

TRAFFIC SIGNAL CLASSIFICATION FOR DRIVER LESS VEHICLES

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

PRANATI C (37110587)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafied work of **PRANATI (37110587)** who carried out the project entitled "**TRAFFIC SIGNAL CLASSIFICATION FOR DRIVER LESS VEHICLES**"

under

my supervision from **August 2020 to March 2021.**

Internal Guide

Mr. R. VIGNESH, M.E.,(Ph.D.,)

Head of the Department

Dr.S.VIGESHWARI,M.E.,Ph.D.,

Dr.L.LAKSHMANAN,M.E.,Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **PRANATI. C (37110587)** here by declare that the Project Report entitled **“TRAFFIC SIGNAL CLASSIFICATION FOR DRIVER LESS VEHICLES”** is done by me under the guidance of **Mrs. R. VIGNESH, M.E.,(Ph.D.,)** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D.,** and **Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. R. Vignesh, M.E.,(Ph.D.,)** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Recognizing Traffic Signs using intelligent systems can drastically reduce the number of accidents happening worldwide. With the arrival of Self-driving cars it has become a staple challenge to solve the automatic recognition of Traffic and Hand-held signs in the major streets. Various machine learning techniques like Random Forest, SVM as well as deep learning models has been proposed for classifying traffic signs. Though they reach state-of-the-art performance on a particular data set, but fall short of tackling multiple Traffic Sign Recognition benchmarks. In this paper, we propose a novel and one-for-all architecture that aces multiple benchmarks with better overall score than the state-of-the-art architectures. Our model is made of residual convolutional blocks with hierarchical dilated skip connections joined in steps. With this we score 99.33% Accuracy in sign recognition benchmark.

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	v
1	INTRODUCTION	1
	INTRODUCTION	1
	AIM AND OBJECTIVES	2
	SCOPE	3
	ADVANTAGES	3
	REDUCTION IN ACCIDENTS	4
	THROUGH EDUCATING SYSTEMS	5
	1.5.1 PROPERTIES THAT WE HAVE TO FOLLOW WHILE CLASSIFYING	
2	LITERATURE SURVEY	6
	TRAFFIC SIGNAL	7
	RECOGNITION USING SVM	8
	DEEP LEARNING	
	FOR LARGE SCALE TRAFFIC-SIGN	
	DETECTION AND	8
	RECOGNITION	
	TRAFFIC SIGN DETECTION UNDER CHALLENGING	
	CONDITIONS	9
		10
3	SYSTEM DESIGN & METHODOLOGY	
	EXISTING SYSTEM	12
	SYSTEM CAPABILITIES	
	SYSTEM OVERVIEW	14

	REQUIREMENTS SPECIFICATION 3.5	16
	SYSTEM ARCHITECTURE	
	MODULES AND IMPLEMENTATION	19
	SETUP THE ENVIRONMENT	
	EXPLORE THE DATASET	19
	BUILD A CNN MODEL	20
	TRAIN AND VALIDATE THE MODEL	21
	TEST THE MODEL WITH THE	23
	TEST DATASET	23
4	ALGORITHM AND MODELS USED	24
	4.1.1 NAIVE BAYES ALGORITHM	25
	4.2.2 DECISION TREE ALGORITHM	27
	4.2 CONVOLUTIONAL NEURAL NETWORK	
	MODEL	28
	RESULTS AND DISCUSSION	29
5	REQUIREMENT ANALYSIS	29
	Functional	29
	Requirements Analysis	
	NON-FUNCTIONAL	
	REQUIREMENTS ANALYSIS	30
	USE-CASE ANALYSIS	
	SEQUENCE ANALYSIS	
	ACTIVITY ANALYSIS	
	COLLABORATION ANALYSIS	31
		32
	OUTPUT IMAGE	
	PRECISION GRAPH	33
		34

6	CONCLUSIONANDFUTUREWORK	36
	REFERENCES	38
	APPENDIX	
	A. PLAGIARISM REPORT	40
	B. JOURNAL PAPER	41
	C. SOURCE CODE	46

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO.
3.1	FLOWCHART OF PROPOSED SYSTEM	14
3.2	OUTER OVERVIEW OF THE SYSTEM	16
3.3	ARCHITECTURE DESIGN	20
3.4	SAMPLE SET UP OF MODULES	20
3.5	EXPLORE DATA	20
3.6	TRAIN THE DATA EXAMPLE	21
3.7	VALIDATING THE MODEL	22
3.8	TESTING THE DATA	23
3.9		
4.1	EXAMPLE FOR SIGN RECOGNIZATION	25
4.2	CNN ARCHITECTURE	26
5.1	USE-CASE DIAGRAM	31
5.2	COMMAND PROMPT	33
OUTPUT IMAGES		
	34	
	ACCURACY AND LOSS GRAPHS	35

CHAPTER 1

INTRODUCTION

INTRODUCTION

Proper management of traffic-sign inventory is an important task in ensuring safety and efficiency of the traffic flow . Most often this task is performed manually. Traffic signs are captured using a vehicle-mounted camera and manual localization and recognition is performed off-line by a human operator to check for consistency with the existing database. However, such manual work can be extremely time consuming when applied to thousands of kilometers of roads. Automating this task would significantly reduce the amount of manual work and improve safety through quicker detection of damaged or missing traffic signs. A crucial step towards the automation of this task is replacing manual localization and recognition of traffic signs with an automatic detection. In the computer-vision community the problem of traffic-sign recognition has already received a considerable attention and excellent detection and recognition algorithms have already been proposed. But these solutions have been designed only for a small number of categories, mostly for traffic signs associated with advanced driver-assistance systems (ADAS) and autonomous vehicles.

Detection and recognition of a large number of traffic-sign categories remains an open question. Various previous benchmarks have addressed the traffic-sign recognition and detection task. However, several of them focused only on traffic-sign recognition (TSR) and ignored the much more complex problem of traffic-sign detection (TSD) where finding accurate location of traffic sign is needed. Other benchmarks that do address TSD mostly cover only a subset of traffic-sign categories, most often ones important for ADAS and autonomous vehicles applications. Most categories appearing in such benchmarks have a distinct appearance with low inter-category variance and can be detected using handcrafted detectors and classifiers. Such examples include round mandatory signs or triangular prohibitory signs. However, arXiv:1904.00649v1 [cs.CV] 1 Apr 2019 2 many other traffic-sign classes that are not included in the existing benchmarks can be much more difficult to detect as they have a high-degree of variation in appearance. Instances of these categories may have a different real-world size, aspect ratio, color, and may contain various text and symbols (e.g., arrows) that significantly differ between instances of the

same class. This often leads to a large degree of intra-category (i.e. within-category) appearance variation and at the same time leads to a low degree of inter-category (i.e. between categories) variations due to similar appearance of objects from different categories

AIM AND OBJECTIVES

The main objective of our project is to design and construct a computer based system which can automatically detect the road signs so as to provide assistance to the user or the machine so that they can take appropriate actions. The proposed approach consists of building a model using convolutional neural networks by extracting traffic signs from an image using color information. We have used convolutional neural networks (CNN) to classify the traffic signs and we used color based segmentation to extract/crop signs from images.

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

The objectives are thus:

- ❖ To understand the properties of road and traffic signs and their implications for image processing for the recognition task.
- ❖ To understand colour, colour spaces and colour space conversion.
- ❖ To develop robust colour segmentation CNN algorithm that can be used in a wide range .
- ❖ To develop a recogniser that is invariant to in-plane transformations such as translation, rotation, and scaling based on invariant shape measures.
- ❖ To identify the most appropriate approach for feature extraction from road signs.
- ❖ To develop an appropriate road sign classification algorithm.

SCOPE

Traffic sign detection and recognition plays an important role in expert systems, such as traffic assistance driving systems and automatic driving systems. It instantly assists drivers or automatic driving systems in detecting and recognizing traffic signs effectively.

ADVANTAGES

- ❖ Nowadays it is difficult to find new models that do not include some sort of driver assistance system. Manufacturers are emphasizing the development of safety equipment with an aim to reducing the number of accidents caused by driver distraction and to reduce the seriousness of such accidents.
- ❖ The commonplace driver assistance systems we have now are precursors of what will eventually become fully automated driving by the year 2025. However, we won't have to wait that long to be able to drive with a special auto-pilot mode for freeways (limited-access highways), a system that is expected by the year 2020.
- ❖ And this is the context where the road sign detection system comes into play, thanks to the use of video cameras inside the vehicles. Signs such as those prohibiting certain maneuvers, speed limit signs and passing indicators are captured by these video cameras and shown on the car's dashboard.
- ❖ This system appeared for the first time in a BMW model in the year 2008. At first it detected only speed limit signs, but it has developed a great deal in terms of precision, quality, night recognition, and the number of different signs identified; there are even models that show these signs on the windshield at the driver's eye level using the head-up display system.
- ❖ The purpose of this technology is to help improve the driver's safety on the road, especially when he or she is tired or has failed to notice signs and signals. At one time or another, hasn't every driver wondered what the speed limit was? This happens because signage can vary from one section of highway to the next, creating confusion and even distracting the driver.
- ❖ Good signs on streets and highways is vital for safety, since in addition to regulating traffic, it informs the driver of the condition of the roads. This is why it is such a great

step for road safety to have a system that is capable of detecting and recognizing signs automatically and in real time on the basis of images captured from a moving vehicle.

- ❖ The current challenge is to **combine this sign detection system with other** driver assistance systems, in order to offer greater comfort and safety to drivers, all with the same final objective: to achieve fully automated driving and zero accidents on our streets and roads.

REDUCTION IN ACCIDENTS THROUGH EDUCATING SYSTEMS

Road and traffic signs, traffic lights and other traffic devices are used to regulate, warn, guide or inform road users. They help achieve an acceptable level of road traffic quality and increase safety with orderly and predictable movement of all traffic, both vehicular and pedestrians . Road and traffic signs are designed to be easily recognised by drivers mainly because their shapes and colours are readily distinguishable from their surroundings . The Swedish Road Administration is in charge of defining the appearance of all signs and road markings in Sweden. Traffic signs in Sweden are fully regulated by this administration. They are placed two meters from the road and the base-sign is at a height of 1.6 meters for roads used by vehicles with motors. According to the Road Administration, the maximum number of signs on a single pole is three with the most important sign at the bottom. In accordance with European signs, all signs are designed to have a reflective layer added on selective parts of the sign. Most Swedish road signs use pictograms to indicate the message of the sign. However, there are some exceptions in which text replaces pictograms. The STOP sign is one example of this kind of sign. All signs use Swedish text except the STOP sign where the English “STOP” word replaces the Swedish “STOPP” word. The usual background colour on warning and prohibition signs on most European signs is white, whereas this colour is yellow in Sweden. The reason is to enhance the visibility of the signs during winter time. White signs would be very hard to see in snowfall conditions. A thicker rim is used for warning and prohibition signs in Sweden compared with their European counterparts.

PROPERTICES THAT WE HAVE TO FOLLOW WHILECLASSIFING:

Road and traffic signs are characterised by a number of features which make them recognisable with respect to the environment:

- ❖ Road signs are designed, manufactured and installed according to strict regulations.
- ❖ They are designed in fixed 2-D shapes such as triangles, circles, octagons, or rectangles .
- ❖ The colours of the signs are chosen to contrast with the surroundings, which make them easily recognisable by drivers .
- ❖ The colours are regulated by the sign category .
- ❖ The information on the sign has one colour and the rest of the sign has another colour.
- ❖ The tint of the paint which covers the sign should correspond to a specific wavelength in the visible spectrum .
- ❖ The signs are located in well-defined locations with respect to the road, so that the driver can, more or less, anticipate the location of these signs.
- ❖ They may contain a pictogram, a string of characters or both .
- ❖ In every country the road signs are characterised by using fixed text fonts, and character heights.
- ❖ They can appear in different conditions, including partly occluded, distorted, damaged and clustered in a group of more than one sign .

CHAPTER 2

LITERATURE SURVEY

SURVEY1

PUBLICATION: Traffic signal recognition using Support Vector Machine (SVM) classifier

Author: Hasan Fleyeh

- ❖ This thesis presents a system to recognise and classify road and traffic signs for the purpose of developing an inventory of them which could assist the highway engineers' tasks of updating and maintaining them. It uses images taken by a camera from a moving vehicle. The system is based on three major stages: colour segmentation, recognition, and classification.
- ❖ Four colour segmentation algorithms are developed and tested. They are a shadow and highlight invariant, a dynamic threshold, a modification of de la Escalera's algorithm and a Fuzzy colour segmentation algorithm. All algorithms are tested using hundreds of images and the shadow-highlight invariant algorithm is eventually chosen as the best performer. This is because it is immune to shadows and highlights. It is also robust as it was tested in different lighting conditions, weather conditions, and times of the day. Approximately 97% successful segmentation rate was achieved using this algorithm.
- ❖ Recognition of traffic signs is carried out using a fuzzy shape recogniser. Based on four shape measures - the rectangularity, triangularity, ellipticity, and octagonality, fuzzy rules were developed to determine the shape of the sign. Among these shape measures octagonality has been introduced in this research. The final decision of the recogniser is based on the combination of both the colour and shape of the sign. The recogniser was tested in a variety of testing conditions giving an overall performance of approximately 88%.
- ❖ Classification was undertaken using a Support Vector Machine (SVM) classifier. The classification is carried out in two stages: rim's shape classification followed by the classification of interior of the sign. The classifier was trained and tested using binary images in addition to five different types of moments which are Geometric moments,

Zernike moments, Legendre moments, Orthogonal Fourier-Mellin Moments, and Binary Haar features. The performance of the SVM was tested using different features, kernels, SVM types, SVM parameters, and moment's orders. The average classification rate achieved is about 97%. Binary images show the best testing results followed by Legendre moments. Linear kernel gives the best testing results followed by RBF. C-SVM shows very good performance, but v-SVM gives better results in some case.

SURVEY2

PUBLICATION: Deep Learning for Large-Scale Traffic-Sign Detection and Recognition

Author: Domen Tabernik and Danijel Skocaj

- ❖ Automatic detection and recognition of traffic signs plays a crucial role in management of the traffic-sign inventory. It provides accurate and timely way to manage traffic-sign inventory with a minimal human effort. In the computer vision community the recognition and detection of traffic signs is a well-researched problem. A vast majority of existing approaches perform well on traffic signs needed for advanced drivers assistance and autonomous systems. However, this represents a relatively small number of all traffic signs (around 50 categories out of several hundred) and performance on the remaining set of traffic signs, which are required to eliminate the manual labor in traffic-sign inventory management, remains an open question.
- ❖ In this paper, we address the issue of detecting and recognizing a large number of traffic-sign categories suitable for automating traffic-sign inventory management. We adopt a convolutional neural network (CNN) approach, the Mask R-CNN, to address the full pipeline of detection and recognition with automatic end-to-end learning. We propose several improvements that are evaluated on the detection of traffic signs and result in an improved overall performance.
- ❖ This approach is applied to detection of 200 traffic-sign categories represented in our novel dataset. Results are reported on highly challenging trafficsign categories that have not yet been considered in previous works. We provide comprehensive analysis of the deep learning method for the detection of traffic signs with large intra-category appearance variation and show below 3% error rates with the proposed approach, which is sufficient for deployment in practical applications of traffic-sign inventory management.

SURVEY3

PULICATION: Traffic Sign Detection under Challenging Conditions: A DeeperLook Into Performance Variations and Spectral Characteristics

Author: Dogancan Temel, Min-Hung Chen, and Ghassan AlRegib

- ❖ Traffic signs are critical for maintaining the safety and efficiency of our roads. Therefore, we need to carefully assess the capabilities and limitations of automated traffic sign detection systems. Existing traffic sign datasets are limited in terms of type and severity of challenging conditions. Metadata corresponding to these conditions are unavailable and it is not possible to investigate the effect of a single factor because of simultaneous changes in numerous conditions. To overcome the shortcomings in existing datasets, we introduced the CURE-TSDReal dataset, which is based on simulated challenging conditions that correspond to adversaries that can occur in real-world environments and systems.
- ❖ We test the performance of two benchmark algorithms and show that severe conditions can result in an average performance degradation of 29% in precision and 68% in recall. We investigate the effect of challenging conditions through spectral analysis and show that challenging conditions can lead to distinct magnitude spectrum characteristics. Moreover, we show that mean magnitude spectrum of changes in video sequences under challenging conditions can be an indicator of detection performance. CURE-TSD-Real dataset.

SURVEY4

PULICATIO: Traffic-Sign Detection and Classification Under Challenging Conditions: A Deep Neural Network Based Approach

Author: Uday Kamal, Sowmitra Das, Abid Abrar and Md. Kamrul Hasan

- ❖ Traffic sign detection and classification is of paramount importance for the future of autonomous vehicle technology. Many promising methods have been proposed in the literature to perform this task. However, almost all of them work well only with clear and noise-free images, and, do not provide satisfactory results in the presence of

challenging image-capture conditions (such as noise, variation of illumination, motion artifacts, etc.)

- ❖ In this paper, we propose a Convolutional Neural Network (CNN)-based approach for detecting and classifying traffic signs which is robust against such challenging environmental conditions. In the proposed model, our approach is to consecutively (i) detect the type of challenge present in the image, (ii) selectively preprocess the image to enhance the sign-features (iii) localize traffic sign proposals from the preprocessed image using a network specifically trained for the detected challenge-type, and, finally, (iv) classify the sign proposals extracted from the image.
- ❖ These tasks are carried out by implementing CNNs of varying architecture and depth. The challenge-detector and classifier are implemented by using a VGG-16-type architecture, whereas the localizer is similar to the U-Net architecture. The proposed model has been evaluated on the Traffic-Sign Recognition Dataset provided for the IEEE VIP Cup 2017. Experiments demonstrate that, it is robust enough to attain satisfactory detection and classification accuracy even in the presence of the given challenges, with an overall precision and recall of 62% and 42% respectively.

CHAPTER 3

SYSTEM DESIGN & METHODOLOGY

EXISTING SYSTEM

In general, traffic sign recognition mainly includes two stages: the first stage is traffic sign detection, which concerns the location and size of the traffic signs in the traffic scene images, and the second stage of the process is traffic sign recognition, which pays close attention to the classification of what exact class the traffic signs belong to. Traffic sign detection is usually based on the shape and color attributes of traffic signs, and traffic sign recognition is often used with classifiers, such as convolutional neural networks (CNNs) and SVM with discriminative features.

Traffic signs have a strict color scheme, which includes red, blue, and white that allow us to distinguish traffic signs from the background scene. It is not difficult for human beings to distinguish traffic signs from a background, so for a computer detection system, color information is also an important feature. In the present research, color-based traffic sign detection method is shown to be the most straightforward and simplest method . As red, green, blue image (RGB) is too sensitive to illumination, color space conversion algorithms have often been applied to traffic sign detection. For example, Shadeed et al. segmented the image by the U and V chromaticity channels in YUV space, where U is positive and V is the color red. This information was combined with the hue channel of the hue, saturation, value (HSV) color space to segment the red road signs from the image. In Reference, thresholds were used in the color channel of the HSV color space, to segment the red road signs. Unfortunately, due to strong light, poor light, and other bad weather conditions, color-based detection methods often fail to achieve better results.

Except for countries like Japan and a few others, traffic signs have pairs of different signs when converted to grayscale which appear the same, and traffic signs from most countries have a strict shape and symbol distinction. This design allows color blind and color weak drivers and pedestrians to recognize traffic signs without or with less color information. Based on such features, some algorithms completely ignore color information and adopt shape-based image segmentation. In addition to the above-mentioned factors, some algorithms use image features other than the raw color space and shape information. The

authors in Reference used the color probability model, and traffic sign proposals were then extracted by finding the most stable extremal regions on these maps. Reference used rotation invariant binary pattern-based features. In any case, finding good feature expression is an important part of traffic sign detection, and is also an important research topic in this field.

Recently, as deep learning methods have demonstrated prominent representation capacity and achieved outstanding performance in traffic sign recognition, more scholars have applied technologies to this area. Jin used a convolution neural network based on deep learning with a hinge loss stochastic gradient descent method, which achieved a high recognition rate. Dan provided a multi-column deep neural network for traffic classification running on a graphic processing unit (GPU), and obtained a better-than-human recognition rate. Qian used CNN as the classifier to learn the discriminative feature of max pooling positions to classify traffic signs and obtained a comparable performance with the-state-of-the-art method. Ali used a procedure based on color segmentation, histogram of oriented gradients, and CNN, for the detection and recognition of traffic signs, which achieved better classification accuracy and computational speed. However, in the field of traffic sign recognition by deep learning methods, to meet the demands of real time application, it is necessary to further study the choice of discriminative features and to research the network structure to improve classification accuracy and processing time.

For a long time, there were no public and challenging datasets available in this area, but this situation changed in 2011. Larsson and Felsberg and Stallkamp et al. introduced challenging databases, including annotations for traffic sign detection classification. These databases included the Belgium Traffic Sign Classification (BTSC), the German traffic sign detection benchmark (GTSDb), and the German Traffic Sign Recognition Benchmark (GTSRB) databases. In particular, the GTSDb and GTSRB have attracted more scholars to find new methods to verify through this database, and some of them have achieved good results. The existence of open databases has made the research methods comparable. The above traffic signs databases belong to European traffic laws and regulations. Some Chinese scholars have paid more attention to the problem of traffic sign detection and recognition in the Chinese traffic environment. The construction of a traffic sign database for Chinese traffic environment is very necessary. Fortunately, the Chinese Traffic Sign Dataset (CTSD) was constructed by Yang [10], which has attracted the attention of more Chinese researchers

SYSTEM CAPABILITIES

The use of traffic sign recognition system is very challenging. Since rain, fog, snow etc. affect the whole system. Another thing is the light variation i.e., shadows, sun, clouds etc. The geometrical shape of the object and the perspective is also a big concern. So, we must be able to come up with a system which can work under light variation and geometrical transformation of the objects in a scene. If we narrow down the scope, we are particularly interested in detection of these signs under low light condition. In short, our objective is to make the system very accurate and efficient.

- ❖ In the detection stage, color information is exploited to detect regions of interest (ROI) that may correspond to traffic signs.
- ❖ The shape of these regions is tested in the classification stage, allowing rejecting many of the initial candidates and grouping traffic signs into classes.
- ❖ The pictogram contained on each ROI (if exist) is extracted, analyzed and compared with the pictogram database. The best match between the ROI and database pictogram, if high enough, is considered the sign that is more likely to appear in that ROI. Each recognized sign is part of the output result of the recognition stage.

Road Sign Recognition is a field which is concerned with the detection and recognition of road and traffic signs in traffic scenes acquired by a camera. It is a technique which uses computer vision and artificial intelligence to extract the road signs from outdoor images taken in uncontrolled lighting conditions where these signs may be occluded by other objects, and may suffer from different problems such as colour fading, disorientation, and variations in shape and size.

The first paper on the subject was published in Japan in 1984 [14]. The aim was to try various computer vision methods for the detection of road signs in outdoor scenes. Since that time many research groups and companies have shown interest, conducted research in the field, and generating an enormous amount of work. Different techniques have been used to cover different application areas (see next Section), and vast improvements have been achieved during the last decade.

The identification of the road signs is achieved through two main stages:

- ❖ Detection
- ❖ Classification
- ❖ Recognition

In the detection phase, the image is pre-processed, enhanced, and segmented according to the sign properties such as colour or shape or both. The output is a segmented image containing potential regions which could be recognised as possible road signs. The efficiency and speed of the detection are important factors because they reduce the search space and indicate only potential regions.

In the classification phase, the images from the training database were invoked either directly or by calculating the following features; Geometric moments, Zernike moments, Legendre moments, Orthogonal Fourier-Mellin moments and Binary Haar Features. Ten pairs of training/test datasets were created either from the binary images or from these features. Each pair of datasets was selected randomly without repetition from the database for binary images or from feature files for features. There were 35 training instances and 15 test instances for every category of traffic signs or Speed-Limit signs. These ten pairs of datasets were invoked, separately, for training and testing of the SVM model which were carried out in this experiment and those which follow.

In the recognition stage, each of the candidates is tested against a certain set of features (a pattern) to decide whether it is in the group of road signs or not, and then according to these features they are classified into different groups. These features are chosen so as to emphasize the differences among the classes. The shape of the sign plays a central role in this stage and the signs are classified into different classes such as triangles, circles, octagons. Pictogram analysis allows a further stage of classification. By analysing pictogram shapes together with the text available in the interior of the sign, it is easy to decide the individual class of the sign under consideration. A prototype of road sign detection and recognition system is shown in Figure 3.1. The system can be implemented by either colour information, shape information, or both. Combining colour and shape may

give better results if the two features are available, but many studies have shown that detection recognition and classification can be achieved even if one component, colour or shape, is missing.



Fig:3.1 Flowchart of proposed system

SYSTEM OVERVIEW

A system to detect and recognise road and traffic signs should be able to work in two modes; the training mode in which a database can be built by collecting a set of traffic signs for training and validation, and a prediction mode in which the system can recognise a traffic sign which has not been seen before. A system to recognise road and traffic signs is depicted in Figure 5.1. It consists of a number of modules which work together to perform this recognition. These modules are as follows: 80 Figure 3.2: A block diagram of the RSRS. 81.

Camera

A good digital camera which gives clear and sharp still images with different sizes is necessary. No special equipment is needed for this purpose. Images collected by this camera are used in later stages to develop and validate the colour segmentation algorithm, the recognition stage, and to build the classification system.

Colour Segmentation

Colour segmentation is an important step to eliminate all background objects and unimportant information in the image. It generates a binary image containing the road signs and any other objects similar to the colour of the road sign. This step reduces the amount of calculation needed in the following steps as it radically reduces the number of probable objects. A colour segmentation algorithm should be robust enough to work in a wide spectrum of environmental conditions and be able to generate binary images even when traffic sign colours are attenuated.

Shape Analysis

The main task of this module includes cleaning the binary image from noise and small objects, applying connected components labelling algorithm, and recognising the traffic sign. This module normalises the recognised traffic sign so that it becomes invariant to the in-plane transformations. This means that the resultant sign has a fixed size and it is located in a standard position where its centre of gravity is located in the centre of the image. This module works in two modes; the training and the prediction mode. In the training mode it is invoked to create or update the training image database. In the prediction mode it prepares every object in the binary image to be in standard format and ready for feature extraction.

Raw Image Database

This database is simply a collection of traffic scenes gathered by the camera. Images in this database are categorised according to the type of the sign, condition of the sign such as occluded, damaged, or faded, weather conditions, light geometry, and defects in the images such as blurring. The database represents the main source of images from which another database called the “training database” is built. The number of images in this database should be extensive enough to cover the large 82 variety of traffic signs which are used for training the system. In addition to this, the number of images containing the same type of sign should be vast to provide a good number of examples for training. An Access database program manages the images and the information about them so that selecting images with certain characteristics is straightforward.

Training Database

The training database consists of binary images of a normalised size such as 36x36 pixels. The database is created and updated in the training mode in such a way that binary images of the desired traffic signs are selected from a set of images. This database is used either directly or by extracting some features to train and validate the classifier. In the prediction mode, the database is invoked to train the classifier before any classification task takes place.

Feature Extraction

This module contains algorithms which are used to extract features from either the training images in the training database or images directly from the shape analysis unit. It allows the classifier to be trained by either binary images or by features. Among the features which can be used are geometric moments, Zernike moments, Legendre moments, Orthogonal Fourier-Mellin Moments and Binary Haar features.

Classification:

Classification is carried out using a Support Vector Machine (SVM) and using convolution neutral network(CNN) model.

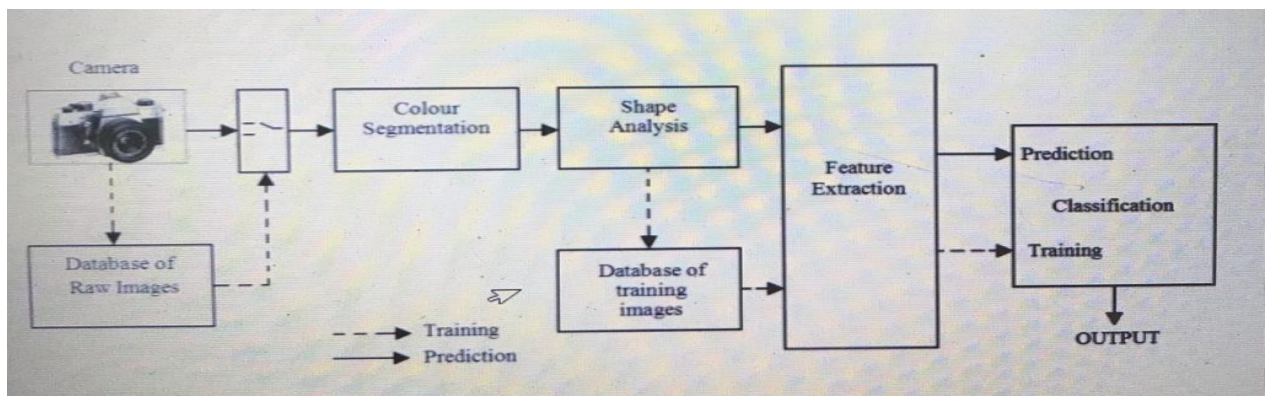


Fig: 3.2 OUTER OVERVIEW OF THE SYSTEM

REQUIREMENT SPECIFICATION

This proposed software runs effectively on a computing system that has the minimum requirements. Undertaking all the equipment necessities are not satisfied but rather exist in their systems administration since we require permissions for installing some modules. So the main need is to introduce appropriate equipment for the application. The requirements are split into two categories, namely:

Software Requirements

The basic software requirements to run the program are:

- i. Microsoft Windows XP, Windows 7, Windows 10
- ii. Database - Dataset for images
- ii. Python IDE version 3.7
- iv. Latest versions of modules like tensorflow, numpy etc.

Hardware Requirements

The basic hardware required to run the program are:

- i. Hard disk of 20 GB
- ii. System memory (RAM) of 512 MB
- iii. I3 processor-based computer or higher.

SYSTEM ARCHITECTURE

The main working principle of the project is to classify the traffic signal the below is the architecture diagram how the process is being happening and the different stages of it at first we need to Extract the training data and then we have to test and split the training data and these data have to under go into the Train models based algorithms such as efficient Bayes naive algorithm and also by using Decision Tree model . Then the data has to be validate.After efficiently validated we need to build a CNN model from that the result be picked.

The processing flow of our traffic sign detection and recognition system is illustrated in figure 3.2. The red line with the arrow indicates the offline training flow of the subclass

classified network, and the blue line with the arrow indicates the whole process of online detection and classification. In detail, during the feature extraction, we filtered the road scene grayscale images by four orientations, and two scale simplified Gabor filters, where its parameters were optimized by experiments. The simplified Gabor filters were used to enhance the edges and smooth the non-edge areas of the images. Then, the eight output

feature maps were synthesized into one map by maximizing the value at each pixel. In the synthesized Gabor feature map, the edges of the traffic signs were strengthened and the areas inside the traffic signs were smoothed. The feature extraction process is shown in the green area of this figure. During the detection stage, we first used the maximally stable extremal regions (MSERs) algorithm to obtain the regions of interest (ROI) and filtered out the areas where there was little possibility for existence of traffic signs, according to our defined filter rules. Then, the proposed regions were classified by SVM to get the super-class of the traffic signs which each area belonged to. The detection process is shown in the yellow area of the figure. Finally, we classified the sub classes of the traffic signs using CNN, which is shown in the blue area of the figure. The input of the CNN was eight simplified Gabor feature maps, which were convoluted to traffic signs by the SGW. What needs to be stressed is that in the real time processing stage, the input of the CNNs was the matrix that was cut from the eight simplified Gabor feature maps in the detection stage. The matrix corresponded with the area that was detected as a traffic sign at the same stage. In the red line flow part of figure 3.2, we trained the CNN with eight simplified Gabor feature maps, convoluted from the traffic signs of GTSRB.

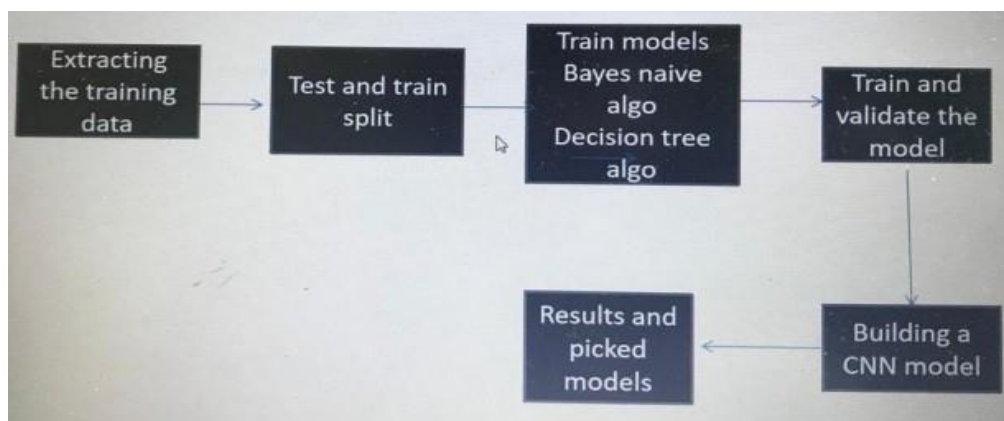


Fig: 3.3 ARCHITECTURE DESIGN

MODULES AND IMPLEMENTATION

This project requires prior knowledge of Keras, Matplotlib, Scikit-learn, Pandas, PIL and image classification.

To install the necessary packages used for this Python data science project, enter the below command in your terminal

- ❖ Pip install tensorflow keras sklearn matplotlib pandas pil

For this project, we are using the public dataset available at Kaggle. The data set 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 data set has a train folder which contains images inside each class and a test folder which we will use for testing our model.

The implementation is involved into various stages the below are the required stages:

- ❖ Set up the environment
- ❖ Explore the dataset
- ❖ Build a CNN model
- ❖ Train and validate the model
- ❖ Test the model with test dataset

SET UP THE ENVIRONMENT

This describes the environment that is where the exact file is located or in which format the files have to be set up. These files or modules in the folder are kind of like meta data that is which again consist of data in it as the meta module describes about the dataset which consist of images and that meta has to be trained by different algorithms and models and after detail training of specific image which we want to explore that be trained and test of the trained be tested after testing the result be given in test phase the data is classified across different data set and the suitable result which matches the in reserved dataset be taken as output.

is PC > Local Disk (D:) > dataflair projects > Project - Traffic sign classification >

Name	Date modified	Type	Size
meta	04-Oct-19 12:52 PM	File folder	
test	04-Oct-19 12:52 PM	File folder	
train	04-Oct-19 12:55 PM	File folder	
Meta	25-Nov-18 6:13 PM	Microsoft Excel C...	2 KB
Test	25-Nov-18 6:13 PM	Microsoft Excel C...	418 KB
Train	25-Nov-18 6:13 PM	Microsoft Excel C...	1,896 KB

Fig : 3.4 SAMPLE SETUP OF MODULES

EXPLORE THE DATASET

Our „train“ folder contains 43 folders each representing a different class. The range of the folder is from 0 to 42. With the help of the OS module, we iterate over all the classes and append images and their respective labels in the data and labels list. The PIL library is used to open image content into an array. Finally, we have stored all the images and their labels into lists (data and labels). We need to convert the list into numpy arrays for feeding to the model. The shape of data is (39209, 30, 30, 3) which means that there are 39,209 images of size 30×30 pixels and the last 3 means the data contains colored images (RGB value). With the sklearn package, we use the train_test_split() method to split training and testing data. From the keras.utils package, we use to_categorical method to convert the labels present in y_train and t_test into one-hot encoding.



Fig: 3.5 EXPLORE DATASET

BUILD A CNN MODEL

To classify the images into their respective categories, we will build a CNN model (Convolutional neural network). CNN is best for image classification purposes. The name “convolutional neural network” indicates that the network employs a mathematical operation called convolution. Convolutional networks are a specialized type of neural networks that use convolution in place of general matrix multiplication in at least one of their layers. In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied. A convolutional layer within a neural network should have the following attributes

- ❖ Convolutional kernels defined by a width and height (hyper-parameters).
- ❖ The number of input channels and output channels (hyper-parameter).
- ❖ The depth of the Convolution filter (the input channels) must be equal to the number channels (depth) of the input feature map.

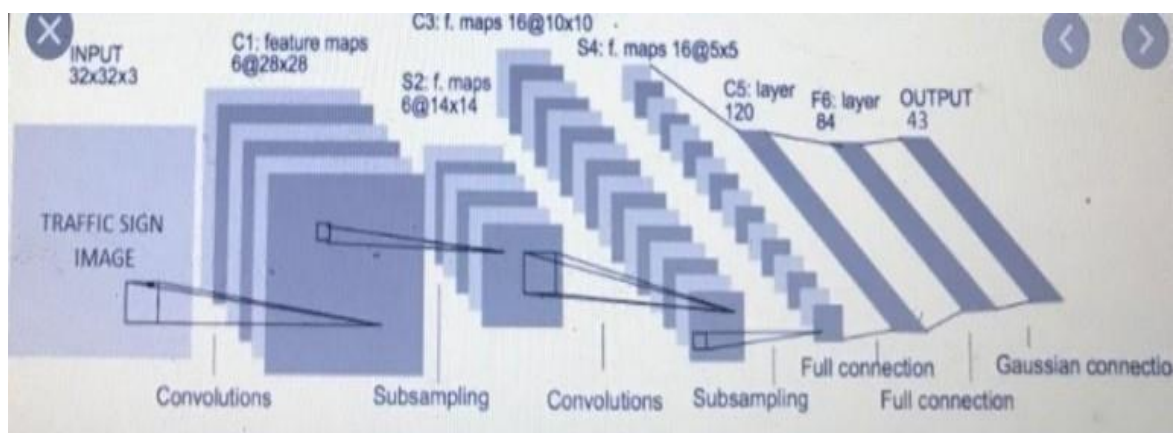


Fig 3.6 CNN ARCHITECTURE

The architecture of our CNN model is:

- ❖ 2 Conv2D layer (filter=32, kernel_size=(5,5), activation="relu")
- ❖ MaxPool2D layer (pool_size=(2,2))
- ❖ Dropout layer (rate=0.25)
- ❖ 2 Conv2D layer (filter=64, kernel_size=(3,3), activation="relu")

- ❖ MaxPool2D layer (pool_size=(2,2))
- ❖ Dropout layer (rate=0.25)
- ❖ Flattenlayer to squeeze the layers into 1 dimension
- ❖ Dense Fully connected layer (256 nodes, activation="relu")
- ❖ Dropout layer (rate=0.5)
- ❖ Dense layer (43 nodes, activation="softmax")

We compile the model with Adam optimizer which performs well and loss is "categorical_crossentropy" because we have multiple classes to categorise.

TRAIN AND VALIDATE THE MODEL

After building the model architecture, we then train the model using model.fit(). I tried with batch size 32 and 64. Our model performed better with 64 batch size. And after 15 epochs the accuracy was stable.

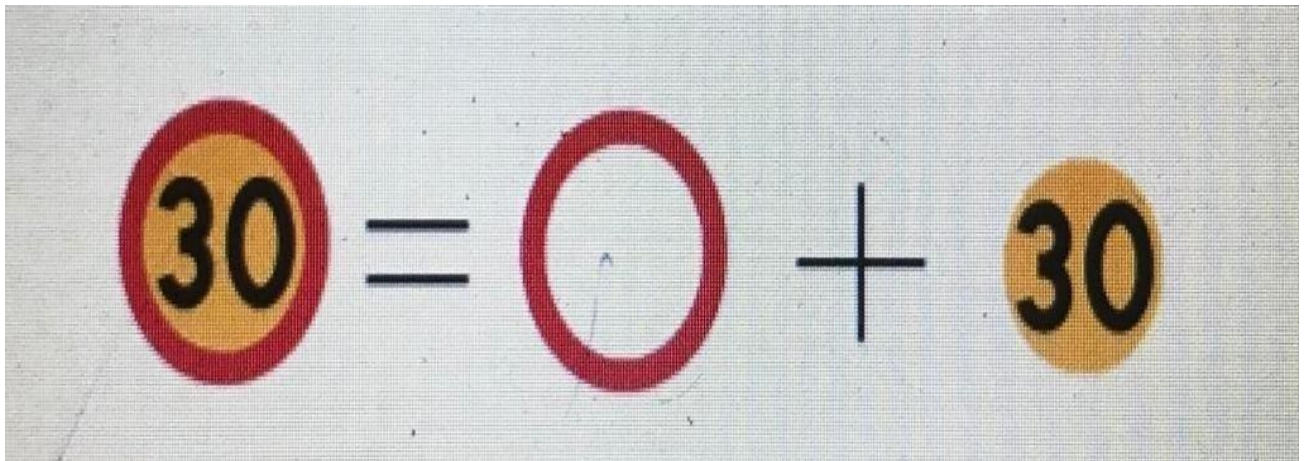


Fig 3.7 TRAIN THE DATA EXAMPLE

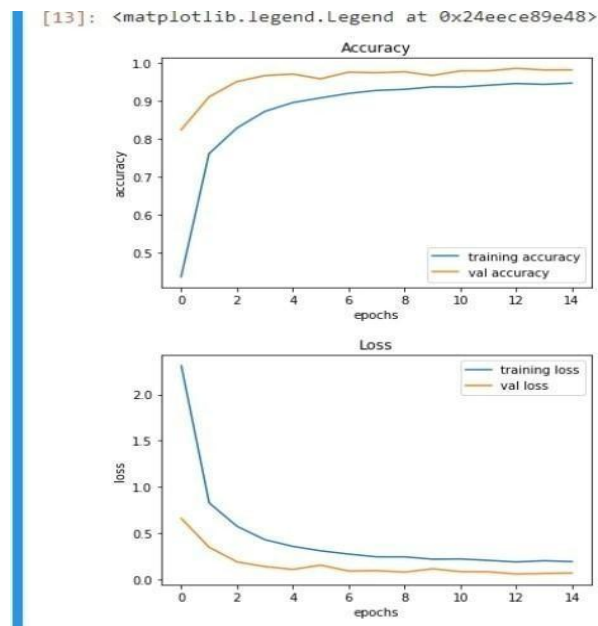


Fig: 3.8 VALIDATING THE MODEL

TEST THE MODEL WITH THE TESTDATASET

Our dataset contains a test folder and in a test.csv file, we have the details related to the image path and their respective class labels. We extract the image path and labels using pandas. Then to predict the model, we have to resize our images to 30×30 pixels and make a numpy array containing all image data. From the sklearn.metrics, we imported the accuracy_score and observed how our model predicted the actual labels. We achieved a 95% accuracy in this model. In the end, we are going to save the model that we have trained using the Keras model.save() function.

```
❖ model.save(„traffic_classifier.h5“)
```

The below diagram describes the accuracy about the training the data and testing the data. The blue line in the figure 3.9 describes the training significants of the data and the red line in the figure 3.9 describes the testing significants of the data. By comparing both testing configurations and training configurations we can tell the exact accuracy of the current traffic sign. The figure 3.9 shows the classification accuracy in the form of percentage.

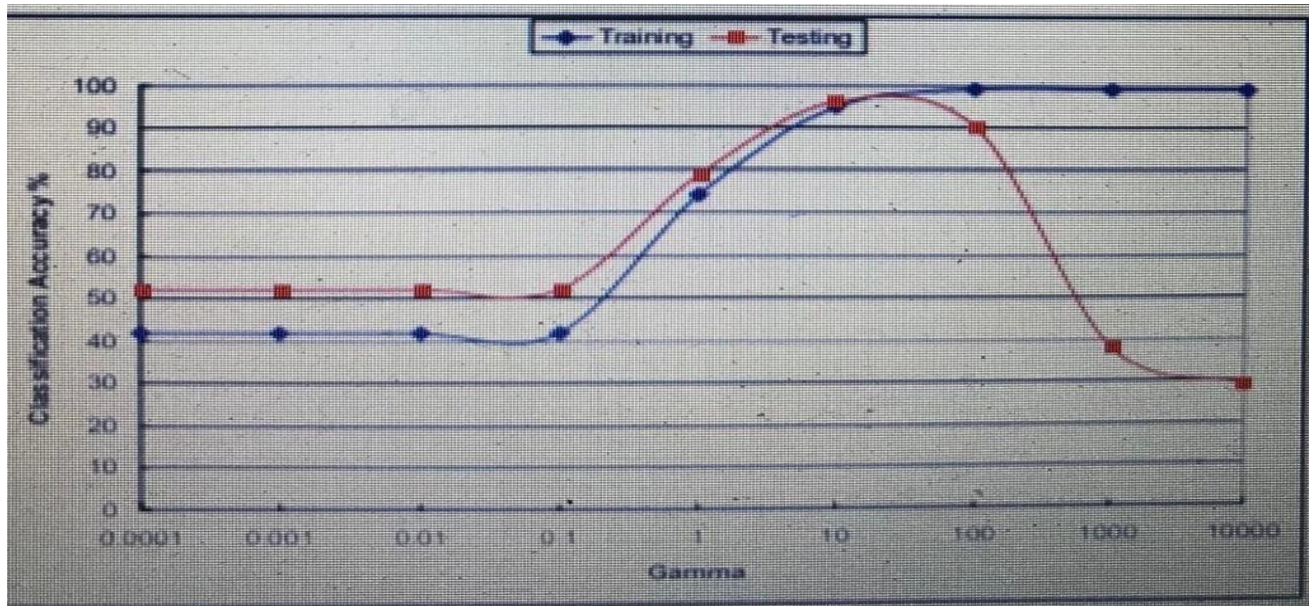


Fig 3.9 TESTING THE DATA

Traffic Signs Classifier GUI

Now we are going to build a graphical user interface for our traffic signs classifier with Tkinter. Tkinter is a GUI toolkit in the standard python library. Make a new file in the project folder and copy the below code. Save it as `gui.py` and you can run the code by typing `python gui.py` in the command line. In this file, we have first loaded the trained model „traffic_classifier.h5“ using Keras. And then we build the GUI for uploading the image 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 we have to provide the same dimension we have used when building the model. Then we predict the class, the `model.predict_classes(image)` returns us a number between (0-42) which represents the class it belongs to. We use the dictionary to get the information about the class. Here's the code for the `gui.py` file.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

ALGORITHM AND MODELS USED

The algorithm starts by converting the RGB image into the HSV colour space. A shadow-highlight invariant colour segmentation algorithm is invoked to segment this image according to the desired colour. This colour segmentation algorithm is applied twice; for the red colour which corresponds to the rim of the sign and for the yellow which represents the interior of the sign.

The algorithm checks every red segmented image for any objects. If any objects are found, it calculates the number of these objects by applying the connected components labelling algorithm. The colour segmentation algorithm is designed to reject noise and all small objects which cannot be recognised by using the region growing algorithm, and hence no median filter is needed. Using the red segmented image, a list of objects is then generated and objects are sent sequentially to the next stage. At this stage, the convex hull of the object under consideration is computed using Graham's algorithm. Using the convex hull of the red segmented image, the corresponding area of the yellow segmented image is checked for the presence of pixels. If yellow pixels are found with a reasonable ratio, this area is considered as a probable sign and sent to the next stage, otherwise the object is rejected and the next object in the list is treated in exactly the same manner. Once an object with these specifications is found, the fuzzy shape recogniser starts to put the object into one of the four categories described earlier. Once the object fits in any category, a rectangle is drawn around it; otherwise the object is removed from the list of objects. Figure 3.9 shows a number of images recognised using this algorithm. It is tested under bad light conditions, blurred images, faded signs, under highlight condition, occlusion, rain and snow, and it shows very good robustness. Further details of these tests are presented in the next chapter.



Fig : 4.1 EXAMPLE FOR SIGN RECOGNIZATION

CONVOLUTIONAL NERUAL NETWORK MODEL

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections,

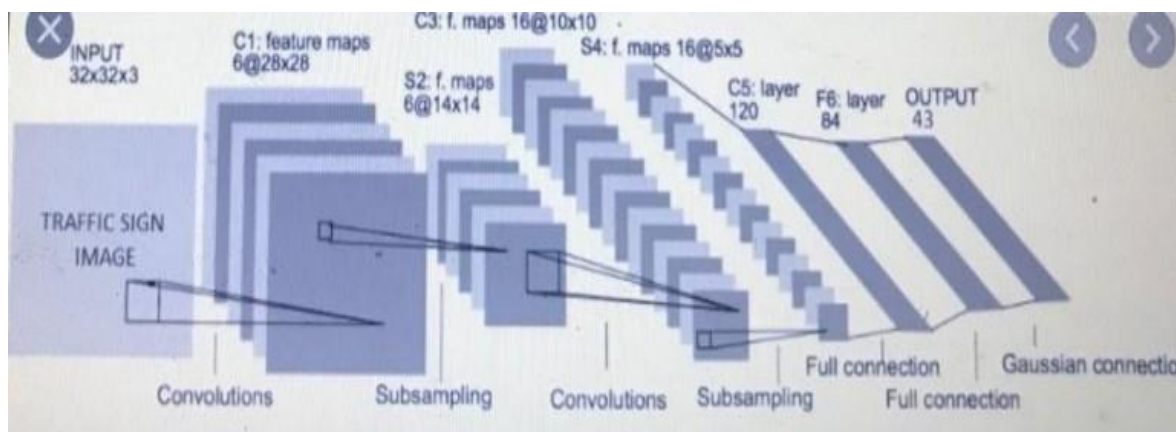
recognition faces etc., are some of the areas where CNNs are widely used. CNN image classifications takes an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). Eg., An image of $6 \times 6 \times 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 \times 4 \times 1$ array of matrix of grayscale image. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernal), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values

BUILD A CNN MODEL

To classify the images in to their Respective categories ,we will build a CNN model (Convolutional neural network). CNN is best for image classification purposes. The name “convolutional neural network” indicates that the network employs a

networks that use convolution in place of general matrix multiplication in at least one of their layers. In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to image recognition tasks. A convolutional layer within a neural network should have the following attributes

- ❖ Convolutional kernels defined by a width and height (hyper-parameters).
- ❖ The number of input channels and output channels (hyper-parameter).
- ❖ The depth of the Convolution filter (the input channels) must be equal to the number of channels (depth) of the input feature map.



The architecture of our CNN model is:

- ❖ `2Conv2DLayer(filter=32,kernel_size=(5,5),activation="relu")`
- ❖ `MaxPool2DLayer(pool_size=(2,2))`
- ❖ `Drop out layer(rate=0.25)`
- ❖ `2Conv2DLayer(filter=64,kernel_size=(3,3),activation="relu")`
- ❖ `MaxPool2DLayer(pool_size=(2,2))`
- ❖ `Drop out layer(rate=0.25)`
- ❖ `Flatten layer to squeeze the layers into 1 dimension`
- ❖ `Dense Fully connected layer(256 nodes, activation="relu")`
- ❖ `Dropout layer(rate=0.5)`
- ❖ `Dense layer(43 nodes, activation="softmax")`

We compile the model with Adam optimizer which performs well and loss is "categorical_crossentropy" because we have multiple classes to categorise.

❖

NAIVE BAYES ALGORITHM

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as „Naive“. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

DECISION TREE ALGORITHM

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split. There are many algorithms out there which construct Decision Trees, but one of the best is called as ID3 Algorithm.

ID3 Stands for Iterative Dichotomiser.

4.3 1 Assumptions while creating Decision Tree

Below are some of the assumptions we make while using Decision tree:

- In the beginning, the whole training set is considered as the **root**.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

CHAPTER 5

RESULTS AND DISCUSSION

REQUIREMENT ANALYSIS

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

FUNCTIONAL REQUIREMENTS ANALYSIS

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

Flexibility

The flexibility of the project is provided in such a way that it has the ability to run on different environments being executed by different users.

Safety

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

NON- FUNCTIONAL REQUIREMENTS ANALYSIS

Portability

It is the usability of the same software in different environments. The project can be run in any operating system.

Performance

These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

Accuracy

The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

Maintainability

Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how

easy it is to maintain the system, analyse, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

USE-CASE ANALYSIS

Use case diagram overview the usage requirement for system they are useful for presentations to management and/or project stakeholders, but for actual development you will find that use cases provide significantly more value because they describe the meant of the actual requirements. A use case describes a sequence of action that provide something of measurable value to an action and is drawn as a horizontal ellipse. A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

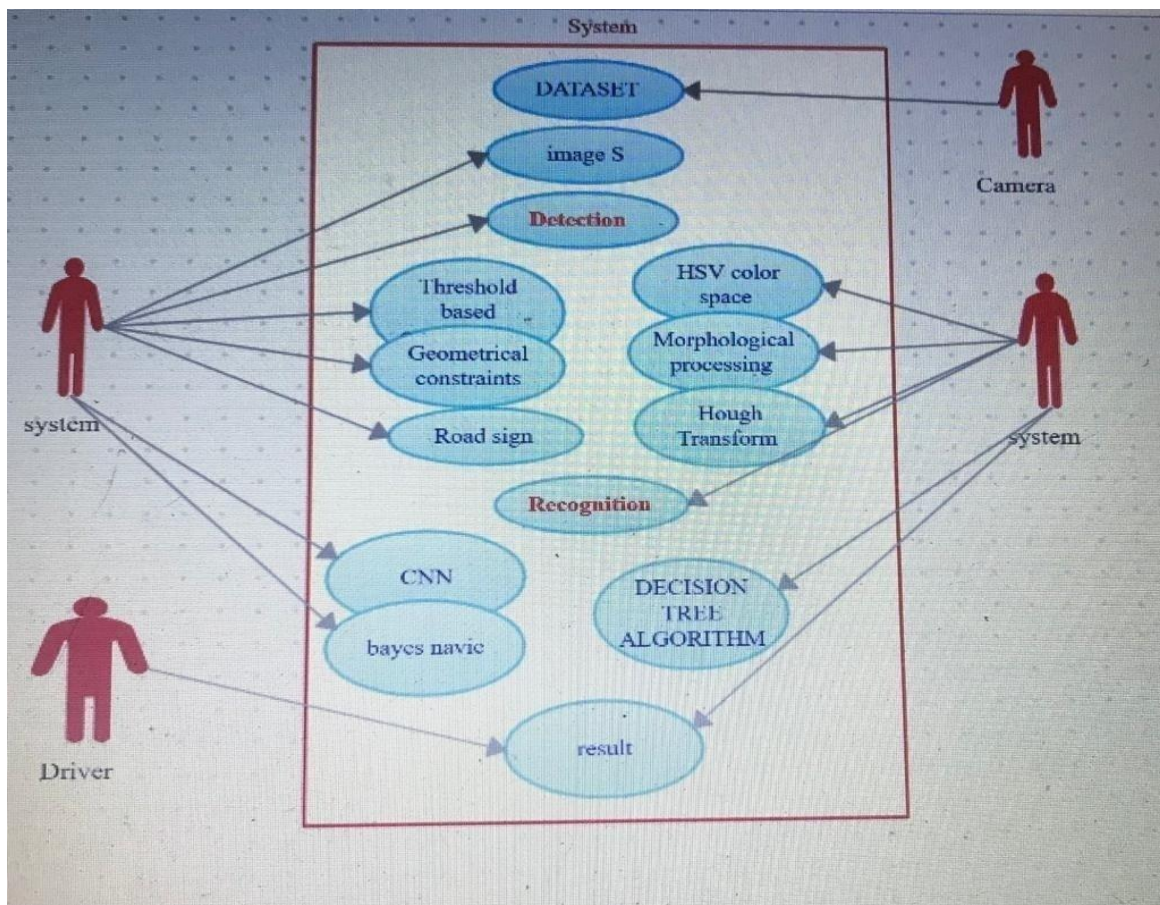


fig5.1use-case diagram

SEQUENCE ANALYSIS

Sequence diagram model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and commonly used for both analysis and design purpose. Sequence diagram are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

ACTIVITY ANALYSIS

Activity diagram is a graphical representation of work flows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

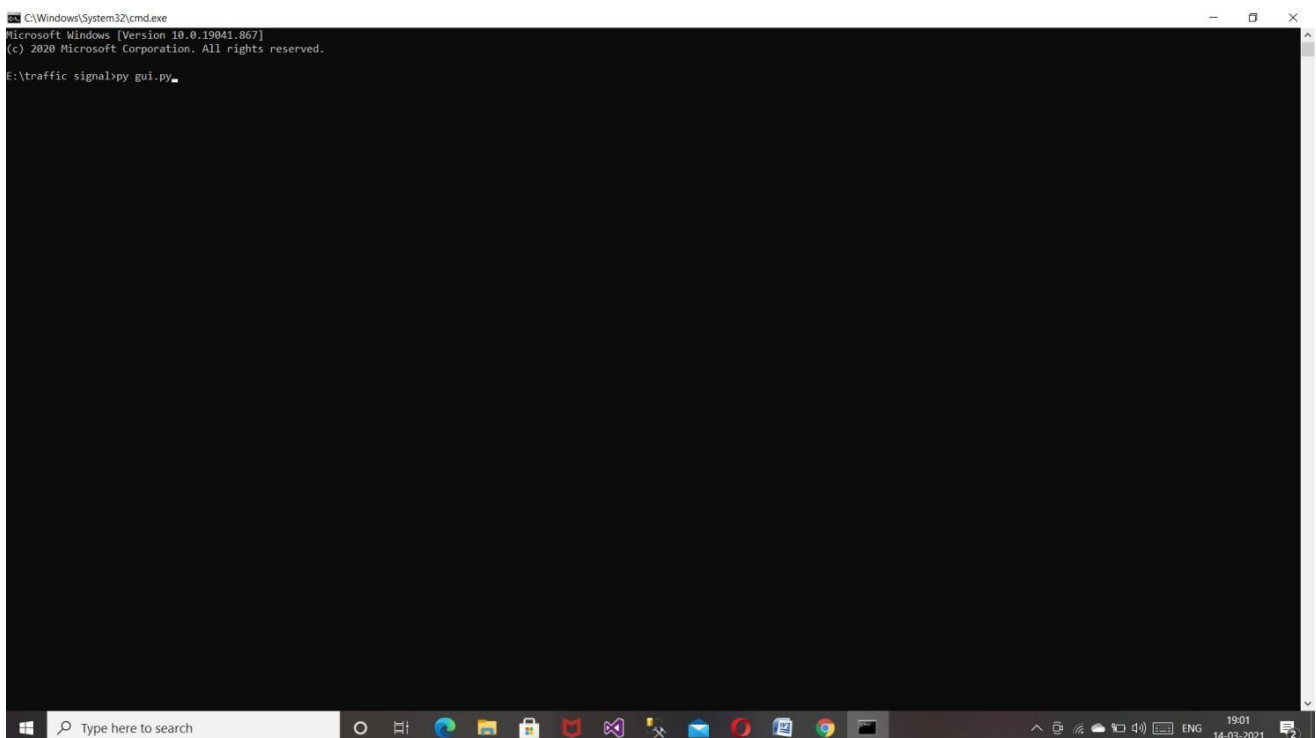
- ❖ Rounded rectangles represent activities.
- ❖ Diamonds represent decisions.
- ❖ Bars represent the start or end of concurrent activities.
- ❖ A black circle represents the start of the work flow.
- ❖ An encircled circle represents the end of the work flow.

COLLABORATION ANALYSIS

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and object.

RUN THE CODE:

To run the code first we need to open the command prompt and then we should set the path to the drive consisting the code file. Then we should give the command as `python filename.py`. Here we saved the file as app so we gave the command as `python gui.py` in the cmd.



5.2 Command prompt

OUTPUT IMAGES:

When we use command prompt for the output the output screen will be appear then we should select an image from the data set which we already have . Then after uploading the image we have to classify the image then the output will be appeared displaying the meaning of the traffic signal.

OUTPUT 1:



OUTPUT 2:



OUTPUT 3:



5.9 LOSS AND ACCURACY GRAPH:

Our dataset contains a test folder and in a test.csv file, we have the details related to the image path and their respective class labels. We extract the image path and labels using pandas. Then to predict the model, we have to resize our images to 30×30 pixels and make a numpy array containing all image data. From the sklearn.metrics, we imported the accuracy_score and observed how our model predicted the actual labels. We achieved a 95% accuracy in this model.

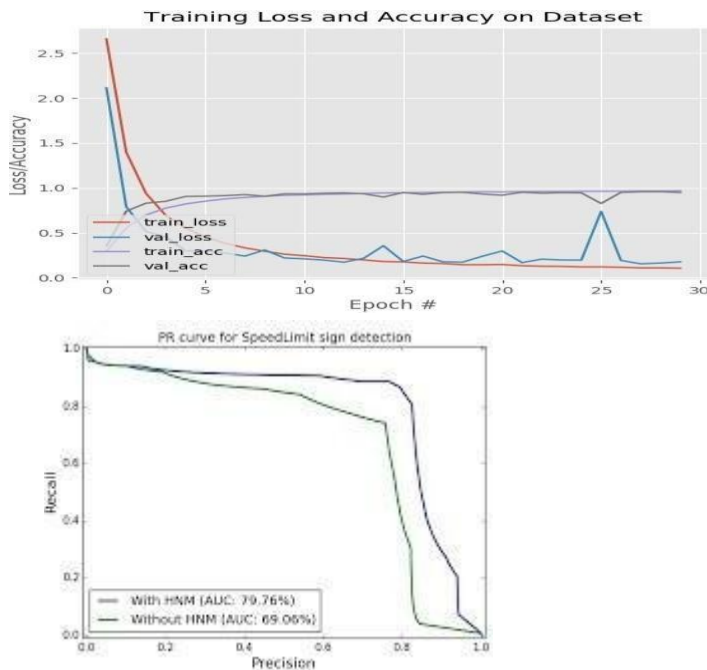


fig 5.3 accuracy and loss graphs

CHAPTER 6

CONCLUSION AND FUTUREWORK

CONCLUSION

We covered how deep learning can be used to classify traffic signs with high accuracy, employing a variety of pre-processing and regularization techniques (e.g. dropout), and trying different model architectures. We built highly configurable code and developed a flexible way of evaluating multiple architectures. Our model reached close to close to 98% accuracy on the test set, achieving 99% on the validation set. Personally, I thoroughly enjoyed this project and gained practical experience using Tensorflow, matplotlib and investigating artificial neural network architectures. Moreover, I delved into some seminal papers in this field, which reinforced my understanding and more importantly refined my intuition about deep learning. From the following results we can see that the CNN is doing a good job in classifying different types of traffic signs when the extracted signs are cropped perfectly from the image. Our approach fails to give good results when the extracted signs from test images are cropped incorrectly. Another drawback of our approach is that when the color of the traffic signs vary which may be due to bad weather conditions and poor camera quality, the image masks obtained are not perfect and hence the signs are not detected properly.

FUTURE WORK

In the future, the recognition phase can be sped up using dimension reduction of feature vectors. The number and parameters of Gabor filters will greatly influence the results of the classification results. It is necessary to research the optimization of Gabor parameters using optimization algorithms, to improve the efficiency and accuracy of traffic sign detection and classification. Finally, we accelerated our method by a GPU and further improved the efficiency. Traffic sign detection is the most important stage in the whole process. It decides whether the sign can be classified or not. In this research a fuzzy shape recogniser is used to detect the sign. However, there are other methods which can be used at this stage. One is image similarity or Hausdorff distance which measures the similarity between two images. These methods can be used as part of the future work. Another factor, the aspect ratio of the object under consideration, can be used as an additional

measure to detect the traffic signs. This is because all traffic signs have uniform shapes and are designed to have the same width and height. A combination of image similarity.

REFERENCES

- [1] S. Vitabile, A. Gentile, and F. Sorbello, "A neural network based automatic road sign recognizer," presented at The 2017 Inter. Joint Conf. on Neural Networks, Honolulu, HI, USA, 2017.
- [2] J. Miura, T. Kanda, and Y. Shirai, "An active vision system for real-time traffic sign recognition," presented at 2011 IEEE Intelligent Transportation Systems, Dearborn, MI, USA, 2012.
- [3] A. de la Escalera, J. Armingol, and M. Mata, "Traffic sign recognition and analysis for intelligent vehicles," *Image and Vision Comput.*, vol. 21, pp. 247-258, 2014.
- [4] S. Lu, "Recognition of traffic signs using a multilayer neural network," presented at 1994 Canadian Conf. Electrical and Computer Engineering, Halifax, NS, Canada, 2015.
- [5] P. Paclik and J. Novovicova, "Road sign classification without color information," presented at Sixth Annual Conf. of the Advanced School for Computing and Imaging, Lommel, Belgium, 2017.
- [6] Avinash N. Ho Gi Jung Hyuckmin Na Monika Singh, Dev Dutt Yadav. Traffic sign recognition-based vehicle speed regulation. 2018.
- [7] Andrzej Ruta, Yongmin Li, and Xiaohui Liu. Real-time traffic sign recognition from video by class-specific discriminative features. *Pattern Recognition*, 43(1):416-430, 2010.
- [8] Soendoro W.D., Supriana I. Traffic sign recognition with Color-based Method, shape-arc estimation and SVM; Proceedings of the 2011 International Conference on Electrical Engineering and Informatics; Bandung, Indonesia. 17-19 July 2011.
- [9] Ardianto S., Chen C., Hang H. Real-time traffic sign recognition using color segmentation and SVM; Proceedings of the 2017 International Conference on Systems, Signals and Image Processing (IWSSIP); Poznan, Poland. 22-24 May 2017.

- [10]Shadeed W.G., Abu-Al-Nadi D.I., Mismar M.J. Road traffic sign detection in color images; Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems; Sharjah, UAE. 14-17 December 2013

A. PLAGIARISM REPORT



Plagiarism Checker X Originality Report

Similarity Found: 9%

Date: Sunday, March 14, 2021

Statistics: 186 words Plagiarized / 1959 Total words

Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

TRAFFIC SIGNAL RECOGNITION AND CLASSIFICATION FOR
AUTONOMOUS VEHICLES Pranati C1, Pachipulusu Niharika2,Vigesh R3
1B.E. Computer science and Engineering, Sathyabama Institute of Science
and Technology, Chennai, India 2B.E. Computer science and Engineering,
Sathyabama Institute of Science and Technology, Chennai, India 3Assistant
Professor, Department of Computer Science and Engineering, SIST,
Chennai, India ABSTRACT---As increasing the technology and the
modernness in the world using the autonomous vehicles became normal
and technologies involved in that has also been modernized and it also
helps to move towards efficiency .

B. JOURNAL PAPER

TRAFFIC SIGNAL RECOGNITION AND CLASSIFICATION FOR AUTONOMOUS VEHICLES

Pranati C¹, Pachipulusu Niharika², Vigesh R³

¹B.E. Computer science and Engineering, Sathyabama Institute of Science and Technology, Chennai, India

²B.E. Computer science and Engineering, Sathyabama Institute of Science and Technology, Chennai, India

³Assistant Professor, Department of Computer Science and Engineering, SIST, Chennai, India

ABSTRACT---As increasing the technology and the modernness in the world using the autonomous vehicles became normal and technologies involved in that has also been modernized and it also helps to move towards efficiency. The traffic signal classification and recognition is crucial for driver less vehicles which helps to classify and to identify the signals. Traffic signal classification is kind of software which gives ability to the system to classify or to identify in which class the traffic signal exactly belongs to. The techniques which have been recognized earlier are lacking in the accuracy to achieve accuracy in this methodology the vehicles has be able to interpret traffic signs and it have to make according decisions by keeping all the points we have designed a model that is efficiently gives the exact accuracy in classifying and in recognizing the signal by using CNN(convolution neural network) model and also by detail explore of data set and also by using Decision Tree algorithm included with Naive Bayes Architecture.

INTRODUCTION

In the world of Artificial Intelligence and advancement in the technologies, many researchers and big companies like Tesla, Uber, Google, Toyota, Ford, Audi, etc are working on autonomous vehicles and self-driving vehicles. As there are thousands of traffic signal in the world so the self driving vehicles has to recognize the image or the data set more accurately to make it's own decisions and technique which has involved in prediction also have to be more efficient. So in this case to help and to predict the accuracy we have to deal with the traffic signal classification and also they are many various of technology also have been involved in these classification of traffic based images.

As the self driving vehicles which have to make decisions on their own and the people who fully depends on it can save time if they are going for long drives etc. But to obtain level 8 in technology related to autonomous necessary it is mandatory for vehicles to automatize themselves and to achieve success in following the traffic rules successfully and without consuming more time. The main objective of this work is to prevent all this fatality and disasters that happen in the society and the drivers should get some kind of assistance that guide them or suggest them to automatize the vehicles. It's never a feasible or efficient idea to expect a person to be with you when you are driving to assist you. Hence it is like a system that be made like an assistance to the driver. The main motivation of these project is to Road safety According to global road rash statistics nearly 1.4 million people die in road related accidents each year that's 3,500 people die on average, sadly driver error due to drunk and drive, reckless driving, driver distraction are the leading causes of deaths on road. Automation recognition for autonomous driving.

The methodology involved in the project is CNN model which means convolutional neural network indicates that the network employs a mathematical operation called convolution. Convolutional neural networks is a type of networks which comes under neural. Normal systems generally use matrix multiplication but convolution system uses convolution in place in one of the layers at least for one time. In deep learning the CNN model (convolutional neural network or ConvNet) is a class of deep neural networks which is most commonly applied and a convolutional layer within a neural network should have the following attributes: Convolutional kernels which is initialized by hyper parameters like height and width and length and it also focus on the number of input channels and also outputs like height and width. The depth of the input channels must be equal to the depth of number of channels which features the input map.

RELATED WORK

Automatic Traffic signal Detection Based On Colour Shape and SVM: They developed a TSDR system with enriched data set and which have invariant in viewing angle rotation and colour segmentation. This system uses a RGB for colour segmentation and also decides the correlated accuracy towards its positive and negative rate. This was proposed by Safat Wali.

Traffic Signs Classification Based On Deep Feature Learning: This technique is based on support vector Machine it classifies the present image or present sign based on the old version by unique segmentation in it and they have used the YCbCr colour space which have been introduced in the project to classify colour and it also stores the tested details in database for future references which helps in reusability this was proposed by Yan Lai and Nanxin Wang.

Overview Of Traffic Sign Classification methods : This mainly focus on detecting the efficient ways and best accuracy result from the past developed techniques. It mainly focus on learning based methods including deep learning and also focus on hand crafted methodologies it summarized the data in the form of tables and the content coincides with the data set this was published under the International Journal of Multimedia.

Real Time Traffic Sign Detection Based On Simplified Gabor Wavelets : In this the images of the road will be converted to grayscale with the help of Gabor Wavelets where the optimization happens related to the edges and it compares with data set and exact classification will happen and based on the classification the exact region be recognized and also some techniques involved to increase the performance this was proposed by Faming Shao, Xinqing.

EXISTING RECOGNITION SYSTEM

Naive Bayes Algorithm Based Running Systems:

Naive Bayes which is also known as naive bayes classifiers which belongs to the classification family. It works based on independence assumptions. The naive Bayes takes the parameters like colour or its length or height based on that it classifies the object into its corresponding set. It requires the number of parameters in line with the number of variables in a problem which helps to achieve high scalability. As the process is linearly dependent and have to take many assumption so the systems which believe in Naive Bayes may not be good in case efficiency and also it consumes times. This may also causes confusion to the system since it uses various parameters. This method is easy to build and this method is particularly used for very large data set classification.

Decision Tree Algorithm based Systems: Decision Trees which comes under the type of Supervised Machine Learning. This algorithm helps the system to understand the data and classifies the data into its corresponding data set and also to interpret. It helps to solve regression and classification problems. The systems which work based on Decision tree can be good but it depends on which algorithm we used while constructing the Decision Tree one of the best is called as ID3 Algorithm. ID3 stands for Iterative Dichotomiser 3.

PROPOSED SYSTEM

In proposed, Decision Tree Machine Learning Algorithm predicts best. It is implemented to increase operational efficiency. It reduces Query retrieval time. Accuracy is improved using Machine Learning algorithm. The proposed system begins with the thought that was not executed by the ancestors. It makes use of the Decision Tree machine learning procedure for classification and also for calculating.

RESULTS & DISCUSSIONS

The main working principle of the project is to classify the traffic signal the below is the architecture diagram how the process is being happening and the different stages of it at first we need to Extract the training data and then we have to test and split the training data and these data have to under go into the Train models based algorithms such as efficient Bayes naive algorithm and also by using Decision Tree model . Then the data has to be validate.After efficiently validated we need to build a CNN model from that the result be picked.

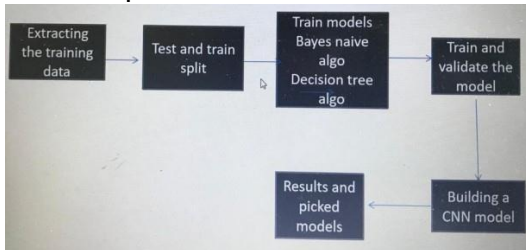


Figure 1.1

Explore the data set:

The trainfolder contains 43 folders each consist a different type of class. The range of the folder from 0 to 42.With the help of the OS module we can iterate over the classes which we want or we can iterate over all the classes and extend the images and their labels in the data and labels list.The PIL library is helps to explore image content into an array.



Figure 1.2

Building a CNN model:

To classify the images into their respective categories we have to build a CNN model (Convolutional Neural Network). CNN is best and efficient for image classification and to purposes the requirement.

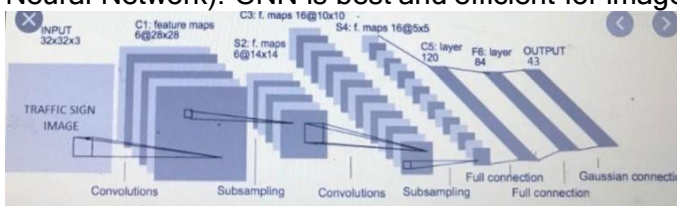


Figure 1.3

Train and validate the model:

After enabling the architecture we have to train the model by using `model.fit()`. we can train with batch size 64 and 32. Our architecture model performs good and efficient with 64 batch size. After some average epochs the accuracy will be efficient and the accuracy can be stable.

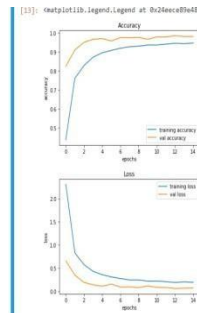


Figure 1.4

Test our model with test dataset:

The dataset which we are using contains a test folder and in a test.csv file we have the details related to the image path and their corresponding class labels. We have to extract the image path and the labels respective to the image or data using pandas. Then to identify the model, we have to resize images to 30×30 pixels and we have to make a numpy array which consist of all images data. From the metrics which is in sklearn module, we have to import the accuracy_score and we have to observe the model predicted and the exact labels. And We will be achieving 95 percentage of accuracy in this architecture.

output snapshots:



Figure 1.5



FIGURE 1.6



FIGURE 1.7

Graph for accuracy and loss:

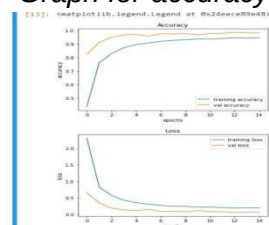


FIGURE 1.8

CONCLUSION

Thus, the aim of this paper is to bring the 95 percent accuracy in the classification and recognition of the traffic signals. This helps to find the exact signal and according to that the self driving vehicles takes action t

he efficiency in the project is achieved by using CNN model with detail explore of the data set. It also gives the accuracy towards graph towards its loss and profit.

REFERENCES

- [1] A. de la Escalera and M. Mata, Develops image and compute vision of a traffic signal recognition and also provides analysis for vehicles which are worked autonomously in 2014.
- [2] Andrzejruta. Recognized from a video by classifying class-specific discriminative and its features which is a Real-time signal recognition. Pattern Recognition in 2010.
- [3] Avinash Ho and Dev Dutt Yadav. Introduced a regulation which helps to recognize traffic signals in terms of speed in 2018.
- [4] Ardianto and Hang. Recognized traffic signs using color through its segmentation and SVM which is a real time project; Proceeded in 2017 by International Conference, Signals and Image Processing (IWSSIP); Poznan, Poland. 22-24 May 2017.
- [5] J. Miura and Shirai "Recognized active vision software for traffic sign identification which is a real time project," published in 2011 at IEEE Intelligent Transportation, Dearborn, MI, USA, 2012.
- [6] Paclik and Novovicova, Introduced road signal classification without color segmentation and its information published at Sixth Annual of advanced School for Computing Belgium in 2017.
- [7] S. Vitabile, A. Gentile, and F. Sorbello, "A neural network based automatic road sign recognizer," presented at The 2017 Inter. Joint on Neural Networks, Honolulu, HI, USA, 2017.
- [8] S. Lu Recognition and classification of traffic signs using a neural network multilayer presented in the year 1994 Canadian Configuration. Electrical and Computer Engineering, Halifax, NS, Canada, 2015.
- [9] Soendoro and Supriana. They recognized traffic sign based on color and its corresponding methods have been involved base and also by using shape arc and SVM; Bandung, Indonesia. 17-19 July 2011.
- [10] Shadeed and Abu-Al-Nadi. Detection of traffic signals based on colors in the images; Proceed by of 10th IEEE International Conference, Circuits and Systems; Sharjah, UAE. 17 December 2013.

C. SOURCE CODE

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

import numpy
#load the trained model to classify sign
from keras.models import load_model
model = load_model('traffic_classifier.h5')

#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' }

#initialise GUI

top=tk.Tk()

top.geometry('800x600')

top.title('Traffic sign classification')

top.configure(background='#CDCDCD')

label=Label(top,background='#CDCDCD', font=('arial',15,'bold'))

sign_image = Label(top)

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(sign)
label.configure(foreground='#011638', text=sign)

```

```

def show_classify_button(file_path):
    classify_b=Button(top,text="Classify Image",command=lambda:
classify(file_path),padx=10,pady=5)
    classify_b.configure(background='#364156', foreground='white',font=('arial',10,'bold'))
    classify_b.place(relx=0.79,rely=0.46)

```

```

def upload_image():
    try:
        file_path=filedialog.askopenfilename()
        uploaded=Image.open(file_path)
        uploaded.thumbnail(((top.winfo_width()/2.25),(top.winfo_height()/2.25)))
        im=ImageTk.PhotoImage(uploaded)

        sign_image.configure(image=im)
        sign_image.image=im
        label.configure(text="")
        show_classify_button(file_path)
    except:
        pass

```

```

upload=Button(top,text="Upload an image",command=upload_image,padx=10,pady=5)
upload.configure(background='#364156', foreground='white',font=('arial',10,'bold'))

```

```

upload.pack(side=BOTTOM,pady=50)
sign_image.pack(side=BOTTOM,expand=True)
label.pack(side=BOTTOM,expand=True)
heading = Label(top, text="Know Your Traffic Sign",pady=20, font=('arial',20,'bold'))
heading.configure(background='#CDCDCD',foreground='#364156')
heading.pack()
top.mainloop()

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

data = []
labels = []
classes = 43
cur_path = os.getcwd()

#Retrieving the images and their labels
for i in range(classes):
    path = os.path.join(cur_path,'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)

```

```

        #sim = Image.fromarray(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)

print(data.shape, labels.shape)
#Splitting training and testing dataset
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=42)

print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)

#Converting the labels into one hot encoding
y_train = to_categorical(y_train, 43)
y_test = to_categorical(y_test, 43)

#Building the model
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'))

```

```

#Compilation of the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

epochs = 15
history = model.fit(X_train, y_train, batch_size=32, epochs=epochs, validation_data=(X_test,
y_test))
model.save("my_model.h5")

#plotting graphs for accuracy
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()

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()

#testing accuracy on test dataset
from sklearn.metrics import accuracy_score

y_test = pd.read_csv('Test.csv')

labels = y_test["ClassId"].values

```

```
imgs = y_test["Path"].values

data=[]

for img in imgs:
    image = Image.open(img)
    image = image.resize((30,30))
    data.append(np.array(image))

X_test=np.array(data)

pred = model.predict_classes(X_test)

#Accuracy with the test data
from sklearn.metrics import accuracy_score
print(accuracy_score(labels, pred))
```