,and the day isn't far when fully autonomous vehicles would make transportation extremely inexpensive and effective. But to reach this ambitious goal, which aims to change the very foundations of transportation as an industry, we need to first solve a few challenging problems which will help a vehicle make decisions by itself. This is one such problem and solving it would take us one step closer to L5 autonomy.

The developed software package "Analysis.ai" is designed to help an analyst to build datasets for traffic sign recognition, implement wide variety of augmentation and transformation techniques on the dataset with wide flexibility. Furthermore, the package provides user with a latest deep learning model E-DUQ clubbed with an Spatial Transformer Network (STN) to not only make a prediction but also give information regarding noise and confidence level of the prediction. The detailed analysis of the trained model has been done using uncertainty, architecture and augmentative analysis.

2.2.1 STN Model

Spatial transformer networks are a generalization of differentiable attention to any spatial transformation. Spatial transformer networks (STN for short) allow a neural network to learn how to perform spatial transformations on the input image in order to enhance the geometric invariance of the model. For example, it can crop a region of interest, scale and correct the orientation of an image. It can be a useful mechanism because CNNs are not invariant to rotation and scale and more general affine transformations.

Chapter - 3

Traffic Sign Recognition

3.1 Problem

When driving on congested roads, it's sometimes difficult to keep your eyes everywhere at once. Checking the road ahead, oncoming traffic, what's behind you, all while trying to maintain your speed can sometimes become quite distracting. Car firms realise this and are constantly looking to introduce new technologies, which make things easier. With the introduction of Traffic Sign Recognition systems, the chances of not noticing a change in speed limit, or the warning of a potential hazard ahead have been vastly reduced. In the latest of our technical pieces, we explain how this clever system could benefit you.

Figure 3.1: *TSDR model*

3.2 How does it work?

As with most of the best technology, it's a relatively simple principle which has the potential to reduce the hassle for road users. Essentially, the system consists of a forward-facing camera, which scans the road ahead for traffic signs. This camera is connected to character recognition software, which then makes a note of any changes described by the signs, and relaying it onto the car's instrument panel.

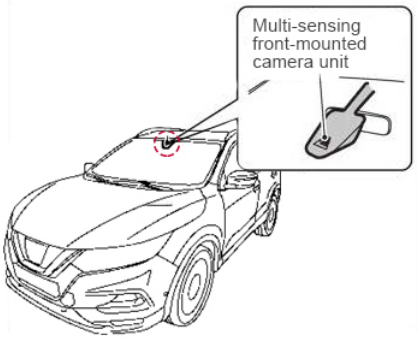

Figure 3.2: *Camera positioning*

3.3 Importance

Every country has some standards set for the design of different traffic signs like U-turn, Left-turn, Right-turn, No-entry, etc. Traffic sign recognition is the process of automatically identifying which of the following class the sign belongs to. The earlier Computer Vision techniques required lots of hard work in data processing and it took a lot of time to manually extract the features of the image. Now, deep learning techniques have come to the rescue and today we will see how to build a traffic recognition system for autonomous vehicles.

Traffic signs are an essential part of our day to day lives. They contain critical information that ensures the safety of all the people around us. Without traffic signs, all the drivers would be clueless about what might be ahead to them and roads can become a mess. The annual global roach crash statistics say that over 3,280 people die every day in a road accident. These numbers would be much higher in case if there were no traffic signs.

On the other hand, researchers and big companies are working extensively on proposing solutions to self-driving cars. Just to name a few these include Tesla, Uber, Google, Audi, BMW, Ford, Toyota, Mercedez, Volvo, Nissan, etc. These autonomous vehicles need to follow the traffic rules and for that, they have to understand the message conveyed through traffic signs.

Chapter - 4

Introduction to Neural Networks

4.1 Neural Networks

Neural networks are artificial systems that were inspired by biological neural networks. These systems learn to perform tasks by being exposed to various datasets and examples without any task-specific rules. The idea is that the system generates identifying characteristics from the data they have been passed without being programmed with a pre-programmed understanding of these datasets. Neural networks are based on computational models for threshold logic. Threshold logic is a combination of algorithms and mathematics. Neural networks are based either on the study of the brain or on the application of neural networks to artificial intelligence. The work has led to improvements in finite automata theory.

4.2 Evolution

In 1943, Warren McCulloch and Walter Pitts laid the first brick in the foundation of an advanced future of artificial neural networks. Warren McCulloch and Walter Pitts developed a mathematical model of an artificial neural network using threshold logic to mimic how a neuron works in a human brain. Thereafter, Frank Rosenblatt created the "Perceptron" model, which was the first of its kind to perform pattern recognition, in 1958. But, Marvin Minsky and Seymour Papert found multiple problems with the Perceptron model, which were later solved by Paul Werbos in 1975 using Back Propagation. Between 2009 and 2012, recurrent neural networks and deep feedforward neural networks were created by Jürgen Schmidhuber's research group, which won eight international competitions in pattern recognition and machine learning.

4.3 Limitations

The neural network is for a supervised model. It does not handle unsupervised machine learning and does not cluster and associate data. It also lacks a level of accuracy that will be found in more computationally expensive neural network.

The next steps would be to create an unsupervised neural network and to increase computational power for the supervised model with more iterations and threading.

4.4 Types of Neural Networks

4.4.1 Artificial Neural Networks(ANN)

ANN is a group of multiple perceptrons or neurons at each layer. ANN is also known as a Feed-Forward Neural network because inputs are processed only in the forward direction. This type of neural networks are one of the simplest variants of neural networks. They pass information in one direction, through various input nodes, until it makes it to the output node. The network may or may not have hidden node layers, making their functioning more interpretable. ANN learning is robust to errors in the training data and has been successfully applied for learning real-valued, discrete-valued, and vector-valued functions containing problems such as interpreting visual scenes, speech recognition, and learning robot control strategies. The study of artificial neural networks (ANNs) has been inspired in part by the observation that biological learning systems are built of very complex webs of interconnected neurons in brains.

Advantages

- Storing information on the entire network.
- Ability to work with incomplete knowledge.
- Having fault tolerance.
- Having a distributed memory.

Disadvantages

- Hardware dependence.
- Unexplained behavior of the network.
- Determination of proper network structure.

4.4.2 Convolutional Neural Network (CNN)

A convolutional neural network is a specific kind of neural network with multiple layers. It processes data that has a grid-like arrangement then extracts important features. One huge advantage of using CNNs is that you don't need to do a lot of pre-processing on images.With most algorithms that handle image processing, the filters are typically created by an engineer based on heuristics. CNNs can learn what characteristics in the filters are the most important. That saves a lot of time and trial and error work since we don't need as many parameters. A convolution is used instead of matrix multiplication in at least one layer of the CNN. Convolutions take to two functions and return a function.

CNNs work by applying filters to your input data. What makes them so special is that CNNs are able to tune the filters as training happens. That way the results are fine-tuned in real time, even when you have huge data sets, like with images.

Convolution Neural Networks or covnets are neural networks that share their parameters.A covnets is a sequence of layers, and every layer transforms one volume to another through differentiable function.

Layers used to build ConvNets

Convolution Layer: This layer computes the output volume by computing dot product between all filters and image patch.
Activation Function Layer: This layer will apply element wise activation function to the output of convolution layer. Some common activation functions are RELU: max(0, x), Sigmoid, Tanh, Leaky RELU, etc.
Pool Layer: This layer is periodically inserted in the covnets and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents from overfitting. Two common types of pooling layers are max pooling and average pooling.
Fully-Connected Layer: This layer is regular neural network layer which takes input from the previous layer and computes the class scores and outputs the 1-D array of size equal to the number of classes.

Training in CNN

Training a neural network typically consists of two phases:

- A forward phase, where the input is passed completely through the network.During the forward phase, each layer will cache any data (like inputs, intermediate values, etc) it'll need for the backward phase. This means that any backward phase must be preceded by a corresponding forward phase.

- A backward phase, where gradients are backpropagated and weights are updated.During the backward phase, each layer will receive a gradient and also return a gradient. It will receive the gradient of loss with respect to its outputs and return the gradient of loss with respect to its inputs.

Advantages

- Very High accuracy in image recognition problems.
- Automatically detects the important features without any human supervision.
- Weight sharing.

Disadvantages

- CNN do not encode the position and orientation of object.
- Lack of ability to be spatially invariant to the input data.
- Lots of training data is required.

Chapter - 5

System Design

5.1 Block Diagram

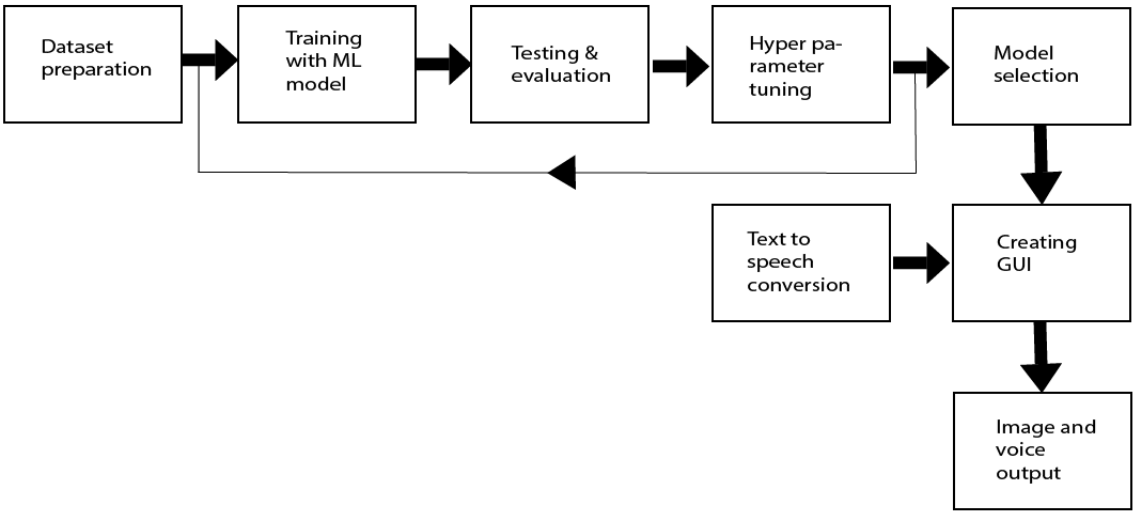

Figure 5.1: *Block diagram*

5.2 Gathering Data

The step of gathering data is the foundation of the machine learning process. Mistakes such as choosing the incorrect features or focusing on limited types of entries for the data set may render the model completely ineffective. That is why it is imperative that the necessary considerations are made when gathering data as the errors made in this stage would only amplify as we progress to latter stages.

5.3 Training

Training requires patience and experimentation. It is also useful to have knowledge of the field where the model would be implemented. For instance, if a machine learning model is to be used for identifying high risk clients for an insurance company, the knowledge of how the insurance industry operates would expedite the process of training as more educated guesses can be made during the iterations. Training can prove to be highly rewarding if the model starts to succeed in its role.

5.4 Testing and Evaluation

With the model trained, it needs to be tested to see if it would operate well in real world situations. That is why the part of the data set created for evaluation is used to check the model's proficiency. Evaluation becomes highly important when it comes to commercial applications. Evaluation allows data scientists to check whether the goals they set out to achieve were met or not. If the results are not satisfactory then the prior steps need to be revisited so that root cause behind the model's underperformance can be identified and, subsequently, rectified. If the evaluation is not done properly then the model may not excel at fulfilling its desired commercial purpose.

5.5 Hyper parameter tuning

If the evaluation is successful, we proceed to the step of hyperparameter tuning. This step tries to improve upon the positive results achieved during the evaluation step.here are different ways we can go about improving the model. One of them is revisiting the training step and use multiple sweeps of the training data set for training the model. This could lead to greater accuracy as the longer duration of training provides more exposure and improves quality of the model. Another way to go about it is refining the initial values given to the model. Random initial values often produce poor results as they are gradually refined by trial and error. However, if we can come up with better initial values or perhaps initiate the model using a distribution instead of a value then our results could get better. There are other parameters that we could play around with in order to refine the model but the process is more intuitive than logical so there is no definite approach for it.

5.6 Model Selection

The selection of the model type is our next course of action once we are done with the data-centric steps. There are various existing models developed by data scientists which can be used for different purposes. These models are designed with different goals in mind. In more complex scenarios, we need to make the choice that matches our intended outcome. The options for machine learning models can be explored across 3 broad categories. The first category is supervised learning models. In such models the outcome is known so we continuously refine the model itself until our output reaches the desired accuracy level. The linear regression model chosen for our fruit model is an example of supervised learning. If the outcome is unknown and we need classification to be done then the second category, unsupervised learning, is used. Examples of unsupervised learning include K-means and Apriori algorithm. The third category is reinforcement learning. It focuses on learning to make better decisions on the basis of trial and error. They are often used in business environments.

Chapter - 6

Implementation

6.1 Dataset Preparation

6.1.1 Collecting data

Collecting the images of these traffic signs is the most time consuming process. In our project we have used a German traffic sign dataset from the site kaggle.com. The signs which are common among German and Indian signs are filtered. There were around 35 signs which are common. After thorough research we have selected 12 signs, that if the driver didn't notice any of these sign boards the chances of getting to accident is very high. The signs are selected based on the collision accident rates in India.

6.1.2 Data Augmentation

During dataset preparation we encountered a problem called class skew. It arises when there is wide variation in training images among different classes. In our project we have around 500 images for "hump" sign but only 40 images for "no parking" signal. So the solution is data augmentation.Data augmentation in data analysis are techniques used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. It will create new images by changing the properties such as orientation, size etc.., so that there is enough data for each class.

6.2 Training

The model for traffic sign detection is implemented with the help of Convolutional Neural Network (CNN).The observations in the training set form the experience that the algorithm uses to learn.

6.2.1 Model Parameters

- No. of Filters = 60
- No. of nodes in hidden layers = 500
- Activation function in conv. layer = Relu
- Dropout = 0.5
- Activation function in output layer = Softmax
- Epochs = 10

```
Epoch 1/10
981/981 [==============================] - 74s 75ms/step - loss: 3.7390 - accuracy: 0.2706 -
val_loss: 0.8561 - val_accuracy: 0.8009
Epoch 2/10
981/981 [==============================] - 92s 93ms/step - loss: 1.0241 - accuracy: 0.6987 -
val_loss: 0.3275 - val_accuracy: 0.9083
Epoch 3/10
981/981 [==============================] - 95s 97ms/step - loss: 0.5496 - accuracy: 0.8361 -
val_loss: 0.1706 - val_accuracy: 0.9462
Epoch 4/10
981/981 [==============================] - 96s 98ms/step - loss: 0.4135 - accuracy: 0.8801 -
val_loss: 0.1109 - val_accuracy: 0.9663
Epoch 5/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2817 - accuracy: 0.9212 -
val_loss: 0.0887 - val_accuracy: 0.9755
Epoch 6/10
981/981 [==============================] - 98s 100ms/step - loss: 0.2694 - accuracy: 0.9227 -
val_loss: 0.0800 - val_accuracy: 0.9773
Epoch 7/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2519 - accuracy: 0.9291 -
val_loss: 0.0708 - val_accuracy: 0.9809
Epoch 8/10
981/981 [==============================] - 98s 100ms/step - loss: 0.2126 - accuracy: 0.9398 -
val_loss: 0.0887 - val_accuracy: 0.9768
Epoch 9/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2591 - accuracy: 0.9308 -
val_loss: 0.0646 - val_accuracy: 0.9821
Epoch 10/10
981/981 [==============================] - 97s 99ms/step - loss: 0.2193 - accuracy: 0.9406 -
val_loss: 0.1633 - val_accuracy: 0.9573
```

Figure 6.1: Accuracy of training

6.3 Voice assisted message

If a particular sign is detected a voice message is given according to the sign. Suppose "School Ahead" traffic sign is detected, then a voice message "School is Ahead, so please control your speed" is given by the system. This is implemented with the help of Google Text to Speech library.

6.4 Creating GUI

Figure 6.2: Graphical User Interface

6.5 Testing with images

In order to calculate the efficiency of the model, we need to test it under various conditions. Real time images from the roads were taken. The images were taken under the following conditions.

- Morning
- Evening
- Tilted
- Real time using web camera

Figure 6.3: Picture - morning

Figure 6.4: Picture - evening

Figure 6.5: Picture - tilted

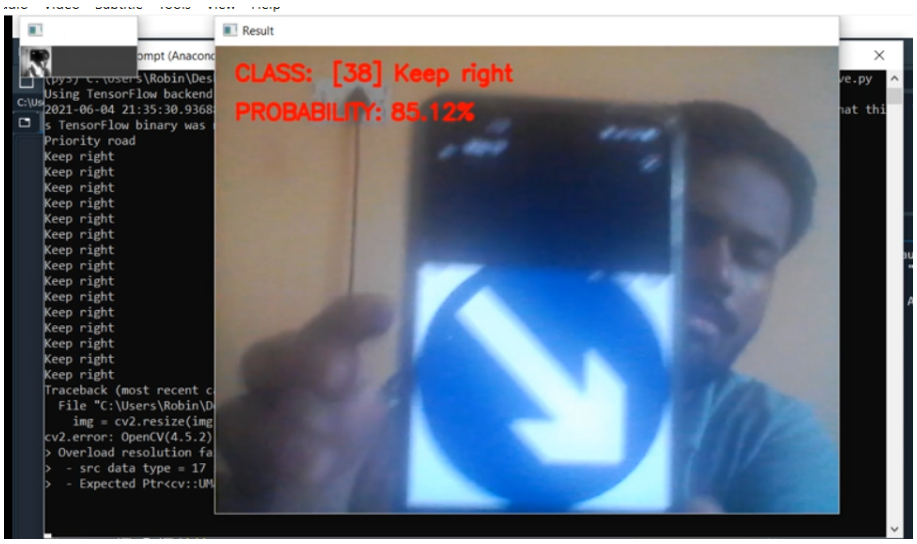

Figure 6.6: Real time

Chapter - 7

System Analysis

7.1 Accuracy

Machine learning model accuracy is the measurement used to determine which model is best at identifying relationships and patterns between variables in a dataset based on the input, or training, data. The better a model can generalize to 'unseen' data, the better predictions and insights it can produce, which in turn deliver more business value.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.1}$$

TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

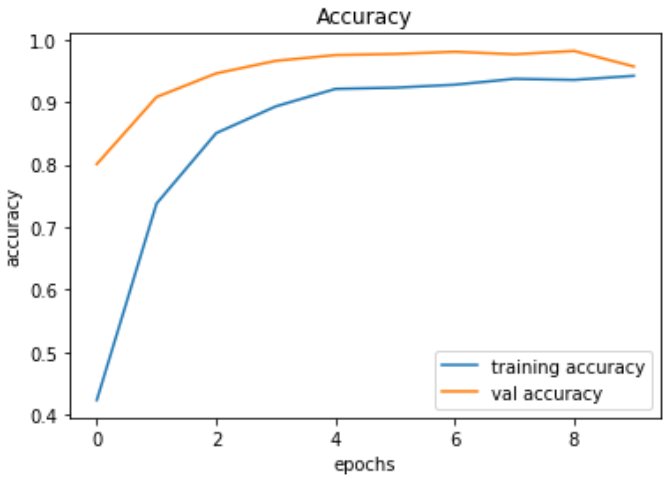

Figure 7.1: Accuracy plot

7.2 Loss

Cross-entropy loss is used when adjusting model weights during training. The aim is to minimize the loss, i.e, the smaller the loss the better the model. A perfect model has a cross-entropy loss of 0.

$$loss = -\sum_{i=0}^{n} t_i \log(p_i) \tag{7.2}$$

n= total no. of classes

t = Truth label for i th class

p = Softmax probability for i th class

Categorical cross-entropy is used when true labels are one-hot encoded.

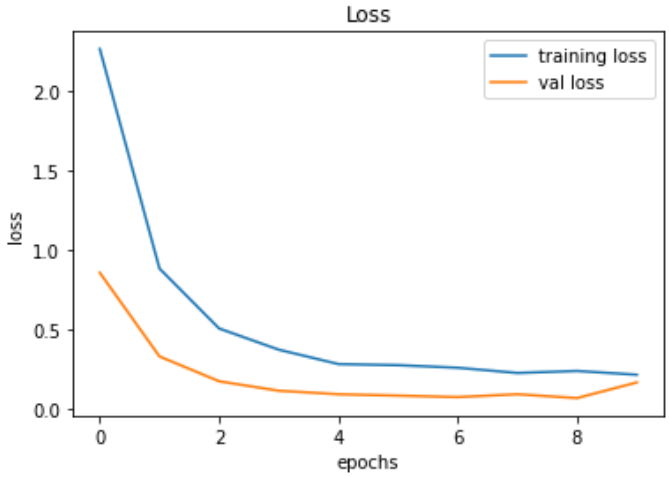

Figure 7.2: Loss plot

Chapter - 8

Results and Observations

- The German Traffic sign dataset containing over 43 traffic signs were preprocessed and trained using CNN algorithm.
- A User Interface is build for traffic sign detection.
- The text to speech conversion is done for Voice assisted message system.
- The model is tested with real time images and works perfectly.
- The CNN algorithm shows an accuracy of 0.9573.
- Categorical cross-entropy is calculated as a part of evaluating the performance of model.CNN shows a loss of 0.1633.

Chapter - 9

Conclusion and Future scope

The project focuses on a new way of traffic sign detection using CNN which is very effective among the Indian traffic signs on the road without the use of extensive and expensive evaluations.The model developed remains to be a foundation that can be adjusted to different types of vehicles, such as bus or trucks, by considering the properties and constraints that defines each one. This also serves as a meaningful development in field of neural networks and its learning systems, as well as provide valuable insights for future progress.

9.1 Future Scope

In the world of Artificial Intelligence and advancement in technologies, research in autonomous driving will dominate in such a way that the vehicles should be able to interpret traffic signs and make decisions accordingly.A lot of methods will be integrated based on colour, shape, priority etc, which can lead to an efficient model in respective countries.Natural obstacles such as fog, smoke etc cause noise while image acquisition, models will be developed which will overcome these challenges.

Bibliography

[1] Ma Weixin, Xiong Changzhen, Wang Cong and Shan Yanmei." A traffic sign detection algorithm based on deep convolutional neural network",*Eng.Appl. Artif. Intelli.*, 48, 2016.

[2] Kang Kim, Hee Seok Lee. "Simultaneous traffic sign detection and boundary estimation using convolutional neural network" *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[3] T.Tan, J.Zhang and L. Ma. "Invariant texture segmentation via circular gabor Filters", *Proc. Object Recognit. Supported User Interact. Service Robot*, pages 901-904, 2002.

[4] Bei Tong ,Hengliang Luo, Yi Yang and Fuchao Wu." Traffic sign recognition using a multi-task convolutional neural network" *IEEE Transactions on Intelligent Transportation Systems*, 2017.

Appendix

Appendix A

Presentation Slides

Traffic Sign Recognition

Robin CR - RET17EC126
Sanjay MS - RET17EC136
Vidhu Krishnan - RET17EC178
V S Achuthan - RET17EC186

S8 ECE Gamma
Rajagiri School of Engineering and Technology
Guided By: Mr. Ajai V Babu

June 16, 2021

TABLE OF CONTENTS

Problem Statement

Problem Statement

Problem Statement

- Traffic signs are sometime ignored by the drivers, which may lead to accidents - Traffic sign recognition

Problem Statement

Objective

Objective

To detect the traffic sign boards and to provide a voice message according to the sign.
Our goal is realized through the following objectives :

- Dataset preparation.
- Design an efficient algorithm for detecting traffic signs on the road.
- Train a model to detect traffic signs, classify them into one or more classes.
- Evaluate the detection accuracy and performance of the trained model.

Traffic Sign Recognition June 16, 2021 4 / 29

Block Diagram

Block Diagram

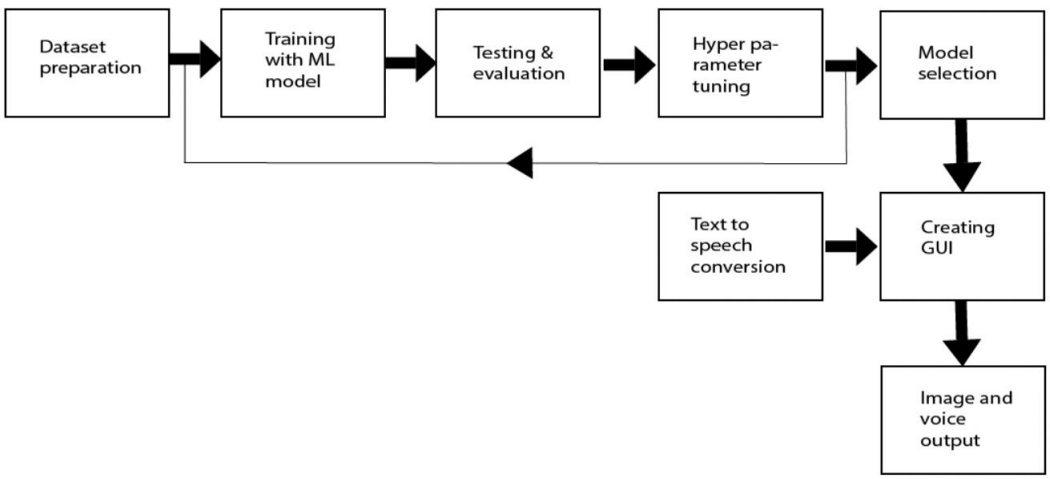

Figure 1: Block Diagram

Milestones Completed

Milestones Completed

Dataset Preparation

- In our project we have used a German traffic sign dataset from the site kaggle.com . The dataset include 43 signs. The signs which are common among German and Indian signs are filtered.
- After thorough research 12 signs are selected, that if the driver didn't notice any of these sign boards the chances of getting to accident is very high.
- Hindusthan Times published a report on road accidents in India on 2019. The traffic signs were selected based on the collision accidents mentioned in the report.

Milestones Completed

Dataset Summary

- Number of training examples = 34799
- Number of validation examples = 4410
- Number of testing examples = 12630
- Image data shape = (32, 32)
- Number of classes = 43

Milestones Completed

Data Augmentation

- During dataset preparation we encountered a problem called class skew. It arises when there is wide variation in training images among different classes.
- In our project there is around 500 images for "Hump" sign but only 40 images for "No parking" signal. So the solution is data augmentation.
- It will create new images by changing the properties such as orientation, size etc.., so that there is enough data for each class.

Milestones Completed

Milestones Completed

Building a Model

- A deep learning model is developed with the help of neural network.
- Convolutional Neural Network (CNN) algorithm is used in our project.
- The model for traffic sign detection is implemented with the help of CNN.
- CNNs can learn what characteristics in the filters are the most important. That saves a lot of time and trial and error work since there is no need to look for that parameters.

Traffic Sign Recognition June 16, 2021 9 / 29

Milestones Completed

Voice Assistant Setup

- If a particular sign is detected a voice message is given according to the sign.
- Suppose "School Ahead" traffic sign is detected, then a voice message "School is Ahead, so please control your speed" is given by the system.
- This is implemented with the help of Google Text to Speech library in python.

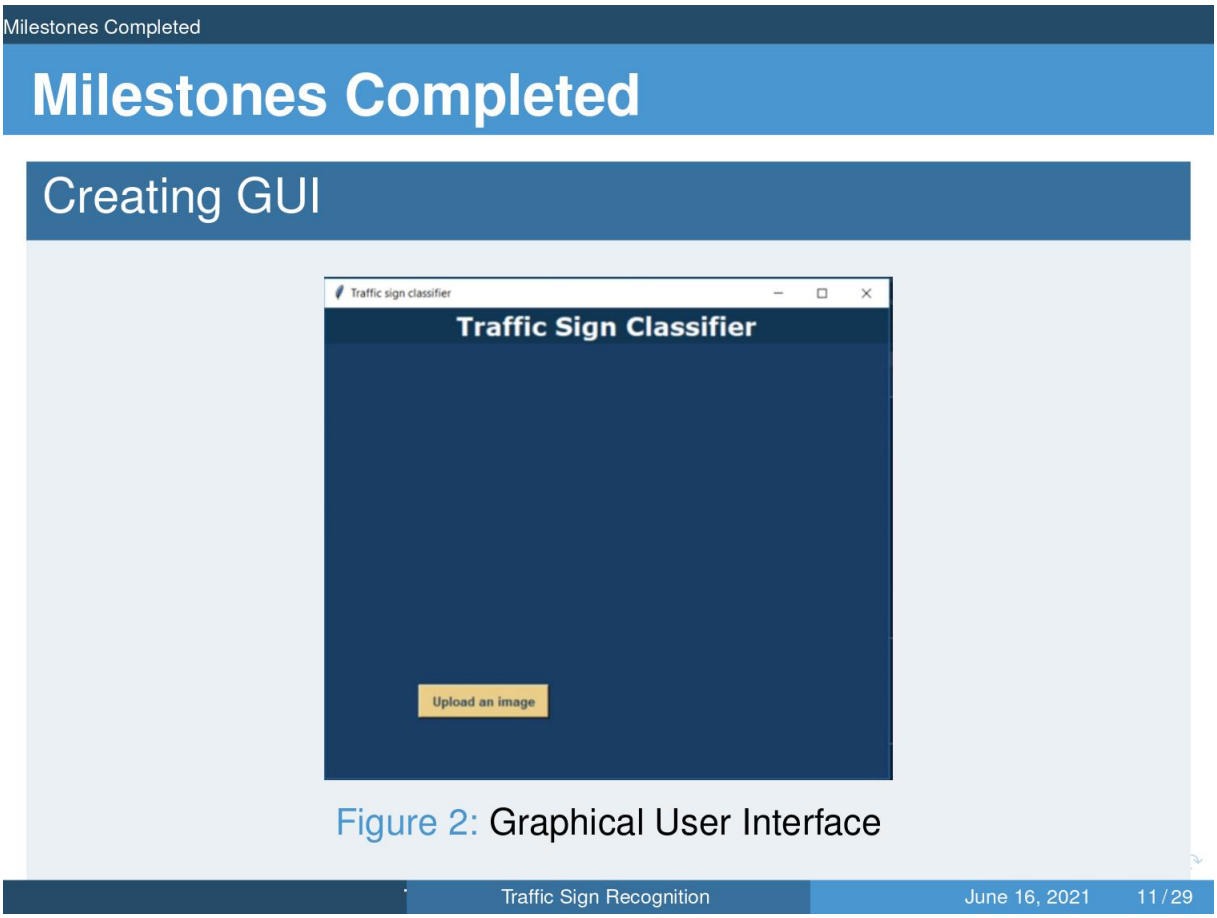

Figure 2: Graphical User Interface

Training in CNN

Training in CNN

CNN consists of three layers Convolution layer, a Max Pooling layer, and a Softmax layer.

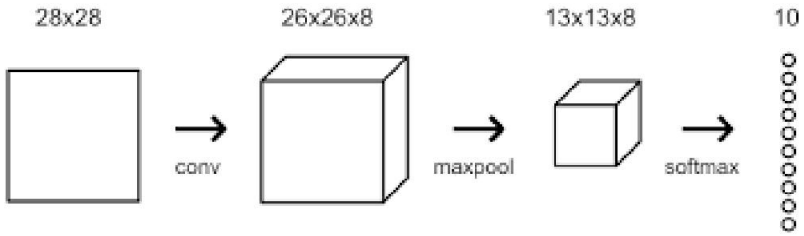

Figure 3: CNN layers

Training in CNN

Training in CNN

Training a neural network typically consists of two phases:

- A forward phase, where the input is passed completely through the network.During the forward phase, each layer will cache any data (like inputs, intermediate values, etc) it'll need for the backward phase. This means that any backward phase must be preceded by a corresponding forward phase.
- A backward phase, where gradients are backpropagated and weights are updated.During the backward phase, each layer will receive a gradient and also return a gradient. It will receive the gradient of loss with respect to its outputs and return the gradient of loss with respect to its inputs.

Model Parameters

Model Parameters

- No. of Filters = 60
- No. of nodes in hidden layers = 500
- Activation function in conv. layer = Relu
- Dropout = 0.5
- Activation function in output layer = Softmax
- Epochs = 10

Output of Training

Output of training

```
Epoch 1/10
981/981 [==============================] - 74s 75ms/step - loss: 3.7390 - accuracy: 0.2706 -
val_loss: 0.8561 - val_accuracy: 0.8009
Epoch 2/10
981/981 [==============================] - 92s 93ms/step - loss: 1.0241 - accuracy: 0.6987 -
val_loss: 0.3275 - val_accuracy: 0.9083
Epoch 3/10
981/981 [==============================] - 95s 97ms/step - loss: 0.5496 - accuracy: 0.8361 -
val_loss: 0.1706 - val_accuracy: 0.9462
Epoch 4/10
981/981 [==============================] - 96s 98ms/step - loss: 0.4135 - accuracy: 0.8801 -
val_loss: 0.1109 - val_accuracy: 0.9663
Epoch 5/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2817 - accuracy: 0.9212 -
val_loss: 0.0887 - val_accuracy: 0.9755
Epoch 6/10
981/981 [==============================] - 98s 100ms/step - loss: 0.2694 - accuracy: 0.9227 -
val_loss: 0.0800 - val_accuracy: 0.9773
Epoch 7/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2519 - accuracy: 0.9291 -
val_loss: 0.0708 - val_accuracy: 0.9809
Epoch 8/10
981/981 [==============================] - 98s 100ms/step - loss: 0.2126 - accuracy: 0.9398 -
val_loss: 0.0887 - val_accuracy: 0.9768
Epoch 9/10
981/981 [==============================] - 96s 98ms/step - loss: 0.2591 - accuracy: 0.9308 -
val_loss: 0.0646 - val_accuracy: 0.9821
Epoch 10/10
981/981 [==============================] - 97s 99ms/step - loss: 0.2193 - accuracy: 0.9406 -
val_loss: 0.1633 - val_accuracy: 0.9573
```

Figure 4: Accuracy of training

Constraints

Constraints

- Road and Traffic conditions in various regions.
- Time constraints can restrict a certain property of the system. These could be events either stimuli from the environment (e.g. camera sends in an image) or responses from the system (e.g. control outputs an action).
- Response time is less than 300ms.

Testing

Testing the model

In order to calculate the efficiency of the model, we need to test it under various conditions. Real time images from the roads were taken. The images were taken under the following conditions.

- Morning
- Evening
- Tilted
- Real time using web camera

Testing

Test images - morning

Figure 5: Picture - morning

Testing

Test images - evening

Figure 6: Picture - evening

Testing

Test images - tilted

Figure 7: Picture - tilted

Testing

Real time/ video

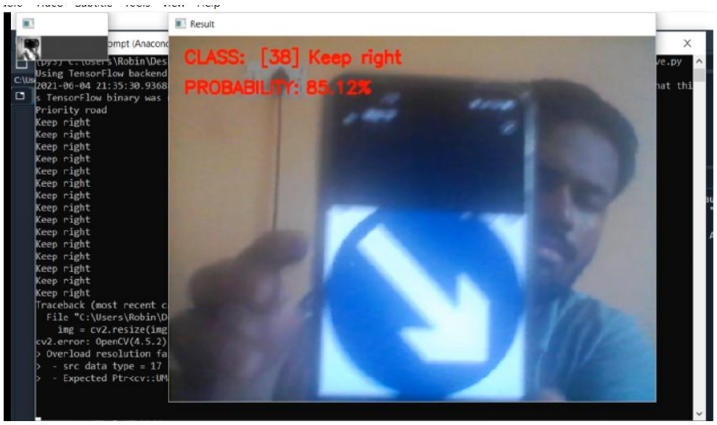

Figure 8: Real time

Evaluation

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

Evaluation

Accuracy

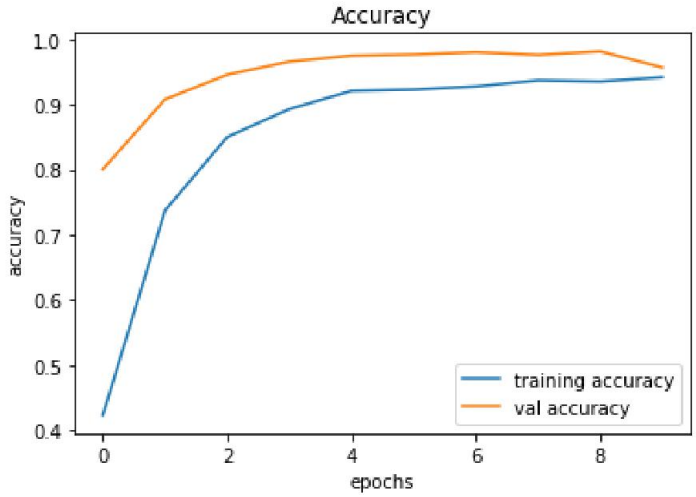

Figure 9: Accuracy plot

Evaluation

Loss

Cross entropy loss

Cross-entropy loss is used when adjusting model weights during training. The aim is to minimize the loss, i.e, the smaller the loss the better the model. A perfect model has a cross-entropy loss of 0.

$$loss = -\sum_{i=0}^{n} t_i \log(p_i) \tag{2}$$

n= total no. of classes
t = Truth label for i th class
p = Softmax probability for i th class
Categorical cross-entropy is used when true labels are one-hot encoded.

Evaluation

Loss

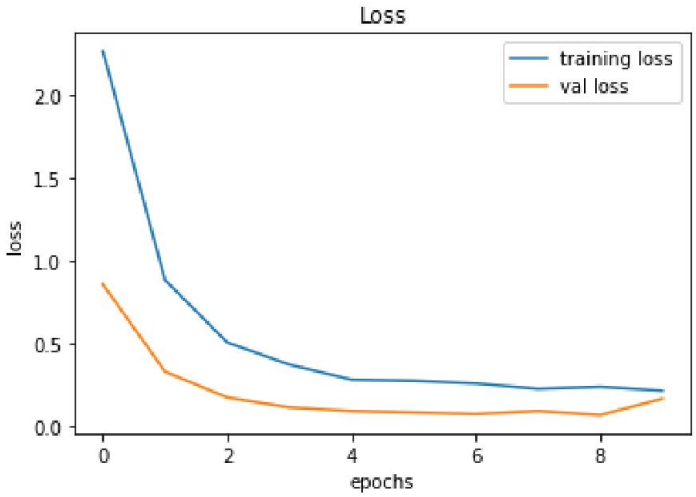

Figure 10: Loss plot

Results

Results

- The German Traffic sign dataset containing over 43 traffic signs were preprocessed and trained using CNN algorithm.
- A User Interface is build for traffic sign detection.
- The text to speech conversion is done for Voice assisted message system.
- The model is tested with real time images and works perfectly.
- The CNN algorithm shows an accuracy of 0.9573.
- Categorical cross-entropy is calculated as a part of evaluating the performance of model.CNN shows a loss of 0.1633.

Future Scope

Future Scope

- In the world of Artificial Intelligence and advancement in technologies, research in autonomous driving will dominate in such a way that the vehicles should be able to interpret traffic signs and make decisions accordingly.
- A lot of methods will be integrated based on colour, shape, priority etc, which can lead to an efficient model in respective countries.
- Natural obstacles such as fog, smoke etc cause noise while image acquisition, models will be developed which will overcome these challenges.

Conclusion

Conclusion

- The project focuses on a new way of traffic sign detection using CNN which is very effective among the Indian traffic signs on the road without the use of extensive and expensive evaluations.
- The model developed remains to be a foundation that can be adjusted to different types of vehicles, such as bus or trucks, by considering the properties and constraints that defines each one.
- This also serves as a meaningful development in field of neural networks and its learning systems, as well as provide valuable insights for future progress.

Reference

Reference

[1] Ma Weixin, Xiong Changzhen, Wang Cong and Shan Yanmei." A traffic sign detection algorithm based on deep convolutional neural network",*Eng.Appl. Artif. Intelli.*, 48, 2016.

[2] Kang Kim, Hee Seok Lee. "Simultaneous traffic sign detection and boundary estimation using convolutional neural network" *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[3] T.Tan, J.Zhang and L. Ma. "Invariant texture segmentation via circular gabor Filters", *Proc. Object Recognit. Supported User Interact. Service Robot*, pages 901-904, 2002.

[4] Bei Tong ,Hengliang Luo, Yi Yang and Fuchao Wu." Traffic sign recognition using a multi-task convolutional neural network" *IEEE Transactions on Intelligent Transportation Systems*, 2017.

Reference

THANK YOU

Appendix B

Questionnaire

1.What are the challenges to Traffic Sign Detection and Recognition?
In TSDR moving camera needs to detect stationary objects, so if the speed of camera increases beyond a threshold value noise adds to the image. So in that case TSDR system will not detect the signs. Secondly signs from private firms are mistakenly detected which creates a problem to the driver.

2.Define Neural Networks?
A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

3.What is class skew?
Skewed classes basically refer to a dataset, wherein the number of training example belonging to one class out-numbers heavily the number of training examples belonging to the other. for example in our project we have around 500 images for "Hump" traffic sign but only 200 images for "railway cross".

4.What is convolution in convolution neural networks(CNN)?
Convolution is a specialized kind of linear operation. Convnets are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.Convolution between two functions in mathematics produces a third function expressing how the shape of one function is modified by other.

5.Differentiate training data and testing data?

Training Data : The observations in the training set form the experience that the algorithm uses to learn. In supervised learning problems, each observation consists of an observed output variable and one or more observed input variables.

Test Data : The test set is a set of observations used to evaluate the performance of the model using some performance metric. It is important that no observations from the training set are included in the test set. If the test set does contain examples from the training set, it will be difficult to assess whether the algorithm has learned to generalize from the training set or has simply memorized it.

Appendix C

Python Code

C.1 Training

```
import numpy as np
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


################### Parameters #####################

path = "myData" # folder with all the class folders
labelFile = 'labels.csv' # file with all names of classes
batch_size_val=50  # how many to process together
steps_per_epoch_val=2000
epochs_val=2
```

```
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 fo
##############################################################


################################### 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)

################################### Split Data
X_train, X_test, y_train, y_test = train_test_split(images, classNo, test
X_train, X_validation, y_train, y_validation = train_test_split(X_train, y

# X_train = ARRAY OF IMAGES TO TRAIN
# y_train = CORRESPONDING CLASS ID

################################### TO CHECK IF NUMBER OF IMAGES MATCHES TO NU
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 e
assert(X_validation.shape[0]==y_validation.shape[0]), "The number of image
assert(X_test.shape[0]==y_test.shape[0]), "The number of images in not equ
assert(X_train.shape[1:]==(imageDimesions)),"  The dimesions of the Trainin
assert(X_validation.shape[1:]==(imageDimesions)),"  The dimesionas of the V
assert(X_test.shape[1:]==(imageDimesions)),"  The dimesionas of the Test im


################################### READ CSV FILE
data=pd.read_csv(labelFile)
print("data shape ",data.shape,type(data))

################################### DISPLAY SOME SAMPLES IMAGES  OF ALL THE CL
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)
        axs[j][i].axis("off")
        if i == 2:
            axs[j][i].set_title(str(j)+ "-"+row["Name"])
            num_of_samples.append(len(x_selected))


################################### DISPLAY A BAR CHART SHOWING NO OF SAMPLES
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
    return img

X_train=np.array(list(map(preprocessing,X_train)))  # TO IRETATE AND PREPR
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.shape[2]
X_validation=X_validation.reshape(X_validation.shape[0],X_validation.shape
X_test=X_test.reshape(X_test.shape[0],X_test.shape[1],X_test.shape[2],1)


#################################### AUGMENTATAION OF IMAGES: TO MAKEIT MORE GE
dataGen= ImageDataGenerator(width_shift_range=0.1,   # 0.1 = 10%
IF MORE THAN 1 E.G 10 THEN IT REFFERS TO NO. OF  PIXELS EG 10 PIXELS
                            height_shift_range=0.1,
                            zoom_range=0.2,  # 0.2 MEANS CAN GO FROM 0.8 T
                            shear_range=0.1,  # MAGNITUDE OF SHEAR ANGLE
                            rotation_range=10)  # DEGREES
dataGen.fit(X_train)
batches= dataGen.flow(X_train,y_train,batch_size=20)  # REQUESTING DATA GE
BATCH SIZE = NO. OF IMAGES CREAED EACH TIME ITS CALLED
X_batch,y_batch = next(batches)

# TO SHOW AGMENTED IMAGE SAMPLES
fig,axs=plt.subplots(1,15,figsize=(20,5))
```

```
fig.tight_layout()

for i in range(15):
    axs[i].imshow(X_batch[i].reshape(imageDimesions[0],imageDimesions[1]))
    axs[i].axis('off')
plt.show()


y_train = to_categorical(y_train,noOfClasses)
y_validation = to_categorical(y_validation,noOfClasses)
y_test = to_categorical(y_test,noOfClasses)

#################################### CONVOLUTION NEURAL NETWORK MODEL
def myModel():
    no_Of_Filters=60
    size_of_Filter=(5,5) # THIS IS THE KERNEL THAT MOVE AROUND THE IMAGE T
                         # THIS WOULD REMOVE 2 PIXELS FROM EACH BORDER WHE
    size_of_Filter2=(3,3)
    size_of_pool=(2,2)  # SCALE DOWN ALL FEATURE MAP TO GERNALIZE MORE, TO
    no_Of_Nodes = 500   # NO. OF NODES IN HIDDEN LAYERS
    model= Sequential()
    model.add((Conv2D(no_Of_Filters,size_of_Filter,input_shape=(imageDime
# ADDING MORE CONVOLUTION LAYERS = LESS FEATURES BUT CAN CAUSE ACCURACY TO
    model.add((Conv2D(no_Of_Filters, size_of_Filter, activation='relu')))
    model.add(MaxPooling2D(pool_size=size_of_pool)) # DOES NOT EFFECT THE

    model.add((Conv2D(no_Of_Filters//2, size_of_Filter2,activation='relu'))
    model.add((Conv2D(no_Of_Filters // 2, size_of_Filter2, activation='rel
    model.add(MaxPooling2D(pool_size=size_of_pool))
    model.add(Dropout(0.5))

    model.add(Flatten())
    model.add(Dense(no_Of_Nodes,activation='relu'))
    model.add(Dropout(0.5)) # INPUTS NODES TO DROP WITH EACH UPDATE 1 ALL
    model.add(Dense(noOfClasses,activation='softmax')) # OUTPUT LAYER
    # COMPILE MODEL
    model.compile(Adam(lr=0.001),loss='categorical_crossentropy',metrics=
    return model
```

```
##################################### TRAIN
model = myModel()
print(model.summary())
history=model.fit_generator(dataGen.flow(X_train,y_train,batch_size=batch

##################################### PLOT
plt.figure(1)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.legend(['training','validation'])
plt.title('loss')
plt.xlabel('epoch')
plt.figure(2)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.legend(['training','validation'])
plt.title('Acurracy')
plt.xlabel('epoch')
plt.show()
score =model.evaluate(X_test,y_test,verbose=0)
print('Test Score:',score[0])
print('Test Accuracy:',score[1])


# STORE THE MODEL AS A PICKLE OBJECT
pickle_out= open("model_trained.p","wb")  # wb = WRITE BYTE
pickle.dump(model,pickle_out)
pickle_out.close()
cv2.waitKey(0)
```

C.2 Realtime Detection

```
import numpy as np
import cv2
import pickle
import os

#############################################################

frameWidth= 640         # CAMERA RESOLUTION
frameHeight = 1000
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
pickle_in=open("model_trained.p","rb")  ## rb = READ BYTE
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 to
    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 me

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)
    cv2.putText(imgOrignal, "PROBABILITY: ", (20, 75), font, 0.75, (0, 0,
    # PREDICT IMAGE
    predictions = model.predict(img)
    classIndex = model.predict_classes(img)
    probabilityValue =np.amax(predictions)
    if probabilityValue > threshold:
    #print(getCalssName(classIndex))
        cv2.putText(imgOrignal,str(classIndex)+" "+str(getCalssName(classI
        cv2.putText(imgOrignal, str(round(probabilityValue*100,2) )+"%", (
        cv2.imshow("Result", imgOrignal)
        print(getCalssName(classIndex))
        x=getCalssName(classIndex)




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

        break
```

C.3 Graphical User Interface

```
import tkinter as tk
from tkinter import filedialog
from tkinter import *
from PIL import ImageTk, Image


import numpy
import shutil
from tensorflow.keras.models import load_model
import warnings
from playsound import playsound
from gtts import gTTS
import os


model = load_model('Trafic_signs_model.h5')
warnings.filterwarnings("ignore", category=DeprecationWarning)



#dictionary to label all traffic signs class.
classes = { 1:'Speed limit (20km/h)',
            2:'Speed limit (30km/h)',
            3:'Speed limit (50km/h)',
            4:'Speed limit (60km/h)',
            5:'Speed limit (70km/h)',
            6:'Speed limit (80km/h)',
            7:'End of speed limit (80km/h)',
            8:'Speed limit (100km/h)',
            9:'Speed limit (120km/h)',
           10:'No passing',
           11:'No passing veh over 3.5 tons',
           12:'Right-of-way at intersection',
           13:'Priority road',
           14:'Yield',
```

```
    15:'Stop',
    16:'No vehicles',
    17:'Veh > 3.5 tons prohibited',
    18:'No entry',
    19:'General caution',
    20:'Dangerous curve left',
    21:'Dangerous curve right',
    22:'Double curve',
    23:'Bumpy road',
    24:'Slippery road',
    25:'Road narrows on the right',
    26:'Road work',
    27:'Traffic signals',
    28:'Pedestrians',
    29:'Children crossing',
    30:'Bicycles crossing',
    31:'Beware of ice/snow',
    32:'Wild animals crossing',
    33:'End speed + passing limits',
    34:'Turn right ahead',
    35:'Turn left ahead',
    36:'Ahead only',
    37:'Go straight or right',
    38:'Go straight or left',
    39:'Keep right',
    40:'Keep left',
    41:'Roundabout mandatory',
    42:'End of no passing',
    43:'End no passing veh > 3.5 tons' }



window=tk.Tk()
window.geometry('600x500')
window.title('Traffic sign classifier')

window.configure(background='#1e3e64')
```

```
heading = Label(window, text="Traffic Sign Classifier",padx=220, font=('V
heading.configure(background='#143953',foreground='white')
heading.pack()

sign = Label(window)
sign.configure(background='#1e3e64')

value = Label(window,font=('Helvetica',15,'bold'))
value.configure(background='#1e3e64')



def classify(file_path):
    global label_packed
    image = Image.open(file_path)
    image = image.resize((30,30))
    image = numpy.expand_dims(image, axis=0)
    image = numpy.array(image)
    #print(image.shape)
    pred = model.predict_classes([image])[0]
    sign = classes[pred+1]
    print(pred)
    print(sign)
    value.configure(foreground='#ffffff', text=sign)
    x = sign
    tts = gTTS(text=x, lang='en')
    ttsname=("name.mp3")
    tts.save(ttsname)
    playsound("name.mp3")
    os.remove(r"C:\Users\Robin\Desktop\project\DatasetCNN\archive\name.mp3



def show_cb(file_path):
    classify_b=Button(window,text="Classify Image",command=lambda: classif
    classify_b.configure(background='#147a81', foreground='white',font=('a
    classify_b.place(relx=0.6,rely=0.80)

def uploader():
```

```
    try:
        file_path = filedialog.askopenfilename()
        uploaded = Image.open(file_path)
        uploaded.thumbnail(((window.winfo_width()/2.25),(window.winfo_heig
        im = ImageTk.PhotoImage(uploaded)

        sign.configure(image=im)
        sign.image=im
        value.configure(text='')
        show_cb(file_path)
    except:
        pass

upload = Button(window,text="Upload an image",command=uploader,padx=10,pad
upload.configure(background='#e8d08e', foreground='#143953',font=('arial',
upload.pack()
upload.place(x=100, y=400)

sign.pack()
sign.place(x=230,y=100)
value.pack()
value.place(x=240,y=300)
window.mainloop()
```

Appendix D

IEEE PAPER

Traffic Sign Recognition Using Neural Networks

Robin CR[1]
B.Tech, Electronics and Communication Engineering[1]
Rajagiri School of Engineering and Technology[1]
Kochi, India
robin005cr@gmail.com[1]

Sanjay MS[2]
B.Tech, Electronics and Communication Engineering[2]
Rajagiri School of Engineering and Technology[2]
Kochi, India
sanjayasap2@gmail.com[2]

Vidhu Krishnan[3]
B.Tech, Electronics and Communication Engineering[3]
Rajagiri School of Engineering and Technology[3]
Kochi, India
vidhu099@gmail.com[3]

VS Achuthan[4]
B.Tech, Electronics and Communication Engineering[4]
Rajagiri School of Engineering and Technology[4]
Kochi, India
achuvadassery1999@gmail.com[4]

Ajai V Babu[5]
Asst. Professor, Dept. of ECE[5]
Rajagiri School of Engineering and Technology[5]
Kochi, India
ajaiv@rajaritech.edu.in[5]

***Abstract*—Road signs give out a number of messages regarding the road and what you as a driver should expect on the road. They keep the traffic flowing freely by helping drivers reach their destinations and letting them know entry, exit and turn points in advance. Pre-informed drivers will naturally avoid committing mistakes or take abrupt turns causing bottlenecks. Road signs, indicating turns, directions and landmarks, also help to save time and fuel by providing information on the route to be taken to reach a particular destination. Road signs are placed in specific areas to ensure the safety of drivers. These markers let drivers know how fast to drive. They also tell drivers when and where to turn or not to turn. In order to be a terrific driver, you need to have an understanding of what the sign mean. Our project implements a procedure to extract the road sign from a natural complex image, processes it and alerts the driver using voice command. It is implemented in such a way that it acts as a boon to drivers to make easy decisions.**

***Index Terms*—Neural Networks,Accuracy,Cross Entropy,Recognition**

I. Introduction

Traffic signs provide valuable information to drivers and other road users. They represent rules that are in place to keep you safe, and help to communicate messages to drivers and pedestrians that can maintain order and reduce accidents. In order to solve the concerns over road and transportation safety, automatic traffic sign detection and recognition (TSDR) system has been introduced. An automatic TSDR system can detect and recognise traffic signs from and within images captured by cameras or imaging sensors.In adverse traffic conditions, the driver may not notice traffic signs, which may cause accidents. In such scenarios, the TSDR system comes into action.The main objective of the research on TSDR is to improve the robustness and efficiency of the TSDR system. To develop an automatic TSDR system is a tedious job given the continuous changes in the environment and lighting conditions. Among the other issues that also need to be addressed are partial obscuring, multiple traffic signs appearing at a single time, and blurring and fading of traffic signs, which can also create problem for the detection purpose. For applying the TSDR system in real time environment, a fast algorithm is needed. As well as dealing with these issues, a recognition system should also avoid erroneous recognition of non signs.

Fig. 1. Traffic sign detection model

II. INDIAN ROAD TRAFFIC SIGNS

Indian Road Traffic Signs are standardized and pertinent nationwide, these are broadly classified into the following categories.

1) Regulatory signs-These signs inform the road users about the laws and regulations they have to follow. Violation of these signs is legal offence. They are circular in shape with red circumference.

2) Compulsory signs-These signs are an extension to regulatory signs and similar to the violation of regulatory signs, violation of these is a legal offence, which makes them most important signs. They are circular in shape and are filled with blue color and white circumference.
3) Warning signs-These signs warn road users of certain hazardous conditions. They are triangular in shape and possess a red circumference.
4) Informatory signs - These signs provide information and guidance to road users. They are rectangular and may vary in color, in some cases they might be green with white circumference whereas in others it might be white filled rectangle with blue circumference. Further classification of each of these categories is based on the information contained by these signs which may be a picture, alphanumeric string or a particular direction indicating arrow.

III. SYSTEM OVERVIEW

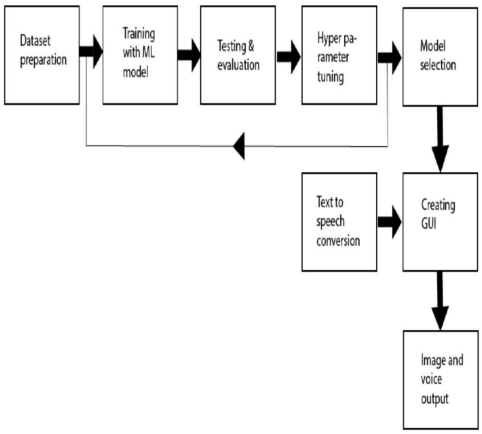

Fig. 2. Block Diagram

A. Dataset Preparation

Collecting the images of these traffic signs is the most time consuming process. In our project we have used a German traffic sign dataset from the site kaggle.com. The signs which are common among German and Indian signs are filtered. There were around 35 signs which are common. After thorough research we have selected 12 signs, that if the driver didn't notice any of these sign boards the chances of getting to accident is very high. The signs are selected based on the collision accident rates in India.

B. Training

The model for traffic sign detection is implemented with the help of Convolutional Neural Network (CNN).The observations in the training set form the experience that the algorithm uses to learn.

C. Testing

In order to calculate the efficiency of the model, we need to test it under various conditions. Real time images from the roads were taken. The images were taken under the following conditions. • Morning • Evening • Tilted • Real time using web camera

Fig. 3. Test image - morning

D. Voice Assisted Messages

If a particular sign is detected a voice message is given according to the sign. Suppose "School Ahead" traffic sign is detected, then a voice message "School is Ahead, so please control your speed" is given by the system. This is implemented with the help of Google Text to Speech library.

IV. SYSTEM ANALYSIS

Machine learning model accuracy is the measurement used to determine which model is best at identifying relationships and patterns between variables in a dataset based on the input, or training, data. The better a model can generalize to 'unseen' data, the better predictions and insights it can produce, which in turn deliver more business value.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative Cross-entropy loss is used when adjusting model weights during training. The aim is to minimize the loss, i.e, the smaller the loss the better the model. A perfect model has a cross-entropy loss of 0.

$$loss = -\sum_{i=0}^{n} t_i \log(p_i) \tag{2}$$

n= total no. of classes
t = Truth label for i th class
p = Softmax probability for i th class
Categorical cross-entropy is used when true labels are one-hot encoded.

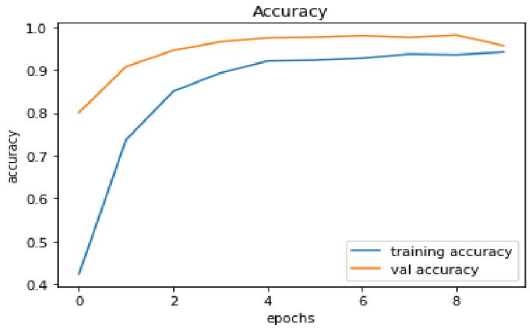

Fig. 4. Accuracy plot

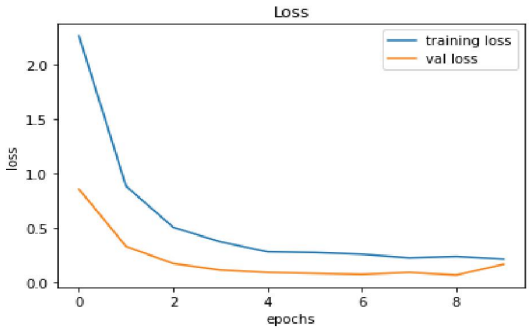

Fig. 5. Loss plot

V. RESULTS

The German Traffic sign dataset containing over 43 traffic signs were preprocessed and trained using CNN algorithm.A User Interface is build for traffic sign detection. The text to speech conversion is done for Voice assisted message system. The model is tested with real time images and works perfectly. The CNN algorithm shows an accuracy of 0.9573. Categorical cross-entropy is calculated as a part of evaluating the performance of model.CNN shows a loss of 0.1633.

CONCLUSION

This paper focuses on a new way of traffic sign detection using CNN which is very effective among the Indian traffic signs on the road without the use of extensive and expensive evaluations.The model developed remains to be a foundation that can be adjusted to different types of vehicles, such as bus or trucks, by considering the properties and constraints that defines each one. This also serves as a meaningful development in field of neural networks and its learning systems, as well as provide valuable insights for future progress

ACKNOWLEDGMENT

We would like to acknowledge our Guide, Coordinator and our institution for extending their help and support in completing our work.

REFERENCES

[1] O. Dabeer *et al.*, "An end-to-end system for crowdsourced 3D maps for autonomous vehicles: The mapping component," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 634–641.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. (2015). "SegNet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling." [Online]. Available: https://arxiv.org/abs/1505.07293

[3] J. Uhrig, M. Cordts, U. Franke, and T. Brox. (2016). "Pixel-level encoding and depth layering for instance-level semantic labeling." [Online]. Available: https://arxiv.org/abs/1604.05096

[4] G. Lin, C. Shen, A. van den Hengel, and I. Reid. (2016). "Exploring context with deep structured models for semantic segmentation." [Online]. Available: https://arxiv.org/abs/1603.03183

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[6] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[8] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The german traffic sign detection benchmark," in *Proc. Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–8.

[9] Y. Yang, H. Luo, H. Xu, and F. Wu, "Towards real-time traffic sign detection and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 2022–2031, Jul. 2016.

[10] Y. Yang and F. Wu, "Real-time traffic sign detection via color probability model and integral channel features," in *Proc. Chin. Conf. Pattern Recognit.*, Nov. 2014, pp. 545–554.

Appendix E

Mapping The Project Objectives with POs and PSOs

Sl. No	PROJECT OBJECTIVES	POs	PSOs
1.	Implemented Traffic sign recognition using deep learning technique called Convolutional Neural Network.	PO4, PO5	PSO2
2.	Performance analysis was implemented along with comparison of multiple algorithms to find out the one with most accurate results	PO2	
3.	Improved programming skills by working on python and familiarised with major libraries in python such as tkinter,opencv,tensorflow,gtts.	PO1, PO5, PO6, PO9	
4.	Gained Competency in Latex and other documentation tools and prepared various group reports and individual reports	PO4, PO3	
5.	Detailed study on different traffic sign detection methods and the scope of nueral networks to improve performance of the model.	PO7	PSO1
6.	To work together as a team to overcome the challenges faced and to convey our ideas and progress to the evaluation panel.	PO3, PO8, PO9,PO10	
7.	To use the knowledge acquired during the four years of study to the problem.	PO12	
8.	To manage the entire expertise and time plan of the project.	PO10	PSO1
9.	To design, build and debug the required system.	PO3,PO6 PO8, PO1	PSO3
10.	To prepare presentation, reports and paper with minimum plagiarism and providing necessary acknowledgements.	P02,P09	PSO2