National College of Ireland

Multiclass Classification of Road Traffic Signs Using Machine Learning Algorithms

MSc Research Project
Data Analytics

Rahul Jaju
Student ID: 18125671

School of Computing
National College of Ireland

Supervisor: Dr. Cristina Muntean

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rahul Jaju		
Student ID:	X18125671		
Programme:	Data Analytics	**Year:**	2018-2019
Module:	Research Project		
Supervisor:	Dr. Cristina Muntean		
Submission Due Date:	12/12/2019		
Project Title:	Multiclass Classification of Road Traffic Signs Using Machine Learning Algorithms		
Word Count:	**9224**	**Page Count** 25	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Rahul Jaju

Date: 12/12/2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	X
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	X
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	X

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

National College of Ireland

Multiclass Classification of Road Traffic Signs Using Machine Learning Algorithms

Rahul Jaju

X18125671

Abstract

Modern world's vehicles are fully simplified and intelligent by means of automatic driving and the detection of road traffic signs is very much critical. While driving, the driver sometimes accidently fails to pay attention of the road traffic signs which are present on the streets. This can be dangerous for the driving person and also for the individuals with him. With an effectual Smart Driving Support System (SDSS) the driver can be informed about the signs. Detection and recognition are the two stages of this smart system based on its features and meaning. The phase of detection is achieved by Canny Edge Detector. The recognition stage is achieved by implementing Convolutional Neural Network (CNN) due to its impressive feature and performance. Further, we also implement other classifiers such as Support Vector Machine, K-Nearest Neighbor, Random Forest and Extreme Gradient Boosting. These models are implemented to check if their performance is better than CNN. The highest accuracy of 97.3 percent was obtained for Convolutional Neural Network and the lowest of 49.08 percent for SVM (Support Vector Machine). The standard GTSRB data set with multi classes is used to show the efficiency of the proposed approaches.

Keywords: Smart Driving Support System, CNN, K-Nearest Neighbor, Random Forest, SVM, Extreme Gradient Boosting.

1. Introduction

Nowadays, we can see a steady ascent in the automobiles on street. This rise is additionally accountable for the enormous number of accidents happening everywhere. The deadly accidents which happen are fundamentally because of the driver's inability to comprehend the visual information which is available as traffic signs. The (Smart Driving Support System) SDSS is developed to avoid these fatal accidents and furthermore guarantees the drivers safety. The significant applications of this system are to identify the blind spots, recognizing the traffic signs, assistance for emergency brakes, etc.

The recognition of traffic signs is a challenge of multi class classification which is implemented in two phases; detection and further interpretation of the detected traffic signs. The signs provide us the necessary information about laws and rules to pursue while driving. The signs are divided into three main categories and 43 sub-classes. Informatory sign gives essential data to drivers about the directions. Cautionary sign cautions the driver from any sort of hazardous circumstance ahead on roads. Multiple times because of substantial traffic or awful climate conditions driver may miss his attention which may lead to road accidents. Another general issue isn't having the alternative to fathom the significance of the sign. Regardless of whether drivers figure out how to detect the sign, quite possibly there is no clue for its interpretation.

This system bolsters the driver and gives incredible amount of info and aides him to drive securely. This system centers significantly on the road traffic and other speed limit signs. The interpretation of road traffic signs addresses a huge issue while analyzing this field. The paper inspects the present traffic sign revelation and recognition methodologies. Various occasions, drivers face numerous troubles on obscure roads and in such circumstances concentrating on road and the traffic signs can end up being a challenging task.

Although, different methodologies are been implemented to carry out previous research in this field. Regrettably, there is no such framework as SDSS system which could successfully accomplish the best accuracy in recognizing the signs. The present system can perceive and detect hardly some classes of sign images. Distinctive climate conditions like shady and stormy can bring about incorrect identification of the signs. Additionally, the present frameworks can't recognize the signs in the night or under the shadows of the trees. That is the place where the challenge in this research lies to precisely perceive these traffic sign under such conditions with more classes. Smart Driving Support System (SDSS) plays an important role in road sign detection and traffic safety. This challenge of successfully interpretation of numerous and diverse signs has always attracted researchers interest and considerable research is done in the field of computer vision network (Shustanov & Yakimov 2017). This project expects to acquire the information from various traffic signs which is critical in various applications that incorporates safety, self-driving automobiles, sign inventory, mapping and navigation.

How can the multiple classes of road traffic signs can be recognized by different modelling techniques (CNN, SVM, Random Forest, XGBoost and K-Nearest Neighbor) to ensure drivers safety on roads for our Smart Driving Support System (SDDS)?

The research question intends to give most extreme security to drivers by providing them information about the traffic signs and road conditions by precisely interpreting them. The research also aims to successfully classify 43 classes of traffic signs from almost 40 thousand images with our proposed classification techniques and provide the necessary information of each classified sign to the driving person. The first major objective of this report is to effectively detect the road traffic signs and interpret it by making use of the machine learning techniques. The proposed techniques will successfully detect and classify 43 classes of traffic signs from almost 50 thousand images. Second objective of this report is to further evaluate the results of the implemented techniques and find out the best machine learning technique for our Smart Driving Support System (SDSS) based on execution run time.

The project report describes the next sections in the following way. For instance, section 2 talks about the literature review which discusses about the various researchers and their contribution in this domain. In section 3, the methodology which we have opted for our implementation is well explained. Section 4 and section 5 is dedicated to the design specification and implementation respectively. The evaluation and results of our analysis is covered in section 6. The final section 7 concludes the performance achieved by each of the implemented model and emphasis more on the gaps and limitations.

2. Related Work

In the past, researchers did their exploration on SDSS (Smart Driving Support System) by utilizing distinctive techniques in Machine Learning. This segment gives us a concise appraisal about the work done in the field of sign detection and accurately recognizing it.

2.1 CNN Used as A Classifier

At present, there are many publications carrying their research in the field of road safety for the purpose of traffic sign detection. The accompanying technique is been used in one of the most recent publication. The publisher (Shustanov & Yakimov 2017) has used the modified model of Generalised Hough Transform to evaluate the coordinates of sign. First, segmentation and recognition on the basis of shape and colour is done. After that CNN is implemented for the purpose of sign validation as a classification method which gave the accuracy of 99.94%. Another paper (Girshick et al. 2014) proposed a model based on Deep Convnet that results in better accuracy for classification. Although, the implemented model is slow in terms of performance as every time a Convnet is progressed without sharing the computation to the proposed object. (Changzhen et al. 2017) have presented a paper totally dependent on Regional Proposed Network (RPN) with the utilization of deep Convolutional Neural Network (CNN). This model was actualized on the Chinese traffic signs. Afterward, the model was tried on sequence of 33 video of the size 640x480. The recordings were from the driving recorder camera set in the vehicle. The accuracy in recognizing the signs achieved by this model was 99%.

The model proposed by (He et al. 2015) speed up the distribution on convolutional network is based on (SPPnets) i.e. Spatial Pyramid pooling network. The proposed implementation has some drawbacks due to its multi stage pipeline. The system proposed by (Qian et al. 2016) has two modules. The first module creates an efficient scheme candidate and a simplified region. This approach is carried by RGB thresholding and Connected Component Analysis (CCA). In second module, CNN is used to classify and categorise each region.. The two-stage system proposed by (Fistrek & Loncaric 2011) for recognition and affirmation of traffic signs is dependent on Neural Network. The detection stage incorporates arranging the images in groups. The pixels from the picture are separated that compares to the sign edges. Arranging of pictures is finished by Neural Network. The distinguished signs are perceived by Hough method. The model proposed by them is appropriate for huge scope of pictures of different resolutions and clicked in lightning conditions. The main disadvantage of this framework was insufficient speed for execution in real time of this proposed model. The algorithm presented by (Changzhen et al. 2017) is totally dependent on profound Convolutional Neural Network (CNN) with the utilization of Regional Proposed Network (RPN). This technique was actualized on the Chinese traffic signs. The model was tested on a sequence of 33 video frames of the size 640x480. The recordings were from the recorder camera put in the vehicle. The accuracy in recognizing the signs achieved by this model was 99%. The two experiments demonstrated by (Algorry, García & Wofmann 2018) to check with the detection quality was fairly improved. Further, YOLO v2 was used with the architecture to identify the region of interest (ROI). For the purpose of classification, a light weight CNN is used to enhance the

quality of detection with less processing time. Only, few classes of images were used for the purpose of training. A single CNN model by (Jurišić, Filković & Kalafatić 2016) was intended to analyse the overall performance on several multiple datasets. Their model had just one layer which was associated completely over every convolution layer. In the technique, training was performed on two distinct sets. The two datasets (BTSC and rMASTIF) brought about result of 86.4% and 91.06% individually.(Wang 2018) discovered that deep convolutional neural network is much efficient to be utilised on larger datasets for training purpose. Large datasets which were utilized had numerous sorts of complex signs. For each image fed as n input, the convolution layer separated the feature from image. Furthermore, then on an alternate scale the coordinates of images are investigated. In the last test of experiment, the iterations were performed for 20,000 times at 0.01 learning rate. The calculated accuracy was 96%.

2.2 Support Vector Machine Used as A Classifier

The canny edge detector made use by (Biswas, Fleyeh & Mostakim 2014) and successfully detected the traffic sign of circular shape. Further, the signs were interpreted by an SVM classifier. The recognition accuracy was around 98%. The method proposed by (Rahmad et al. 2018) which has a useful feature extraction which is effectual that is based on combination of three stages. The initial step is HOG technique. This strategy is utilized to represent the image in a gathering of histograms. The subsequent step is trailed by the use of filter. This filter is utilized to reproduce and stimulate the human visual framework features to disconnect the direction and recurrence of image. Last step is use of LBP (Local Binary Pattern) calculation to classify image texture. Further, SVM is utilized for classification. The model proposed by (Yuan, Xiong & Wang 2017) is based on incremental framework for detection of traffic sign. In this method the video frame is given as an input and later by using an offline trained detector the signs are detected. The implementation comprises of five stages. The last stage is recognition on Scale-Based which is utilized using the multi-class SVM to perceived these identified signs.

An alternative approach of color segmentation by (Fifik & Turán 2010) is based on two-step detection. The two stages are structured as two subsystems. First subsystem quicker and more composed of classification and CSR square. Second subsystem is slower is empowered with Segmentation, feature memory, extraction and Classification square. The research done by (Jain & Gianchandani 2019) was based on MSER and OCR. The two-stage hybrid model makes use of Lucy-Richardson for the purpose of calculation. The first of stage methodology includes steps like; de-blurring and noise removal, conversion from RGB to binary, enhancement and edge detection. Second stage incorporates steps like morphological division, geometric sifting, associating the characters, content line arrangement and the last step is word partition. The novel paper projected by (Creusen, Hazelhoff & De With 2012) for sign interpretation was grounded on color transformation. Extraction of color from color channels leads to color transformation. Recognition of traffic sign was carried on dataset of 33,409 images by linear SVM, which resulted in 90% gain.

2.3 MLP (Multi-Layer Perceptron) Used as A Classifier

In the algorithm proposed by (Swathi & Suresh 2018) the traffic signs were perceived in the form of video frames. The phase of recognition was carried by MLP (Multilayer Perceptron) was trained with HOG (Histogram Oriented Gradient) features. The entire procedure was taken from a real time video which was used as an input for pre-processing. The proposed calculation was executed on Microsoft Visual Studio which brought about recognition pace of 95.71% and 95.71% for circular and triangular signs respectively. The design developed by (Höferlin & Zimmermann 2009) was structured in three stages. The stages were tracking, detection and classification. It was an advance CCD algorithm version. The two modules; SURF and SIFT were additionally added. The method was performed on a sequence of 30-minute consists of 94 sign images. The recognition of signs was carried by use of MLP (Multi-Layer Perceptron) that could successfully identify sign and gave the accuracy of 96.4%

2.4 Other Classification Techniques

The detection of signs using Viola-Jonas detector was discovered by (Houben et al. 2013) for the detection of signs. The binary classifier detector is applied to an input window which is sliding. The recognition process was carried by using AdaBoost. The implementation of this proposed methodology gave accuracy of 91.3% for circular and 90.7% for triangular signs.
(Zavadil, Tuma & Santos 2012) found a paper for image processing which was based on MATLAB. The two procedures are dependent on blob analysis and pattern recognition have. The outcomes are accompanied with description of main algorithms. The model could effectively perceive the traffic sign and images a goat about 2m with a 100% success rate. In spite of the fact that, the time taken with the end goal of image processing was 0.7 s roughly and the dataset utilized had not many images.

Later, the generic system founded by (Siegler & Kardkov 2011) was very much appropriate for mobile computer and personal environments. The highly quality video stream was fed as an input. The analysis was performed in three arrangements in various atmosphere conditions. First arrangement brought about precision of 78.4%. The other two arrangements brought about precision of 69.7% and 68.1% separately. (Torresen, Bakke & Sekanina 2005) did a research where the Norwegian Signs were used. The approach comprised of three stages. First was dependent on colour filter. Second and third were dependent on location of sign in the picture and recognizing the numbers imprinted on it. The technique was performed on 198 pictures and the outcome gave 91% as recognition rate. The unique method proposed by (Wu & Tsai 2013) for detection of traffic signs was mainly for maintenance and road signs inventory. The result of the calculation which was performed on 123 pictures gave recognition rate of 97%. The two ways for improving the image recognition rate presented by (Liu & Maruya 2009). The one way was used to predefined the image processing procedures and the second one was to control the exposure in camera. The system proposed by (Maldonado Bascón et al. 2010) is a special framework that increased an improvement by 4-5% in accuracy. The framework likewise made execution with support vectors which was decreased to nearly 60-70%.

2.5 Challenges and Limitations

The literature review in the above section describes about the various techniques proposed and implemented towards successfully recognition and interpretation of traffic signs. Some of the models were proposed for performance enhancement. Although, various researchers have proposed different models for recognizing the street traffic signs. Here are some of the constraints to comprehend the traps in the current task of SDSS (Smart Driving Support System). The limitation which (Biswas, Fleyeh & Mostakim 2014) faced was that the complete number of false positives were noteworthy than lesser than those of false negatives. Despite the fact that, the accuracy was slight better however it needed calculation for segmentation of variable digits which affected the false positive. The constraint of the framework proposed by (Algorry, García & Wofmann 2018) did not perform well considering the metrices. This was because of the less data utilized for training in image processing. (Yuan, Xiong & Wang 2017) powers strong spatial dispersion on the area of the signs. Another confinement is that the framework likewise requires the vanishing horizon should be in the centre of the clicked image, which is fundamental as a rule. Our proposed framework defeats every one of these impediments and entanglements.

3. Methodology

Data mining is vague procedure which includes numerous steps which starts from creating the data. To streamline the procedure of data mining many algorithms and techniques have been created. The three widely adopted methodologies are (KDD), (SEMMA) and (CRISP-DM). These are generally used techniques for data mining by data scientists and other experts. In our report we have proposed CRISP-DM for data mining. We propose the use of CRISP-DM for our task as it helps in better comprehension of our business objective. It additionally changes over the business objective in a data mining activity.

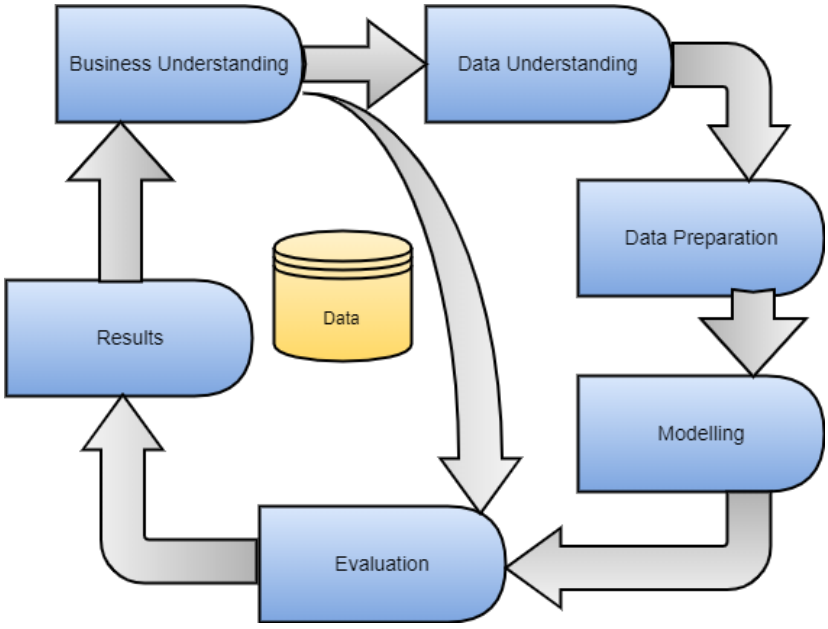


Figure 1: CRISP-DM Methodology

The methodology of CRISP-DM is entirely adaptable and furthermore improves our speculation and the strategies for further iterations in data analysis. We have altered each of the six phases of CRISP-DM procedure according to our proposed SDSS (Smart Driving Support System) which is shown in the figure (1). The principal stage incorporates detection, recognition and understanding of the street traffic sign and that is the area where our examination is centred around. Detecting and correctly classifying the traffic signs acknowledgment is at present a critical aspect in field of Smart Driving Support System implied for drivers. This stage resembles the first stage where we need to classify 43 classes of traffic signs. The following stage is to comprehend the data or information which we are going to use in project implementation. The input data is fed as images. This resembles the stage of data understanding. Data preparation is composed in third stage and in this stage, the data is prepared for modelling techniques implementations. In this stage we will isolate the important data about the sign from rest of the sign image. We will likewise resize the picture and further process the image for noise removal that fits our criteria. Finally, we get the desired dataset of processed images needed for purpose of implementing the machine learning techniques.

3.1 Business Understanding

The first and very critical stage of CRISP-DM methodology is Business Understanding. It is planned for getting hold of complete information on the domain and research as far as business point of view is concerned. It includes careful examination and comprehension of the research objective. Research objectives are set up and are obviously characterized in this stage giving an appropriate comprehension of the Business part needed for the research and analysis. Business Understanding comprises of three significant sub-parts in particular Business Goal, Business Plan and Plan Evaluation.

3.2 Data Understanding

One of another critical stage of CRISP-DM is Data understanding. To understand all the features required for the analysis correctly can bring out several evaluating results. The methodology and its implementation start with downloading and understanding the dataset. The standard GTSRB (German Traffic Sign Road Benchmark) dataset is used. Figure 2 explains about the various classes of signs and its meaning. The dataset can be downloaded from the link[1]. The dataset has over 50 thousand image files. There are around 43 classes of signs out of which we are considering 20 classes for our implementation. They are Speed limit (20km/h), (30km/h), (50km/h), (60km/h), (70km/h), (80km/h), (100km/h), (120km/h), No passing for vehicles over 3.5 metric tons, Right-of-way at the next intersection, Priority road, Yield, Stop, No vehicles, Vehicles over 3.5 metric tons prohibited, No entry, General caution, Dangerous curve to the left, Dangerous curve to the right, Double curve, Bumpy road, Slippery road, Road narrows on the right, Road work, Traffic signals, Pedestrians, Children crossing, Bicycles crossing, Beware of ice/snow, Wild animals crossing, End of all speed and passing

[1] http://benchmark.ini.rub.de/?section=gtsrb&subsection=news

Figure 2: Data Exploration and description

limits, turn right ahead, turn left ahead, ahead only, go straight or right, go straight or left, keep right, Keep left, Roundabout mandatory, End of no passing, End of no passing by vehicles over 3.5 metric tons. The downloaded images come with annotations and they are provided with CSV files and the produces the file, the coordinates of the box where sign is found, the size and the label which is assigned.

3.3 Data Preparation and Pre-processing

The downloaded images are in ppm (Portable Pixel Map) format of 250x250 and 15x15 pixels. We first need to convert them to 'PNG' from 'PPM'. In this procedure we will also crop the sign images from the coordinates which are provided in the additional CSV file in such a way that only images of signs are visible. This is achieved by calling out a function labeller.py. This program is found the directory corresponding to it. The techniques like intensity scaling, rotation in a range of angles, smoothing or blurring, shearing, brightness adjustment, colour normalisation is systematically deployed. They are grounded on sk-image and open-cv. The brightness adjustment is done to adapt the various lightning conditions in the environment where the sign is present. The sign which appears are captured from various angles from the camera installed in the car. The rotation is done to identify the signs from different angles. Also, signs will appear differently from different viewing angles with certain degree of distortion. Also, the size of sign differ is terms of scale due to the distance between the vehicle and the sign board. Some gets bigger with the vehicle moving closer to it. Therefore, scaling of images is done to get different sizes of sign. Also, sometimes due to dust and fog the sign images are not clearly visible to driver. So, we also deploy the technique of blurring and smoothing to produce the similar conditions. Brightness adjustment is deployed to produce variant lightning conditions like night, cloudy and shady.

3.4 Modelling

To build u a model dependent on machine learning and algorithms for precisely detecting and recognizing the road traffic sign from various classes after pre-processing and altering several parameters such as blurring of image, rotating , intensity adjustment, tilting, shearing, resizing and colour normalization and other factors, a vague research is been done and discussed in the

section of literature. This report is also focused on how algorithms based on Deep Learning have beaten the machine learning techniques that are conventional in nature. Hence the various methods like CNN ensemble and other machine learning is adopted have been applied to accomplish the examination objective. Satisfying a similar reason CNN, Random Forest, SVM, XGBoost and KNN modelling techniques are implemented.

Convolutional Neural Network

Convolutional neural network is just like or similar to artificial neural network where multiple layers are present. These layers are between input and the output. This neural network finds the accurate and right manipulation in mathematics to switch from input to output, whether it's some sort of non-linear or linear relationship. (Qian et al. 2016) This network or part of this system travels through the layers ascertaining the likelihood of each yield. For instance, a CNN that is prepared to perceive the breeds of dogs will go over the given picture and ascertain the likelihood that the detected image of dog is classified to certain breed. Anytime the user can audit the outcomes and select which probabilities the system should show (above a certain threshold) and return the proposed name. Each manipulation done with numbers is considered and viewed as a layer, and complex CNN have numerous layers, so as the name is termed as 'Convolutional' neural network. It interprets images in the form of pixels. For instance, the image with size 64x64 an array of 64x64x2 is generated. Where 64x64 is height and width of the image and 3 is for the channel values of RGB. The intensity of each pixel is defined by this value. (Kavitha et al. 2018)The layer of convolution is always at the first. The matrix with the image values are allowed to enter in it. The reading of the matrix form of image begins from the top of the left. Next another smaller matric is selected which is called as 'filter'. The task of the filter is to produce the convolution. It moves along the image. Secondary task of filter is to multiply the pixel values with its value. The network also includes more seven convolution layers that are mixed with pooling and nonlinear layers. When the traffic sign image is passed through the one layer, the output layer of the first layer becomes second layer input. And it is repeated for the further layers. After each convolution layer a nonlinear is added. The pooling layer is a nonlinear layer as it works with height and width of the image and its is also responsible for down sampling which reduced the image volume. It becomes compulsory to attribute a fully connected layer. A fully connected layer is a N dimensional vector where number of classes is denoted by N.

Random Forests

Random Forest, similar to its name infers, comprises of countless decision trees that work as a group. The group of decision trees is also called as ensemble. Every individual tree brings out a prediction in class and the most voted class turns out to be prediction for the model. The concept of bagging and boot strapping is vividly used. Random Forests is unexcelled in precision with the current algorithms. (Man 2018) It runs productively with more efficiency on large datasets. It can deal with a large number of input variables without variable erasure. It gives appraisals of what factors are significant in the classification. It is very effective method for assessing missing data and keeps up the accuracy when any missing values are found in data. It can very well balance the error considering the unbalanced data and class population.

Support Vector Machine

The discriminative classifier which is theoretically characterized by an isolating hyperplane. In another way, for a labelled data from a training set, the algorithm makes use of the optimal hyperplane as for its output which also categorizes the new examples. For a two-dimensional space the plane is divided by the line of hyperplane in two parts, each class on either side of the plane. It has been one of the most conspicuous algorithms in machine learning which is used for multi-class as well as binary classification. It has excellent supervision quality and very much popular for its simplicity. (Mathur & Foody 2008)It is utilized from simple to complex classification problems. SVM is grounded on perceptron and the simplest model of Neural Network. It is most commonly used for fraud detection, recognition of pattern and detecting spam. It also works pretty well on data which is non-linear in nature. Hence, it is used as a baseline model for traffic sign image detection.

Extreme Gradient Boosting

Another method of ensemble learning is approached by XGBoost or Extreme Gradient Boosting. At times, it may not be adequate to depend upon the machine learning model evaluated results. Ensemble learning offers a methodical answer to gather multiple learners' predictive power. The resultant is a solitary model which gives the total output which is aggregate of several models. The models that structure the ensemble, otherwise called base learners, could be either from a same or distinctive learning algorithms. (Gao et al. 2017)Two broadly utilized are boosting and bagging. Despite the fact that these two procedures can be utilized with a few statistical models, the most prevalent use has been with decision trees. It is a propelled version of Gradient Boosting which uses regularization, parallel preparing, pruning and also greatly handles missing values to balance over fitted model and bias. Nowadays XGBoost is considered as the sovereign of all approaches in machine learning.

KNN (k-Nearest Neighbors)

KNN algorithm is adopted for the regression predictive and classification problems. The three aspects which are generally consider to implement the technique are: Ease of interpreting output, calculation time and the predictive power. The pseudo code for KNN is implemented in the following way. First data is loaded and the value of K is initialized. To get the predicted class, iteration is done from 1 to training data points. The distance between each row of training data set and the test dataset is calculated. The Euclidian distance is most popular method to evaluate this distance. Later, the calculated distance is sorted based on distance values in an ascending order. From this sorted array we get the top K rows and the most frequent class of this rows. Finally, the pseudo code returns to the predicted class.

3.5 Evaluating Parameters

Once all the proposed models are trained for the training dataset, the next step is to use the testing dataset and test on them. It is very much essential to understand and to have an idea of how well was each model was able to detect the traffic sign images so to find the best model

for our SDSS (Smart Driving Support System). The metrices that has been chosen for each of the model evaluation is Precision, Accuracy, Recall and F1 score. Accuracy of higher value infers capability and efficiency of these models to detect and recognize the road traffic sign images. Sensitivity is also referred as True positive rate and the capability to accurately classify the positive class is given by Recall. The model's overall test of accuracy and the balance between recall and precision is given by F1 score.

Classification Report

The summary of all prediction results for a classifier is given by confusion classification report. The report gives a summary of correct and incorrect predictions that are summarized and broken down by an individual class. It also shows us the insights of the types of errors being made by the classifier. For binary outcome we calculate in the form of confusion matrix, where we need a testing dataset with the expected outcomes.

Accuracy or Classification Rate

Accuracy is given by the following relation. For both kinds of errors, it considers equal costs for both. An accuracy of 99 per cent can also be excellent or terrible. The rate is completely dependent on the problem. Mathematically, it is evaluated as the total number of accurate predictions divided by the total number of the dataset. The best accuracy is always '1' and the worst is '0'.

Precision

The evaluation of precision is formulated from link[2] as the division of total number of correctly classified traffic signs by the total predicted traffic signs. A classifier with higher precision indicates that the model labelled is indeed positive.

$$Precision = \frac{Number\ of\ correctly\ detected\ signs}{Number\ of\ detected\ signs}$$

Recall

It is defined as the total number of correctly classified traffic sign images divided by the total number of traffic signs. Higher value of recall indicates that the class of traffic sign is correctly identified. The formula of Recall is given by:

$$Recall = \frac{Number\ of\ correctly\ detected\ sign}{Number\ of\ true\ signs}$$

F-Measure

The value of F-measure is always close to the values of Recall or Precision. This parameter helps to get a measurement which very well represents Precision and Recall. The formula for F-measure is given by.

$$F\ measure = \frac{2\ x\ Recall\ x\ Precision}{Recall + Precision}$$

4 Design Specification

In this section we will describe about the specification. The architecture in the figure illustrates the project specification. In this section we will describe about the specification. The architecture in the figure illustrated the project specification. The detailed specification below is based on CRISP-DM methodology. This first step of our experiment starts with downloading the dataset. The dataset is gathered from GTSRB (German Traffic Sign Road Benchmark) website. The data contains approximately 50 thousand images from 43 classes of different road traffic signs clicked under various lightning conditions, different sizes, occlusions and the images resemble to the data of real-life. The downloaded images are in 'ppm' (Portable Pixel Map) format of 250x250 and 15x15 pixels. They are converted to 'png' format in python. In this procedure we will also crop the sign images from the coordinates which are provided in

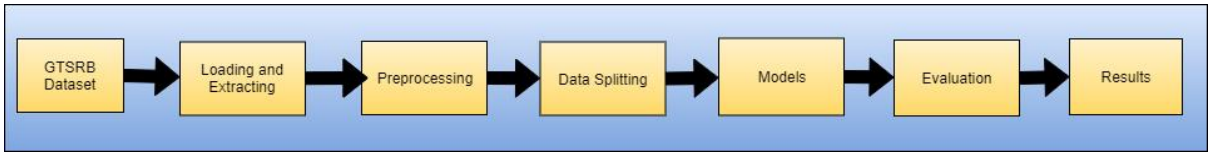


Figure 4: Design flowchart

the additional CSV file in such a way that only images of signs are visible. This is achieved by calling out a function labeller.py. The images are downloaded with their annotations for both testing and training datasets. The extended annotations that includes class id's for test set is also gathered. The next step is pre-processing of the data which involves techniques for adjusting the intensity scaling and rotation in a range of angles, brightness adjustment, colour normalisation is systematically deployed. They are grounded on sk-image and open-cv. The brightness adjustment is done to adapt the various lightning conditions in the environment where the sign is present. The sign which appears are captured from various angles from the camera installed in the car. The rotation is done to identify the signs from different angles. Also, signs will appear differently from different viewing angles with certain degree of distortion. Also, the size of sign differ is terms of scale due to the distance between the vehicle and the sign board. Some gets bigger with the vehicle moving closer to it. Therefore, scaling of images is done to get different sizes of sign. Also, sometimes due to dust and fog the sign images are not clearly visible to driver. So, we also deploy the technique of blurring and smoothing to produce the similar conditions. The blurring of images is done using Gaussian ratio levels on the set of images. Brightness adjustment is deployed to produce variant lightning conditions like night, cloudy and shady. In Next stage we split the dataset (GTSRB) into training and testing folders after pre-processing it with the list of parameters. We split the dataset in the ratio of 80-20 percent. This implies that in our models the training is done on 80 percent of the data and later we test it on remaining 20 percent of the images. Subsequent stage in our architecture includes enacting TensorFlow and Keras in the environment with the goal that the program is properly functioning for our convolutional neural network model which correctly identify the signs where pre-processed images are fed as an input. After that we implement the remaining proposed model (Random Forest, K-nearest neighbour, SVM and

XgBoost) for the training dataset and evaluate which model performs better with a good accuracy and less run time needed for operation. Later, we create the model based on the level of accuracies and test them on testing dataset to evaluate the final accuracy as the output. The output of each model after training on train data set is further tested on test data to calculate the final test accuracies of each of the model implemented. Based on the level of accuracies we find out the best fit model for our SDSS (Smart Driving Support System).

5. Implementation

The most crucial part of this research project is the model implementation. In implementation we will follow the architecture design and the methodology based on figure 6. It gives us an idea about the detailed implementation right from extraction of data till the results.

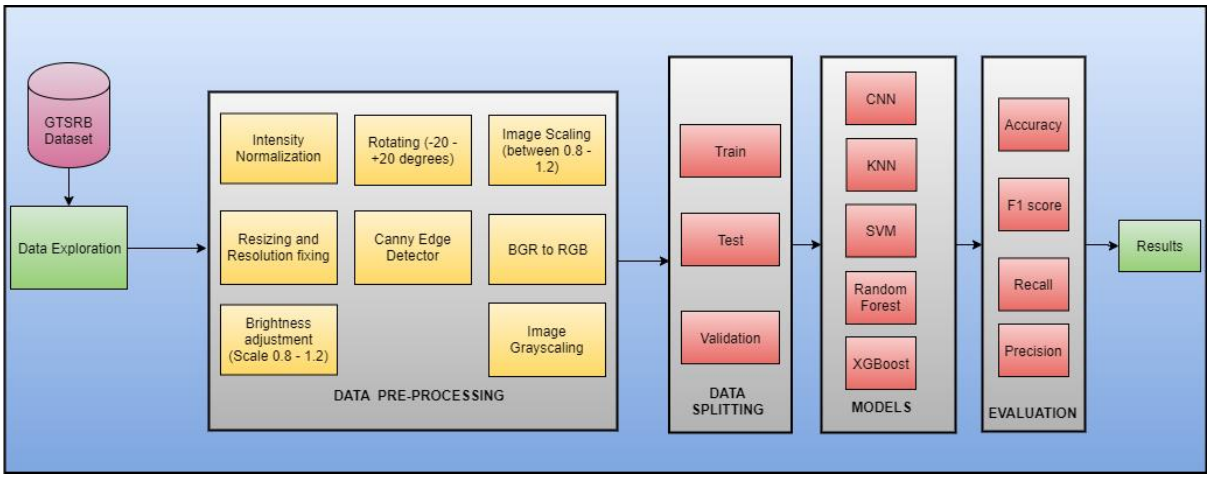


Figure 6: Detailed Implementation

5.1 Initial platform setup

For the initial environment setup, we do the following things. First involves installation of anaconda navigator for coding in Spyder. We make use of python version 3.5 for scripting and running our codes. In the environment setup of Anaconda, we download essential packages like TensorFlow and Keras. Other required libraries like pandas, seaborn, matplotlib, scikit-learn, numpy and opencv are required for installation. Failing to install anyone of them will give some error.

5.2 TensorFlow environment

Google developed TensorFlow library to accelerate the deep neural network and machine learning algorithms. It was developed to run on multiple operating systems like CPUs and GPUs. Several wrappers of TensorFlow is used in languages like C++, Java and Python. Architecture works in three parts: Pre-processing the data, building the model and training and estimating the model. It takes multi-dimensional array as an input which are also known as tensors. The input is fed at one end and then it goes through the multiple operations of whole system and is given out as output. The tensor goes through various list of operations; hence it is called as TensorFlow. The library of TensorFlow is incorporated with different API for developing deep learning architecture like RNN or CNN.

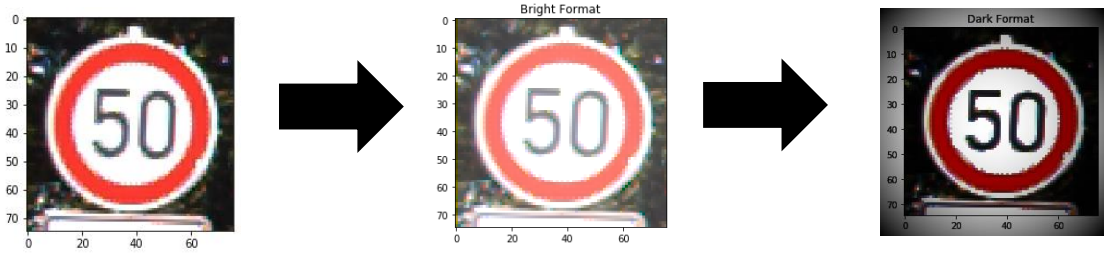
5.3 Data downloading

Our model implementation starts with downloading the data from GTSRB website[3]. The data contains folders of images in 'PPM' format. To convert it into 'PNG' we make use of the 'Pillow' or 'PIL' module imported in python for format conversion. After converting the image file format, a new directory is created and all the converted files are stored in the new formed directory. Data exploration: The GTSRB dataset which we are using to implement our models has 43 classes of sign images with almost 50 thousand image files. All classes are well explained in the methodology section above.

5.4 Data Exploration and Pre-Processing

The below parameters describe the various parameters which we have used and implemented for the purpose of data pre-processing. For image transformation and pre-processing we make use of OpenCV library.

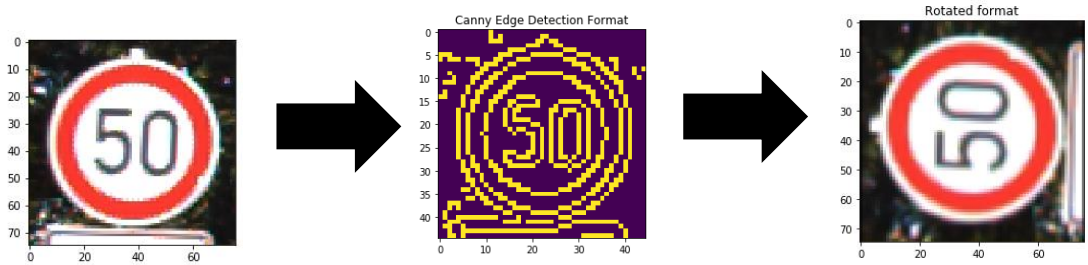
Exposure adjustment: The original image is set with the parameters of gamma=0.5 and gain=1for adjusting and changing the exposure of an image. The images below show the converted image after increased and decreased brightness.



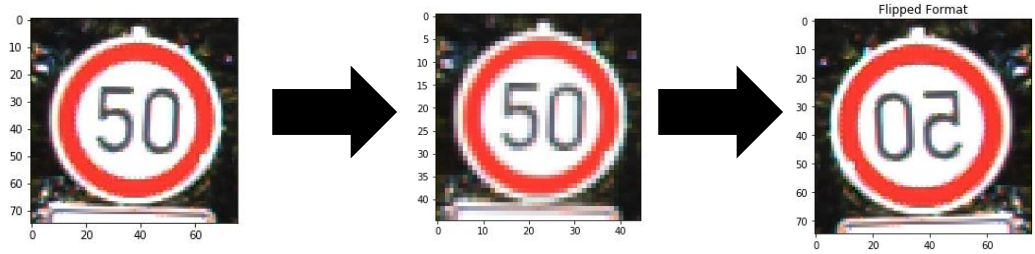
Transformation:

Canny Edge Detection: This detection method is very much useful to detect the edges of sign. Once the edges are detected, it becomes easy to further identify and classify the signs. The processed image after implementing Canny Edge Detection looks like the below image. The single function which we use for this is cv2.Canny().

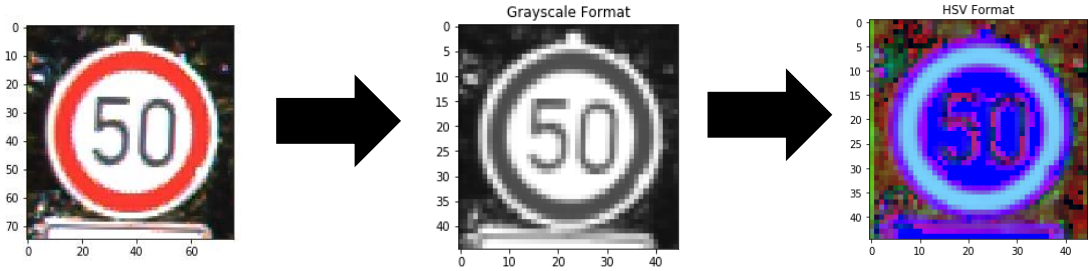
Rotation: The image is rotated by an angle of θ and the scaled rotation can be achieved with a centre of rotation that can be adjusted which allows us to rotate in of our preferred location. The matrix transformation is found by using OpenCV by the function named cv2.getRotationMatrix2D.



Scaling and Flip: When resizing of image is done it is said to be a scaled image. For this purpose, we use function cv2.resize() which comes in OpenCV. We specify the scaling factor manually for which the image can be resized. The scaled resized image looks like the image below.



Changing colour space: In OpenCV there are 150 plus methods of colour-space conversion available. The images are converted from colour space to grayscale space. First, we convert BGR image to RGB colour space. We also make use of this colour space in a way that colour is extracted from the image. The functions which are used are cv2.inRange() and cv2.cvtColor(). To convert from BGR to HSV the flag cv2.COLOUR_BGR2HSV is used. While comparing these values with the value of OpenCV the range is normalised. Both, the converted images to colour space looks like the images given below.



5.5 Implementation of Convolutional Neural Network

The necessary phases for implementation of CNN are model construction, training, testing and evaluation. With the object we begin by model = Sequential (). To define the layers and adding them with their type; model.add (*type_of_layer ()*). We also pass the following arguments for ImageDataGenerator.

Rescaling: The image is rescaled along its width and height. Some fraction of the total width and height is used for rescaling the image. For rescaling we use, rescale = 1.0/255.0

Zoom_range: we use this for random zooming at zoom_range = 0.2

Rotation: It is used to rotate the image randomly at a given degree in the range of 0-180 degrees.

The architecture of CNN model in implementation of code is includes of three layers. The convolution layers or Conv 2D with Relu and Max Pooling 2D layers. Relu and Max 2D are the nonlinear and pooling layers. In our first layer Conv 2D where we have defined the amount of output filter to be 32. The (3,3) defines the kernel size which also determines the height and width of the convolution window. The input array of the first convolution layer is in the form of pixels. Relu is the activation function of the model. the function sets the threshold. The function helps in eliminating unnecessary details in the channel. For the spatial data the pooling

operation is achieved by the Max pooling 2D. (2,2) denotes the size of pool. We now have two layers that are connected layers. The next is densely connected layer with Relu function and output value of 64. It also prevents the problem of overfitting and follow Dropout. Overfitting occurs when the model recognizes the training set samples but it has a poor performance on the test samples. Our next step is compilation of the model. It includes a function for binary cross entropy loss. For recurrent neural networks we use the optimizer algorithm. The performance of the model is given by 'Accuracy'. The number of training samples is given by batch size. Then the Image Data Generator which is already created is added for our training and testing samples with a new transformation of rescaling. It helps in data multiplication. The directory path for the training and testing folders are specified followed by the target size. The height and width to which images will be resized is mentioned. We also add the batch size. With the help of model.fit_generator training is possible. The number of epochs is indicated to define the number of times the training data will repeat. Also, we define the steps_per_epoch and validation_steps. The steps per epoch is calculated from dividing the total number of samples for training by the batch size. The validation_steps is the batches of samples. The model is trained and saved. The running of the whole implementations takes some time. In the end you get the validation accuracy which shows the models ability to test on new data. Finally, it's time to check the accuracy on the testing data. In the end we do visualization of epochs, accuracy and loss.

5.6 Implementation of other classification models

After evaluation of the 'Accuracy' from the implemented CNN model we further implement the other conventional classifiers to satisfy our secondary objective. The dataset is been split in the ratio of 80-20 percent of training and test ratios for all our models. By splitting in this ratio, the configuration homogeneity is sustained across all the models which allows us to compare the models easily. We start with the implementation of XGBoost (eXtreme Gradient Boosting) model. To implement this, we call the library 'xgboost' in python. We prepare the data from file and use it for the purpose of training and testing for our XGBoost model. We import the necessary functions such as numpy. We load the file in the form of NumPy array using the NumPy function. We fit the XGBClassifier on our training data using the function model.fit(). Parameters that are used in training the model are passed on further to constructor. Basically, XGBoost makes predictions that are probabilities. Finally, we can use the fit model for our testing datset. We also deduce the accuracy for the rest of the classifiers by using the function scikit-learn as accuracy_score(). We also evaluate the classification report to check the performance of the model on individual class.

The third model which we implement is Random Forest. To implement this model, we first we need to standardise the input and use the function RandomForestClassifier. This function also allows fitting of the training data. The implementation starts with selecting random samples from the dataset and constructing a decision tree for each of the sample and get an individual prediction for each decision tree. After splitting the dataset, we will train the training dataset and predictions are performed on testing dataset. The fourth model which we implement is KNN or K-Nearest Neighbour.

Further we do some exploration of the classes in the dataset. Later, we split the data into training and testing. To import the KneighboursClassifer we create KNN classifier in the form of KNeighboursClassifier () function. Then we fit the function on the model for training dataset using function fit () and later perform the predictions using predict () on testing dataset. The last model which we implement is SVM (Support Vector Machine). We use the linear kernel to fit the training as well as testing data. We use the necessary inbuilt functions such as OpenCV and sklearn. We create a SVM classifier by importing SVC (support vector class). After dividing the dataset into training and testing we train the SVM classifier on training dataset. The Scikit-Learn module has the library svm, which has built in classes for all SVM algorithms. In SVC class we use the parameter fit () to train the training dataset. The 'predict' Method is used in svc class to make the predictions. Further, we testing the evaluation on our testing dataset and deduce the prediction for testing dataset.

6. Evaluation

In this section we will discuss about the evaluation and results generated by each of the implemented classification technique. The detailed evaluation of each classifier is given by a classification report. The classification report gives us the summary of the performance of the classification model. The report shows the important classification metrics for every class. The important metrices which are a part of classification report are 'Precision', 'F measure', 'Support' and recall. The report also consists of macro average, weighted average, sample average and micro average. This section also includes graphical classification report for each of the classifier implemented. Further, this section also comprises of the 'Accuracy' of each classification technique for all 43 classes of traffic signs.

Accuracy and Performance of all Models

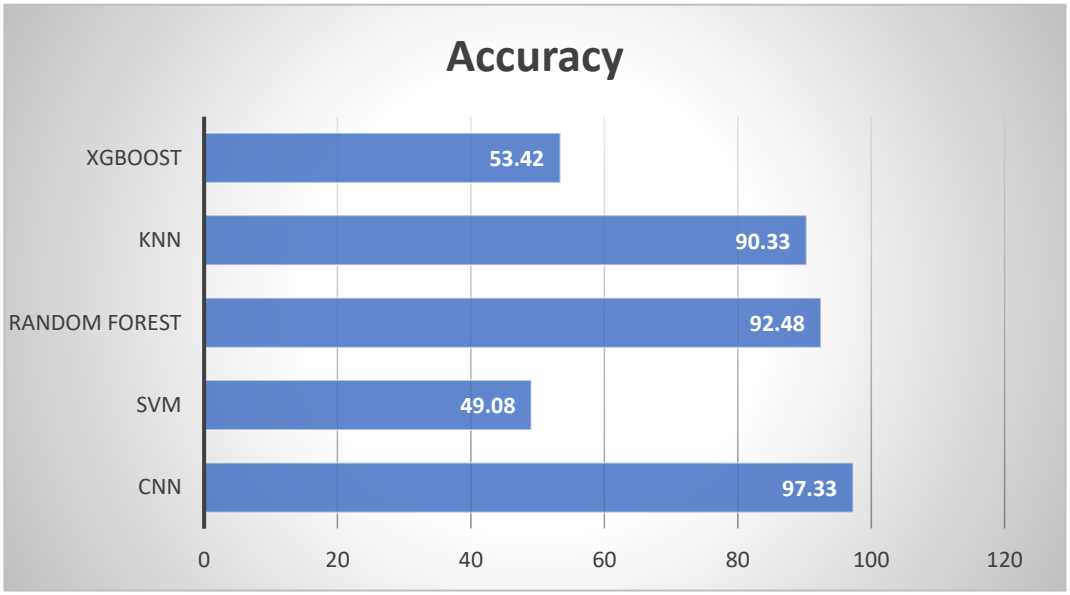


Figure 2: Accuracies of all Models

From the above Fig 2, it is clearly visible that the highest accuracy of 97.33 percent was obtained for CNN model. It means the model is good in performance when compared to other modelling techniques. The lowest accuracy was measured for SVM of 49.08 percent and second lowest was for XGBoost with 53.42 percent. The accuracies for Random forest and KNN were recorded as 92.48 and 90.33 percent. These models also performed quite better.

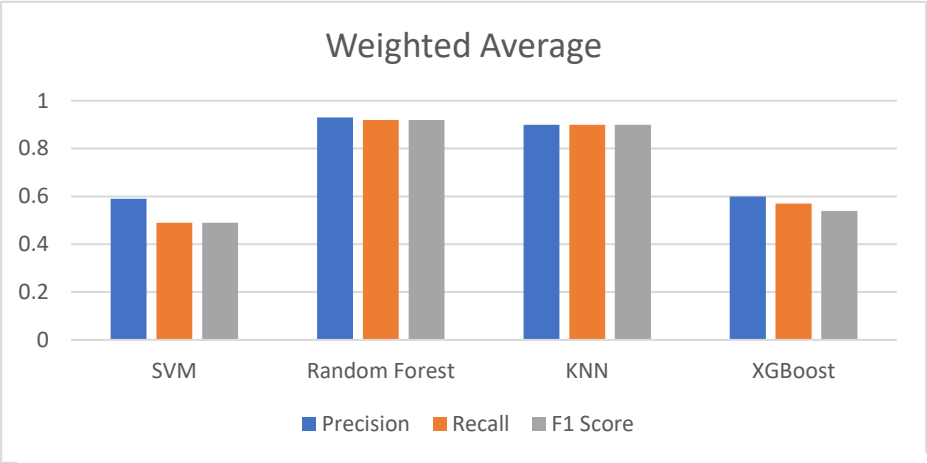


Figure 3: Weighted average of Precision, Recall and F1 Score

The above figure 3, shows the weighted average of Precision, Recall and F1 Score for all the models. The Harmonic mean of Recall and Precision is given by F1 score. Generally, the F1-score is more against the max of precision and recall but closer to minimum of Precision and Recall. When both the metrices are high even F1 score rises up. But it can be low even if any one of the metrices is low. One way to give more importance to Recall and Precision is by adding weights. It is done by utilising Weighted average of Precision and Recall.

6.1 Experiment 1: Convolutional Neural Network

Convolutional Neural Network is the one of the special architectures based on artificial neural network used for image classification. The primary task of this classification technique is to accept the input image and its class definition. The implementation of this architecture starts with utilization of Keras with TensorFlow installed in backend. After implementation of convolutional neural network (CNN), the validation accuracy was 0.9877. It shows the model

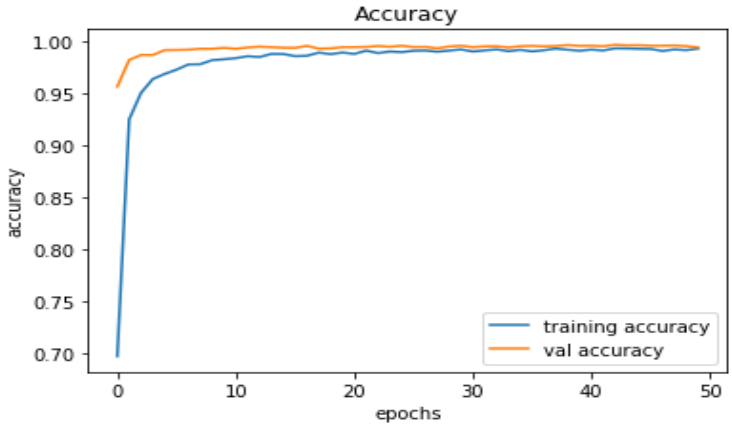


Figure 4: Training Accuracy

capability to perform with the testing data. After, implementing this model on testing data, the classification accuracy was 97.33 percent which is best and highest measured amongst all. The above graphs illustrated the relation between Accuracy and Epochs also termed as learning curve. The test data accuracy looks perfectly normal and everything seems to be fine. The model fits on training data and it modifies weights as per the data. There is no such major variation in the train accuracy so there is nothing to worry about the learning curve and regularization.

6.2 Experiment 2: Support Vector Machine

Table 1: Classification report for Support Vector Machine

Class	0	1	2	3	4	5	6	7	8	9
Precision	.97	.89	.90	.91	.94	.91	1.0	.91	.95	.99
Recall	.78	.96	.91	.93	.95	.89	.99	.86	.91	.96
F1 score	.87	.92	.91	.92	.95	.90	.99	.89	.93	.98

Support Vector Machine or SVM is used for regression and classification problems. It is a model which is linear in nature and draws a line that helps in data separation into classes. In this algorithm the traffic sign images are fed as an input and it classifies as well as separates them according to their classes. The table 1 above displays the results of this model. The final accuracy was 49.08 percent which is the lowest among all. Thus, this model would be the least choice for our implementation it in our system.

6.3 Experiment 3: Random Forest

Table 2: Classification report for Random Forest

Class	0	1	2	3	4	5	6	7	8	9
Precision	.97	.89	.90	.91	.94	.91	1.0	.91	.95	.99
Recall	.78	.96	.91	.93	.95	.89	.99	.86	.91	.96
F1 score	.87	.92	.91	.92	.95	.90	.99	.89	.93	.98

Since, we know that Random forest is used for classification as well as for regression. It is supervised learning algorithm. The basic concept behind implementing Random Forest model is very much simple. The main advantage of using this algorithm is that it does not suffer from any kind of overfitting of data. It can very well analyse multi class data which is high dimensional in nature. In this, each test image from the testing dataset passes down to each random tree and reaches leaf node. Further we take an average of all the posterior probabilities and the maximum average of all is considered as input image classification. On implementing this model on our test dataset, we got an accuracy of 92.48 percent. Overall, we can say that model performed pretty well and is considered the second-best choice after CNN for implementation in our system to detect and classify the road traffic signs on street. Other parameters of classification report are mentioned in the above table 2. The table, also shows the obtained precision, recall and f-measure for individual class.

6.4 Experiment 4: Extreme Gradient Boosting or XGBoost

Table 3: Classification report for XGBoost

Class	0	1	2	3	4	5	6	7	8	9
Precision	.83	.56	.58	.58	.47	.66	.76	.70	.56	.91
Recall	.20	.77	.72	.54	.68	.09	.37	.14	.47	.53
F1 score	.32	.65	.64	.56	.56	.15	.50	.23	.51	.67

We used XGBoost classifier for predicting the labels from a given set of image dataset. The hyperparameter specified with a function is responsible for the learning task like regression and classification. The overall classification accuracy after implementing this model was 53.42 percent. The weighted average of precision, recall and F1 score for all set of images is given in the above table 3. The table, also shows the obtained individual precision, recall and f-measure for individual class.

6.5 Experiment 5: K-Nearest Neighbour

Table 1: Classification report for KNN

Class	0	1	2	3	4	5	6	7	8	9
Precision	.93	.93	.92	.87	.90	.84	.96	.87	.91	.97
Recall	.88	.93	.88	.85	.88	.91	.96	.89	.92	.93
F1 score	.90	.93	.90	.86	.89	.87	.96	.88	.92	.90

In this model, each point of the image is represented as a vector in a 2D space. Stacking of all these vectors forms a matrix of all these point in the plane. The implementation of this model gives us the classification accuracy of 90.33%. The other metrices of classification report like Precision, Recall and F1 measure is illustrated in the above table 4 for the first 10 classes.
The above table 4, also shows the obtained individual precision, recall and f-measure for individual class.

Discussion and Results

Extensive and broad research has demonstrated that a good performance on traffic sign data sets can be obtained by various Machine Learning techniques. In spite of the fact that on successful implementation of CNN and other machine learning models was more concentrated on classifying only few classes and the other classes are still the unexplored part and that's where our research could expand in future. Also, some of the machine learning techniques are still unexplored and that could be another objective for future work. The proposed models in our report brings us a step nearer to accomplish the task of developing the perfect Smart Driving Support System and even support a framework towards driverless cars, there is still more that can be improved. For ID of a sign, this paper relies upon shading and state of the sign. This is an issue like the shading of sign can be impacted by the reflection of sun. Also, if the sign board is tapered off accidently, the state of the sign is debilitated, which creates a hurdle in detecting the traffic sign image. Another significant issue to consider is location of the sign in the night. In the event that the camera can't catch nature in

the night because of the dimness, the sign can't be identified and arranged. A module which can convert the text to an audio can be added which can provide more safety and comfort to the driver. The general execution time could likewise be improved and modified by adding more datasets and from various nations.

Results: Using CNN as a classifier we have achieved recognition rate of 97.33 percent, which is highest of all the classifiers. Other models like Support Vector Machine, Extreme Gradient Boosting, Random Forests and K Nearest Neighbour and the evaluated accuracies are 49.08%, 53.42%, 92.48% and 90.33% respectively. Thus, the results show that CNN is the best fit with the highest accuracy for our proposed system when compared to other supervised learning models.

7. Conclusion

Smart Driving Support System is the future of automatic and driverless cars. It provides an intelligent system for navigation and road safety by successfully recognizing the traffic signs. The results and evaluation of the implemented shows that, the proposed SDSS (Smart Driving Support System) can address different real time issues like variations in intensity of light, vibration of the vehicle, location of signs in night and under shadow and also the geometrically distorted traffic signs. We propose a unique technique for detecting and precisely recognizing the traffic sign using image processing and Convolutional Neural Network as our primary classifier. We have used GTSRB (German Traffic Sign Road Benchmark) dataset with images taken in different environmental setup. For the purpose of detection, we implemented canny edge detector for detecting the edges of sign images and other parameters for the pre-processing process. and for classification we utilised CNN due to its high rate of recognition which makes it more desirable for implementation of numerous computer vision tasks. By using CNN we measured the highest accuracy of 97.33 percent which makes it as our first choice for implementing it in our SDSS (Smart Driving Support System).

Acknowledgement

I am exceptionally appreciative to my supervisor Dr. Cristina Muntean for her unrelenting help and support. She is extremely helpful and knowledgeable. I equally want to acknowledge National College of Ireland for providing me the essential mentoring and relevant skills throughout my academic career while pursuing Masters. Also, I would like to thank my family for always being there for me.

References

Algorry, A.M., García, A.G. & Wofmann, A.G. 2018, 'Real-Time Object Detection and Classification of Small and Similar Figures in Image Processing', *Proceedings - 2017 International Conference on Computational Science and Computational Intelligence, CSCI 2017*, pp. 516–9.

Biswas, R., Fleyeh, H. & Mostakim, M. 2014, 'Detection and classification of speed limit traffic signs', *2014 World Congress on Computer Applications and Information Systems, WCCAIS 2014*, pp. 1–6.

Changzhen, X., Cong, W., Weixin, M. & Yanmei, S. 2017, 'A traffic sign detection algorithm based on deep convolutional neural network', *2016 IEEE International Conference on Signal and Image Processing, ICSIP 2016*, pp. 676–9.

Creusen, I.M., Hazelhoff, L. & De With, P.H.N. 2012, 'Color transformation for improved traffic sign detection', *Proceedings - International Conference on Image Processing, ICIP*, pp. 461–4.

Fifik, M. & Turán, J. 2010, 'Real Time Recognition System for Traffic Sign Detection and Classification', *Pace Pacing And Clinical Electrophysiology*, pp. 284–7.

Fistrek, T. & Loncaric, S. 2011, 'Traffic sign detection and recognition using neural networks and histogram based selection of segmentation method', *ELMAR, 2011 Proceedings*, no. September, pp. 51–4.

Gao, X., Fan, S., Li, X., Guo, Z., Zhang, H., Peng, Y. & Diao, X. 2017, *An Improved XGBoost Based on Weighted Column Subsampling for Object Classification*, no. Icsai, pp. 1557–62.

Girshick, R., Donahue, J., Darrell, T. & Malik, J. 2014, 'Rich feature hierarchies for accurate object detection and semantic segmentation', *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580–7.

He, K., Zhang, X., Ren, S. & Sun, J. 2015, 'Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–16.

Höferlin, B. & Zimmermann, K. 2009, 'Towards reliable traffic sign recognition', *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 324–9.

Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M. & Igel, C. 2013, 'Detection of traffic signs in real-world images: The German traffic sign detection benchmark', *Proceedings of the International Joint Conference on Neural Networks*, no. May 2014.

Jain, R. & Gianchandani, P.D. 2019, 'A hybrid approach for detection and recognition of traffic text sign using MSER and OCR', *Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2018*, pp. 775–8.

Jurišić, F., Filković, I. & Kalafatić, Z. 2016, 'Multiple-dataset traffic sign classification with

OneCNN', *Proceedings - 3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015*, pp. 614–8.

Kavitha, D., Hebbar, R., Vinod, P. V, Harsheetha, M.P., Jyothi, L. & Madhu, S.H. 2018, 'of Field Photographs', *2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C)*, pp. 59–63.

Liu, W. & Maruya, K. 2009, 'Detection and recognition of traffic signs in adverse conditions', *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 335–40.

Maldonado Bascón, S., Acevedo Rodríguez, J., Lafuente Arroyo, S., Fernndez Caballero, A. & López-Ferreras, F. 2010, 'An optimization on pictogram identification for the road-sign recognition task using SVMs', *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 373–83.

Man, W. 2018, 'Image Classification Based on Improved Random Forest Algorithm', *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, pp. 346–50.

Mathur, A. & Foody, G.M. 2008, *Multiclass and Binary SVM Classification : Implications for Training and Classification Users*, vol. 5, no. 2, pp. 241–5.

Qian, R., Zhang, B., Yue, Y., Wang, Z. & Coenen, F. 2016, 'Robust Chinese traffic sign detection and recognition with deep convolutional neural network', *Proceedings - International Conference on Natural Computation*, vol. 2016- Janua, pp. 791–6.

Rahmad, C., Rahmah, I.F., Asmara, R.A. & Adhisuwignjo, S. 2018, 'Indonesian traffic sign detection and recognition using color and texture feature extraction and SVM classifier', *2018 International Conference on Information and Communications Technology, ICOIACT 2018*, vol. 2018-Janua, no. c, pp. 50–5.

Shustanov, A. & Yakimov, P. 2017, 'CNN Design for Real-Time Traffic Sign Recognition', *Procedia Engineering*, vol. 201, pp. 718–25.

Siegler, A. & Kardkov, Z.T. 2011, *Real-time traffic sign recognition system Real-time Traffic Sign Recognition System*, no. July 2014.

Swathi, M. & Suresh, K. V. 2018, 'Automatic traffic sign detection and recognition in video sequences', *RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings*, vol. 2018- Janua, pp. 476–81.

Torresen, J., Bakke, J.W. & Sekanina, L. 2005, *Efficient recognition of speed limit signs*, vol. 90, pp. 652–6.

Wang, C. 2018, 'Research and application of traffic sign detection and recognition based on deep learning', *Proceedings - 2018 International Conference on Robots and Intelligent System, ICRIS 2018*, pp. 150–2.

Wu, J. & Tsai, Y. (James) 2013, 'Real-time Speed Limit Sign Recognition Based on Locally

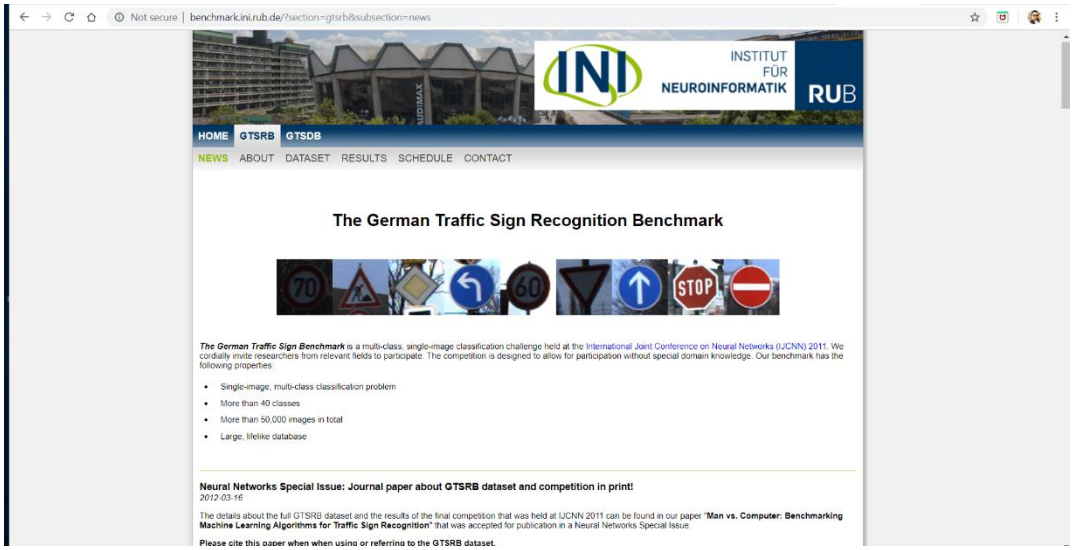
Adaptive Thresholding and Depth-First-Search', *Photogrammetric Engineering & Remote Sensing*, vol. 71, no. 4, pp. 405–14.

Yuan, Y., Xiong, Z. & Wang, Q. 2017, 'An Incremental Framework for Video-Based Traffic Sign Detection, Tracking, and Recognition', *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1918–29.

Zavadil, J., Tuma, J. & Santos, V.M.F. 2012, 'Traffic signs detection using blob analysis and pattern recognition', *Proceedings of the 2012 13th International Carpathian Control Conference, ICCC 2012*, pp. 776–9.

Appendix

The necessary dataset required to implement the models of this Research Project is



downloaded from the URL: http://benchmark.ini.rub.de/?section=gtsrb&subsection=news