A Study on Automatic License Plate Detection and Recognition in the Context of Bangladesh

By

Sheikh Nooruddin
Roll: 1507062
&
Falguni Ahmed Sharna
Roll: 1507065

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh
February, 2020

A Study on Automatic License Plate Detection and Recognition in the Context of Bangladesh

By

Sheikh Nooruddin
Roll: 1507062
&
Falguni Ahmed Sharna
Roll: 1507065

A thesis submitted in partial fulfillment of the requirements for the degree of
"Bachelor of Science in Computer Science & Engineering"

Supervisor:

Dr. Sk. Md. Masudul Ahsan
Professor
Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh.

_____
Signature

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh
February, 2020

Acknowledgment

All the praise to the almighty Allah, whose blessing and mercy succeeded me to complete this thesis work fairly. After that, we humbly acknowledge the valuable suggestions, advice, guidance and sincere co-operation of Dr. Sk. Md. Masudul Ahsan, Professor, Department of Computer Science and Engineering, Khulna University of Engineering & Technology, under whose supervision this work was carried out. His intellectual advices, encouragement and guidance make us feel confident and scientific research needs much effort in learning and applying and need to have a broad view at problems from different perspective. We would like to convey our heartiest gratitude to all the faculty members, official and staffs of the Department of Computer Science and Engineering as they have always extended their co-operation to complete this work. Last but not least, we wish to thank our friends and family members for their constant supports.

Abstract

With the surge in motorized vehicle numbers in the modern era, Automatic License Plate Detection and Recognition (ALPDR) systems have become necessary for proper management of vehicles in roads. Automatic License Plate Detection and Recognition system detects and localizes license plates from provided images, and recognizes the characters in the images. The recognized characters can then be used to find relevant information from central databases and to take necessary actions. In this thesis, we have worked with detection and recognition of license plates in the context of Bangladesh. We have built our own dataset for commercial vehicles by capturing images of over 650 vehicles in dynamic background scenarios and different camera perspectives. In the detection phase, Grey Level Co-occurrence Matrix and Color Histogram with MinPool and MaxPool features (CHPOOL) were tested. Between them, Color Histogram with MinPool and MaxPool features combined with Random Forest Classifier performed better. A sliding window approach was used in the detection phase to perform both detection and localization. In the recognition phase, horizontal and vertical histograms have been used for character segmentation and KNN has been used for character recognition. The effects of various similarity measures such as Manhattan, Euclidean, Squared Euclidean, Cosine similarity and Bhattacharyya distance to measure the minimum distance and varying values of K have been observed. The developed automatic license plate detection and recognition system performed effectively in both the detection and recognition phases. A major limitation of the developed system is that it works effectively in case of colored license plates. The effectiveness of the license plate system can be improved by training on input data under different lighting conditions and different color profiles. A Region Proposal Network (RPN) and parallel processing can further be used to significantly speed up the detection and localization process.

Contents

List of Tables

List of Figures

CHAPTER I

Introduction

1.1 Introduction

Automatic License Plate Detection and Recognition (ALPDR) is the process of detecting a license plate region from an image, localizing and cropping the license plate, and recognizing the contents of the license plate. Due to an increase in both commercial and private vehicles, automated systems such as automatic traffic control systems, automatic license plate detection and recognition systems have become a necessity for proper management of vehicles in the road.

According to Bangladesh Road Transport Authority (BRTA), a grand total of 3419884 registered motor vehicles exists in Bangladesh. Among these, there are around 135081 trucks, 335660 private cars, and 44374 buses [1]. There is also a huge amount of non-registered vehicles that are unaccounted for. The sheer number of vehicles in the roads in Bangladesh require automated management systems for proper management, however, very few such systems exist. Traffic jams, road accidents, hit and run incidents, etc. are among numerous effects of the large number of vehicles and manual control systems [2]. ALPDR systems enable law enforcement agencies to properly monitor the vehicles in the road and swiftly take actions against responsible parties in case of an accident. Applications of ALPDR systems include automated parking systems, access control systems, automated tolling systems, automated traffic control systems, border control systems, etc.

1.2 Problem Statement

Generally, license plates are put in both the front and the back of vehicles. An automatic license plate detection and recognition systems take such an image as input. The system then localizes the license plate area in that image. This is the detection part. The system also

(a) (b)

(c) (d)

Figure 1.1: Sample license plates (a) License plate in back (b) license plate in front (c) A license plate registered in Gazipur (d) A license plate registered in Dhaka.

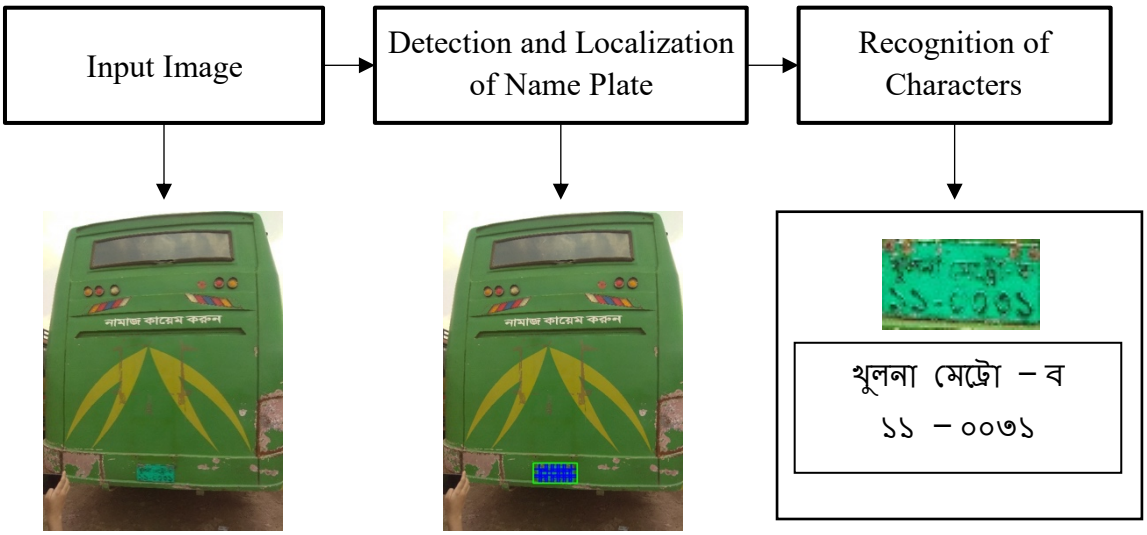


Figure 1.2**:** Flow diagram of Overall ALPDR.

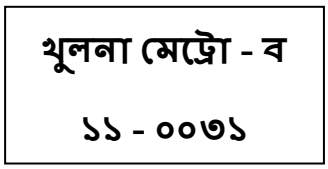


Figure 1.3: A sample output of recognition system

recognizes the characters and numerals in the license plate. This is the recognition part. The recognized text can then be used for various purposes by law enforcement agencies if the system is connected to a central database. Figures 1.1(a) and 1.1(b) show general license plate regions in vehicles. Figures 1.1(c) and 1.1(d) show general Bangladeshi license plates. Figure 1.2 depicts the flow diagram of the overall ALPDR system. At first an image containing the vehicle with the license plate is taken as input. The detection and localization system detected and localized the license plate, the characters are then recognized from the localized license plate.

1.3 Bangladeshi License Plate Convention

The registration numbers and the license plates that are used in vehicles are generally issued by BRTA in Bangladesh. Bangladeshi license plates generally use Bangla numerals and characters to form the registration numbers. However, some vehicles also used license plates printed in English numerals and characters. Bangladeshi license plates are generally printed on metal sheets. Most commercial vehicles such as buses and trucks are required to present a metal-sheet license plate with green background and black text for ease of detection and recognition. The license plates must contain two lines. The standard format is “City Name -Vehicle Class Letter” in the upper line and “Vehicle Class Number – Registration Number” in lower line. Figure 1.3 presents a sample output of a recognition system. In this case “খুলনা মেট্রো” is the city name. “ব” is the vehicle class letter. “১১” is the vehicle class number and “০০৩১” is the registration number. According to BRTA, these have to be written in Bengali numerals and character, printed on flat metal plate, and the total plate dimension must be 524 mm × 112 mm. The plate must be put both in front and in the back of the vehicles.

1.4 Objectives

The main objective of this thesis work is to develop an automatic license plate detection and recognition system that would work in the context of Bangladesh and in the process learn about the various image processing and computer vision algorithms, image feature extraction techniques, machine learning, and the successful synthesis of these knowledge to solve real world problems. As open-access datasets are paramount to research initiatives, one of our main objectives is to introduce the first-ever open-access Bangladeshi vehicle license plate dataset. We introduce a novel feature extraction method for the license plate detection and recognition task. We also present our own methodology for the character segmentation and recognition phase.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows:

- Chapter 2 presents short summaries on the related research works on automatic license plate detection and recognition.
- The various methodologies that were pursued along the course of this thesis work are described in detail in Chapter 3.
- Chapter 4 presents the experimental results of our proposed methodologies.
- The limitations, future works and concluding remarks for this thesis are presented in Chapter 5.

CHAPTER II

Literature Review

2.1 Introduction

Various techniques have been employed to successfully perform the detection and recognition tasks. Some systems employ general image processing techniques, while others used various deep learning architectures to perform the detection and recognition tasks. A lot of the systems also used existing Optical Character Recognition (OCR) systems for the recognition tasks. A lot of the detection systems used edge detection techniques such as Sobel edge detectors [3, 4], shape descriptors such as Hough line transformations [5, 6], feature descriptors such as Histogram of Oriented Gradients (HoG) [7-9], etc. Recognition systems that used novel techniques mostly used various forms of template matching technique to recognize the characters and numerals. Deep learning based systems used Convolutional Neural Network (CNN) [10-12] and other transfer learning approaches [13] to perform both the detection and recognition tasks. The reviewed literature has thus been described in two sections: Image processing based approaches and Machine Learning and Deep Learning based approaches.

2.2 Image Processing Based Approaches

Sulaiman et al. [14] used Mexican Hat Filter (Laplacian of Gaussian (LoG)) for noise reduction and edge identification. Image Cropping functions were then used to crop the detected area. In the recognition phase, the boundary areas of the black pixels were detected. After the boundary was completed, the characters were segmented and template matching was used to find the appropriate matching character. This system only works when the letters are painted in black color. The system was developed in the context of Malaysia.

Sathya et al. [15] provided a detailed comparison of the different algorithms used in various steps in an ALPDR system in the context of India. For thresholding, global, adaptive, and hysteresis thresholding were compared. The application of Canny, Sobel, LoG, Robert, and

Prewitt operators in edge detection were compared. Among various morphological operations, opening, closing, dilation, and erosion were compared. They recommended Canny Edge Detection technique for edge detection, Hough Transform for plate area detection, Top-hat transform for morphological operation, horizontal and vertical projection for character segmentation, and Hybrid Discriminative Restricted Boltzmann Machines (HDRBM) for character recognition according to their research.

Choudhury and Negi [16] used information such as license plate color, font information, license plate dimension information etc. for detection and recognition purposes in the context of India. The license plates were first converted to greyscale; then bilateral filtering was applied to reduce noise while preserving necessary edge information. The horizontal and vertical histograms for the images were generated. The horizontal histogram was smoothed using low pass filter, and the vertical histogram values were filtered by applying dynamic thresholds. Probable candidate regions for number plates were selected after zone based identification. Connected Component Analysis was performed for segmenting the characters. Template matching was then performed for identifying the characters.

John et al. [17] used dynamic threshold on preprocessed gray images to detect rectangular shape of the plates in the context of India. Then opening was performed on the images for noise removal. Connected component analysis was used for proper segmentation and template matching technique was used for character matching.

Shahed et al. [18] proposed an OCR algorithm only for license plate recognition in the context of Bangladesh. However, the dataset contains only 30 samples of closely captured license plate images. Like the previous approaches, after preprocessing of captured images, opening was performed for noise reduction, Sobel edge detector was used for edge detection, and finally connected component analysis was performed for segmentation. Template matching with cross correlation was performed to match the characters.

Hommos et al. [19] also proposed an ANPR system in the context of Qatar. They used mathematical morphological operations and Connected Component Analysis to detect and localize the characters in the number plate. Template matching with cross correlation was finally used to match the characters.

Lubna et al. [20] used edge detection techniques for localization and segmentation in the context of Pakistan. Various edge detection filters such as Gabor, Log Gabor, Sobel, and Canny were compared. Among them, Gabor and Log Gabor filters output clearer segmentation than Sobel and Canny edge detection techniques.

Islam et al. [21] used median filter for removing unwanted noise from the image in the context of India. Median filters perform very well in removing salt and pepper noise in some scenario. After noise removal, Canny Edge Detection was used to find the horizontal and vertical edges in the image. Morphological operations such as opening was performed for license plate recognition. The largest area in the image was selected. CCA was used for character segmentation and template matching was used for character recognition.

2.3 Machine Learning and Deep Learning Based Approaches

Rahman and Islam [22] used five individual process, each consisting of different morphological processing and filtering approaches for license plate detection and a Convolutional Neural Network (CNN) for number plate recognition. The five individual processes solved the problems related to non-standard height, weight and aspect ratio in Bangladeshi vehicle license plates. This system was developed in the context of Bangladesh.

Quiros et al. [23] detected and localized license plate images from video data in the context of Philippines. Background segmentation methods were used to detect the license plate regions. Upon obtaining the plate image, contour detection was performed to find the characters. The contours were then evaluated based on their size, aspect ratio, etc. Validated contours were segmented using their respective contours. The contours were later classified using KNN algorithm.

Kim et al. [24] built an ALPDR system in the context of Korea. After noise removal using Gaussian Blur filter, Canny Edge detection and morphological closing operations were performed. Affine transformation was used for correcting geometric distortions. The license plate areas were cropped from the images. Car areas and backgrounds were identified as non-license plate regions. These regions were then classified by using Support Vector Machine based on their pixel information. Later from detected license plates, contour detection was performed for character segmentation. KNN was used for character and numerals recognition.

Bhardwaj and Kaur [25] preprocessed the images using noise removal techniques. Later Sobel edge detector was used for horizontal and vertical edge detection. Six zonal regional features were proposed for feature extraction. The extracted features were passed to K-start algorithm for further detection. The system was developed in the context of India.

Hamey and Priest [26] preprocessed the images to reduce noise. Wiener filter was used to reduce overall noise. Later they used an Artificial Neural Network (ANN) architecture for proper both license plate detection and character recognition. The system was developed in the context of Australia.

Prabhu et al. [27] compared OpenALPR, K-NN and Convolutional Neural Networks (CNN) based approaches for successful license plate detection and recognition in the context of India. Out of all the systems, CNN performed the best. CNN can be used for both feature extraction and detection purposes. CNN can be used in both license plate detection and character recognition.

Mondal et al. [28] used CNN for ALPDR. Some preprocessing operations were performed on the images before passing them to CNN. Artificial augmentation such as random rotation, scaling, shearing, etc. was performed on the images using data generators. The system was developed in the context of India.

2.4 Discussion

The reviewed systems mostly worked with either basic image processing techniques or feature extraction along with machine learning techniques or direct deep learning techniques. In image processing based techniques, edge detection techniques such as Sobel edge detection, Canny Edge detection are mostly used. Opening and Closing are two most used morphological operations. License plate regions are mostly detected based on their area. Connected Component Analysis (CCA), contour detection techniques are mostly used for character detection. Template matching techniques were mostly used for character recognition. Machine learning based systems used either noble feature extraction techniques, or feature descriptors such as Histogram of Oriented Gradients (HoG) and Scale Invariant Feature Transform (SIFT) for license plate area detection. KNN was mostly used for license plate recognition. Deep learning based approaches mainly used CNN with

varying hidden layers, varying neuron numbers, varying optimizers, etc. for proper detection and recognition.

Most of reviewed systems were developed in the context of India, Pakistan, Philippines, etc. The license plates in those countries contain only a single line consisting of area code and vehicle identification number. In retrospect, Bangladeshi license plates have two lines with various information such as vehicle type, area code, vehicle class, registration number, etc. Thus, those systems are not readily deployable in the context of Bangladesh. The systems that worked in the context of Bangladesh, built the system with very limited amount of images from very close proximity. Moreover, there are currently no open-access Bangladeshi vehicle dataset. The goal of this thesis work is to build the first-ever open-access diverse Bangladeshi vehicle license plate dataset and to also build a reliable automatic license plate detection and recognition system in the context of Bangladesh.

CHAPTER III

Proposed Methodology

3.1 Introduction

Automatic License Plate Detection and Recognition systems deal with both License Plate Detection and Localization and License Plate Recognition. In the License Plate Detection and Localization stage, from a given image as input, the license plate region is detected and a bounding box is generated around the license plate. The process of generating a valid bounding box around the license plate region is considered as localization. In the recognition stage, the characters in the detected license plate region is recognized and serially output as strings to another program or console. The overview of the entire process is shown in Figure 3.1.

3.2 Working Procedure

The overall workflow of the system can be divided into two parts: License Plate Detection and Localization and License Plate Recognition. Both of these parts are dependent on machine learning for proper detection and recognition. In the following sections, we provide in-depth discussions on the working principles of both systems.

3.2.1 Detection and Localization System

The detection and localization system consist of preprocessing steps such as image resizing, color space conversion etc., feature extraction techniques such as the Grey Level Co-Occurrence Matrix (GLCM) features, Color Histogram with Minimum and maximum pooling features (CHPOOL), etc. and machine learning model training. The trained model is then used to detect and localize the license plate. A brief overview of the feature extraction process is illustrated in Figure 3.1. A sliding window approach has been used in our system. The window slides along the image according to set window size and step size and features are extracted from the cropped image within the window. The features are then flattened

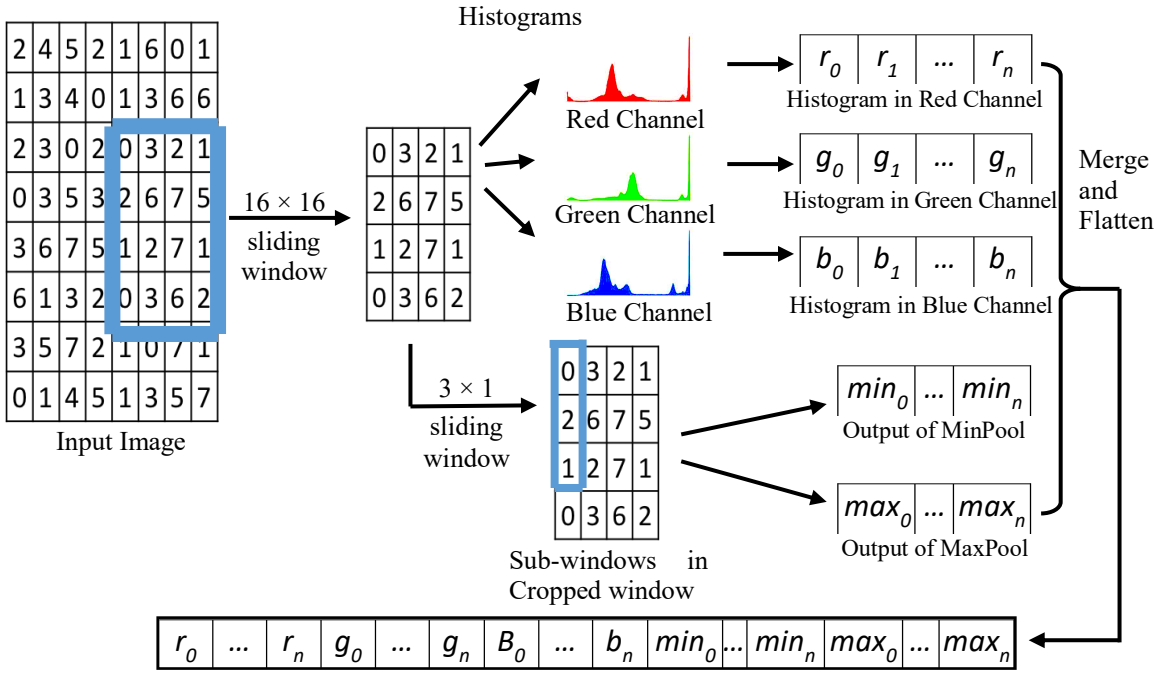


Figure 3.1: Overview of the feature extraction process for the detection and localization phase.

and concatenated together to generate the feature vector for that patch. The window then slides to the next position and features are extracted from that region. These features vectors are then used to train machine learning models. Hyper-parameter tuning is performed on the machine learning model to further improve the performance of the model. The trained model is later used to detect and localize the license plates in images.

A. Preprocessing

Preprocessing for the license plate detection and localization system includes various steps such as image resizing, color models conversion, setting proper window sizes, etc.

I. Input Image

The input image is generally in RGB color space. These images are basically images from the dataset and have varying resolutions.

II. Kernel Size and Step Size

The detection and recognition system use a sliding window approach. Basically, this process "slide" a box with a set window size and step size on an image and classifies the image crop inside the window. After classification of an image crop inside the window, the window

moves to the next position by adding the set step size. W_{size} refers to the set window size. S_{size} refers to the set step size. I_{size} refers to the image size. Image size is generally represented using a tuple (height, width, channels).

III. Image Resizing

Image resizing or image scaling is the process of resizing a digital image. When scaling a vector graphics image, the image quality can be preserved with geometric transformations. However, scaling a raster graphics image results in a new image with higher or lower number of pixels. This visibly results in quality loss. While resizing images, the new pixel position in the new image for every pixel in the original image needs to be calculated. However, most calculations result in approximate floating-point positions. As images are basically two-dimensional matrices, the approximate floating-point positions are rounded and methods such as Nearest-Neighbor interpolation or Bilinear Interpolation are exploited to fill the missing positions.

The Nearest-Neighbor Interpolation algorithm is a very simple algorithm. It chooses the pixel value of the nearest point to fill the unfilled position. It does not consider the values of the neighboring pixels, thus yielding a piecewise constant interpolant. This is commonly used in real-time rendering applications.

The Bilinear Interpolation method is an extension of linear interpolation method. This method is performed by calculating linear interpolation first in one direction, and then again in the other direction. Although both steps are linear interpolations, the resulting interpolation as a whole is quadratic in nature.

We used Bilinear Interpolation method for scaling the images. All landscape images were resized to (height, width, channel) = (480, 640, 3) and all portrait images were resized to (height, width, channel) = (640, 480, 3) for ease of calculation.

IV. Color Models

A color space refers to the specific organization of colors. A color model is an abstract mathematical model that describes the representation of colors as tuples of numbers. Colors models enable reproduces representations of color, which result in consistent imaging experience. A color model generally associates itself with an absolute color space via a

mapping function. The specific mapping function between a color model and a reference color space establishes the "gamut" of the color model. Various color models represent color differently. These representations thus can be used for appropriate purposes. A total of four color models have been used in this thesis to research the effectiveness of the feature extraction techniques in different model. They are: RGB, HSI, LAB, and YCbCr color models. Brief descriptions of the color models are represented below:

1. RGB Color Model:

RGB color model is a widely used additive color model. This model was developed with the specific needs of electronic systems in mind. RGB color models represent a magnitude of colors as additions of Red, Green, and Blue lights. In RGB color model, white color is generated by the presence of all colors and black color is generated by the absence of all colors. The RGB color is model is generally represented by a unit cube with one corner located at the origin of a three-dimensional color co-ordinate system. The axes of the cube are labeled R, G, B. The range of the colors are [0, 1]. The origin (0, 0, 0) of the cube is black, and the diagonally opposite corner (1, 1, 1) is white. The line connecting the origin and the diagonally opposite corner represents a gray scale and has equal components of R, G, B. Grayscale images can be generated either by simple averaging or by weighted averaging.

The simple averaging formula for RGB to grayscale conversion is as follows:

$$Gray = \frac{R + G + B}{3} \tag{3.1}$$

The weighted averaging formula for RGB to grayscale conversion is as follows:

$$Gray = 0.299R + 0.587G + 0.114B \tag{3.2}$$

Although there are multiple formulas for calculating the grayscale values from RGB using weighted averaging,

2. HSI Color Model:

Although RGB color model is very useful for hardware representation of images, HSI color models represent images the way humans perceive images. HSI stands for Hue, Saturation, and Intensity. This color model basically separates the intensity component from the color-carrying components. The hue channel describes the similarity of a color to one of the perceived colors: red, yellow, green, and blue according to visual perception. The saturation channel provides a measure of how much a pure color is diluted with white light. The intensity channel basically represents the grey level values of the image. The conversion formula of an image from RGB color model to HSI color model is as follows:

$$I = \frac{R + G + B}{3} \tag{3.3}$$

$$S = 1 - \frac{3}{R + G + B} \times \min(R, G, B) \tag{3.4}$$

$$H = cos^{-1}\left(\frac{(R - G) - (R - B)}{2\sqrt{(R - G)^2 + (R - B)(G - B)}}\right) \quad assuming\ G > B \tag{3.5}$$

$$H = 360 - H \quad assuming\ B > G \tag{3.6}$$

3. LAB Color Model

LAB stands for Luminance, and two chromatic components A and B. The luminance channel represents the lightness from black (0) to white (100). The A component represents the lightness from green (-120) to red (120). The B component represents the lightness from blue (-120) to yellow (120). This color model is device independent. That means, the images are consistent across devices regardless of the hardware specifications. However, conversion between RGB and LAB can be done as a form of matrix multiplication. To convert an image from RGB to LAB color model, the image must be converted to XYZ color space first. The formula for conversion of an image from RGB to XYZ color model is as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [M] \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.7}$$

$$[M] = \begin{bmatrix} 0.412 & 0.358 & 0.180 \\ 0.213 & 0.715 & 0.072 \\ 0.019 & 0.119 & 0.950 \end{bmatrix} \tag{3.8}$$

The image in the XYZ color model can be converted to the LAB color model using the following formula when a reference white (X_r, Y_r, Z_r) is given:

$$L = 116 f_y - 16 \tag{3.9}$$

$$a = 500(f_x - f_y) \tag{3.10}$$

$$b = 200(f_x - f_y) \tag{3.11}$$

Where,

$$f_x = \begin{cases} \sqrt[3]{x_r} & if\ x_r > \epsilon \\ \dfrac{\kappa x_r + 16}{116} & otherwise \end{cases} \tag{3.12}$$

$$f_y = \begin{cases} \sqrt[3]{y_r} & if\ y_r > \epsilon \\ \dfrac{\kappa y_r + 16}{116} & otherwise \end{cases} \tag{3.13}$$

$$f_z = \begin{cases} \sqrt[3]{z_r} & if\ z_r > \epsilon \\ \dfrac{\kappa z_r + 16}{116} & otherwise \end{cases} \tag{3.14}$$

$$x_r = \frac{X}{X_r} \tag{3.15}$$

$$y_r = \frac{Y}{Y_r} \tag{3.16}$$

$$z_r = \frac{Z}{Z_r} \tag{3.17}$$

With, $\varepsilon = 0.008856$ and $\kappa = 903.3$ according to the actual CIE standard.

4. YCbCr Color Model:

YCbCr is a family of color spaces used mostly in color image pipelines in digital photography and video systems. The Y component is the luma component of the image. The Cb component represents the blue-difference chroma components or blue chromaticity. The Cr component represents the red-difference chroma components or red chromaticity. Images can be converted from RGB color model to YCbCr color model using the following formula:

$$Y = 0.257R + 0.504G - 0.098B + 16 \tag{3.18}$$

$$Cb = 0.148R - 0.291G + 0.439B + 128 \tag{3.19}$$

$$Cr = 0.439R - 0.368G - 0.071B + 128 \tag{3.20}$$

B. Feature Extraction

We pursued two feature extraction techniques in this thesis. The first feature extraction feature employed six features namely: contrast, dissimilarity, homogeneity, energy, correlation, and angular second moment. Although there are 14 texture measures, a lot of the measures are heavily correlated with one another and thus, there are truly 4- 6 independent measures that can be used as unique features. [29]

The second feature extraction technique employed Color Histogram along 3 channels, MinPool and MaxPool features (CHPOOL). Brief descriptions of the features are provided below:

I. GLCM Features

Textures in the case of images quantify the Grey Level differences (also known as contrast), directionality, or lack of it (also known as omnidirectional), defined area size where change happens (window size defining a neighborhood). These features are quantified using texture measures. Textures are widely used features in image processing due to their expressive and distinguishing capability. Image textures are mostly used in image classification and segmentation.

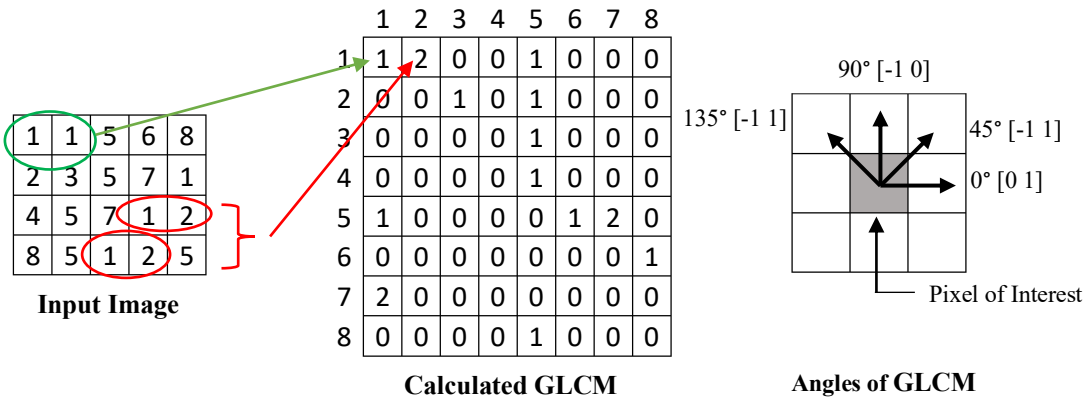


Figure 3.2: Calculating GLCM from a sample 3-bit image.

Haralick et al. mostly systematized the GLCM texture calculations used in modern technologies including remote sensing, satellite imaging, aerial image segmentation, etc. [30]. Mathematical pattern analysis, also known as spatial pattern analysis is the origin of Haralick et al.'s work.

14 different texture measures were proposed by Haralick et al. in 1973 [30]. Other than the measures described below, the proposed measures also included variance of the difference between neighboring pixels, variance based on the sum of neighboring pixels, correlation involving entropies, the maximum correlation coefficient, and entropy of the sum, and entropy of the difference between neighboring pixels. However, the features that were not described below, were not generally adopted. In the original paper, the 14 measures were calculated for the 4 directions and the mean and range of each measure were also used as feature vectors in classification. Grey Level Co-occurrence Matrix (GLCM) is one of the most prominently used texture measures.

GLCM is a tabulation of the frequency of the occurrence of different combinations of pixel brightness values in an image. For example, for an input image with a bit depth of 3, the resultant GLCM is a matrix of size 8 × 8. The cell with the co-ordinates (1,1) contain the number of times two consecutive pixels in the input image had pixel intensity of 1 and 1. Similarly, the cell with the co-ordinates (1,2) contains the number of times two consecutive pixels in the input image had pixel intensity of 1 and 2. GLCM can be calculated in varying angles such as 0°, 45°, 90°, 135°, etc. Figure 3.2 illustrates a sample GLCM tabulation process and the possible angles of GLCM calculation.

Properties of the GLCM:

1. It is square.

2. It has the same number of rows and columns as the quantization level of the image.
3. Normalized GLCM is symmetric around the diagonal.

Normalized GLCM is calculated by following the steps:

1. The transpose of the matrix along one direction is made.
2. The transpose matrix is added to the first matrix.
3. Then the probability of each outcome in the GLCM is calculated using

$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=0}^{N-1} V_{i,j}} \tag{3.21}$$

where I is the row number and j is the column number. V represents the value in the cell i, j in image window, $P_{i,j}$ is the probability value recorded for the cell i, j, and N is the number of rows or columns.

Texture measures are the various single values used to summarize the normalized symmetrical GLCM in helpful ways. Most texture measure calculations are weighted averages of the normalized GLCM cell contents.

1. **Contrast Group**: Weights in respect to the distance from GLCM diagonal are used to calculate measures related to contrast.
   a. **Contrast**: Contrast is also known as "sum of squares variance" and occasiona1lly “inertia”.

$$\sum_{i,j=0}^{N-1} P_{i,j}(i-j)^2 \tag{3.22}$$

   b. **Dissimilarity**: The contrast weights increase exponentially, whereas the dissimilarity weights increase linearly.

$$\sum_{i,j=0}^{N-1} P_{i,j}|i-j| \tag{3.13}$$

   c. **Homogeneity**: While more windows show more contrast among them, the dissimilarity and contrast values are larger. If in the selected window, the weights decrease away from the diagonal, the windows with little contrast will result in

larger texture measures. Homogeneity is inversely related to Contrast weight. Thus, weights decrease exponentially away from the diagonal.

$$\sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1+(i-j)^2} \tag{3.24}$$

2. **Orderliness Group**: Orderliness indicates how regular or orderly the differences among the pixel values in the selected image window are.
   a. **Angular Second Moment (ASM)**: When the image window is very orderly, high values of ASM or energy occur.

$$\sum_{i,j=0}^{N-1} {P_{i,j}}^2 \tag{3.25}$$

   b. **Entropy**: Entropy is the measure of disorderliness in variables. Entropy is calculated using the following formula:

$$\sum_{i,j=0}^{N-1} P_{i,j}(-\ln P_{i,j}) \tag{3.26}$$

3. **Descriptive Statistics**: The third group of texture measures are similar to common descriptive statistics, namely, mean, standard deviation, variance, etc. However, while common descriptive statistics are calculated based on the image intensities, texture based descriptive statistics are calculated based on the calculated GLCM metrics.
   a. **Mean**: The GLCM mean is very different from geometric or arithmetic mean of the image window. The value is calculated with respect to the GLCM matrix. The pixel values are not weighted based on its frequency of occurrence, but by its frequency of the occurrence in combination with a certain neighboring pixel intensity.

$$\mu_i = \sum_{i,j=0}^{N-1} i(P_{i,j}) \tag{3.27}$$

$$\mu_j = \sum_{i,j=0}^{N-1} j(P_{i,j}) \tag{3.28}$$

b. **Variance**: Variance is the measurement of the dispersion of the values in regards to the mean. It is similar to dissimilarity or contrast.

$$\sigma_i{}^2 = \sum_{i,j=0}^{N-1} P_{i,j}(i - \mu_i)^2 \tag{3.29}$$

$$\sigma_j{}^2 = \sum_{i,j=0}^{N-1} P_{i,j}(j - \mu_j)^2 \tag{3.30}$$

c. **Correlation**: Correlation is the measurement of the linear dependency of grey values of neighboring pixels.

$$\sum_{i,j=0}^{N-1} P_{i,j} \left[\frac{(i - \mu_i)(j - \mu_j)}{\sqrt{(\sigma_i{}^2)(\sigma_j{}^2)}} \right] \tag{3.31}$$

II. Color Histogram Feature with MinPool and MaxPool Features (CHPOOL)

In image processing, histograms represent the frequency of pixel intensities in an image. Color histogram represents the distribution of colors in an image. The "bit depth" property of images constitute the range of color the image contains. For example, an image with 8-bit depth has a range of [0, 255]. Color histograms calculate the number or frequency of the pixels for each intensity in the provided range. Color histograms can be built for any kind of color space. An Image in RGB color space has three channels: The Red, Green, and Blue layers. Individual color histograms can be built for each channel. Similarly, a single color or intensity histogram can be built for greyscale images. Color histograms can be represented in 1D arrays whose length is equal to 2^N, where N corresponds to the bit depth.

1. MinPool

Pooling is a general technique used to reduce the dimensionality, computational complexity and variance. MinPool can be expanded as Minimum Pooling. As the name suggests, in

minimum pooling, the minimum pixel value from a batch is selected. This batch is generally generated by a sliding window. In the sliding window approach, in case of minimum pooling, the minimum pixel value among all the pixel intensities in the sliding window crop is selected. MinPool can be used to highlight the darker regions in a window.

If a window is defined as $W = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ then the MinPool is defined as follows:

$$MinPool = \min([a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{32}, a_{33}]) \tag{3.32}$$

2. MaxPool

MaxPool can be expanded as maximum pooling. While maximum pooling, the maximum value in a batch is selected. In case of a sliding window approach, the maximum pixel intensity value among all the pixel intensity values in the window is selected. MaxPool can be used to highlight the brighter regions in a window.

If a window is defined as $W = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ then the MinPool is defined as follows:

$$MaxPool = \max([a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{32}, a_{33}]) \tag{3.33}$$

C. Model Training

Various machine learning algorithms such as Random Forest, Decision Tree, Support Vector Machine (SVM), Stochastic Gradient Descent Classifier (SGDc) were used on the extracted features. Among them, Random Forest worked best. The used data was scaled using Standard Scaling techniques.

Standard scaling is a common requirement for a lot of machine learning models. Standard scaling is used to standardize data by removing the mean and scaling the data to unit variance. The standard scale of a sample D is calculated as:

$$ss = \frac{D - \mu}{\sigma} \tag{3.34}$$

Here μ is the mean of the dataset and $\boldsymbol{\sigma}$ is the standard deviation of the training samples.

I. Random Forest [31, 32, 33]

Random forest is ensemble learning methods used extensively for various tasks such as: classification, regression, etc. Random forests are generated by constructing numerous decision trees each working with a random subsample of the entire dataset. The final output is generated by taking the mode of all the classes of the individual trees in case of classification.

Decision trees are very commonly used predictive models that predict the value of a target based on several input variables. The tree is built by splitting the source set consisting of the root node of the tree into subsets consisting of the successor children. The splitting is based on various splitting rules such as: Gini impurity, Information gain, etc. This process is repeated on each derived subset recursively until a given depth or until leaf nodes are found.

For a dataset containing J classes, Gini impurity is calculated as follows:

$$I_G(p) = 1 - \sum_{i=1}^{J} {P_i}^2 \tag{3.35}$$

Where $i \in \{1, 2, \dots, J\}$ and P_i is the fraction of items labeled with class $\boldsymbol{i}$ in the dataset.

Information gain is based on entropy. Entropy is defined as follows:

$$H(T) = I_E(p_1, p_2, \dots, p_j) = -\sum_{i=1}^{J} p_i log_2 p_i \tag{3.36}$$

Here $p_1 + p_2 + \dots + p_j = 1$. Information gain can be represented as follows:

$$\begin{aligned} IG(T, a) &= H(T) - H(T|a) \\ &= -\sum_{i=1}^{J} p_i log_2 p_i - \sum_{a} p(a) \sum_{i=1}^{J} -\Pr(i|a) log_2 \Pr(i|a) \end{aligned} \tag{3.37}$$

Information gain is thusly used to decide which feature to use as a basis for splitting. The best feature is the one that provides the most information gain.

Decision trees tend to overfit to training data. It is evident from the algorithm that, if no limit is set on the depth, the tree will find route for all the items in the dataset. Random forest overcomes this by providing randomly sampled dataset to each decision tree. Random forests can further battle the overfitting problem by limiting the maximum number of trees used or the depth of the trees.

There has been extensive research on modifications of Random Forests for exploiting its simplistic yet powerful ability in other classification tasks. Various other similar classifiers exist such as: Extremely Randomized Trees Classifier (Extra Trees Classifier), Isolation Forest, Naïve Bayes classifier, etc. These classifiers can be used for specialized tasks. For examples, isolation forests are capable of detecting anomalies in unsupervised feature space. Extra Trees Classifiers are very similar to random forests; however, they are different in two main ways: all trees are trained with the whole dataset instead of bootstrap data and the splitting is done randomly instead of calculating entropy or Gini index.

II. Logistic Regression **[34, 35, 36]**

Logistic Regression is a statistical model that uses a logistic function to model binary dependent variables in its simplest form**.** Logistic regression is very similar to linear regression, however the main difference between them is the binomial response variable used in the logistic regression. A logistic regression will model the chance of an outcome based on individual characteristics. As probabilities can be represented as ratios, logistic regression models the following:

$$\log\left(\frac{\pi}{1-\pi}\right) = B_0 + B_1 x_1 + B_2 x_2 + \cdots + B_m x_m \tag{3.38}$$

Here, π is the probability of an event, B_i indicates the regression co-efficient related to the reference group, and x_i is the explanatory variable.

Logistic regression has many advantages over other classifiers. Logistic regression models are less inclined to overfit in low-dimensional datasets. In higher-dimensional datasets, regularization techniques such as L1 and L2 can be used to reduce overfitting. Logistic regression models also perform very well and provide good accuracy when the classes in the dataset are linearly separable. Logistic regression also provides a natural probabilistic view of class predictions and is considerably computationally less demanding.

III. Stochastic Gradient Descent Classifier [37, 38, 39]

Stochastic Gradient Descent Classifier (SGD-C) is a linear classifier that uses the Stochastic Gradient Descent method as an optimizer. Gradient Descent (GD) functions are generally used to optimize a cost function. While other GD function such as Batch GD or Mini-batch GD use the entire dataset or subsets of the dataset to calculate the minima, SGD uses single samples. Stochastic gradient descent calculates the following:

$$x^{(k)} = x^{(k-1)} - t_k . \nabla f_{i_k}(x^{(k-1)}), \text{ k} = 1, 2, 3, \ldots. \tag{3.39}$$

Where $i_k \in \{1, \ldots, m\}$ is a chosen index at iteration k when minimizing the average of function:

$$\min_x \frac{1}{m} \sum_{i=2}^{m} f_i(x) \tag{3.40}$$

One of the main advantages of using SGD-C over other linear classifiers such as Linear Regression, Support Vector Machine (SVM), Logistic Regression is that SGD-C reduces the training time significantly for larger datasets. For example, the cost function of Logistic Regression cannot be calculated directly. However, the SGD function reaches closer to the minima with each training sample. Thus, SGD-C is considerably faster in finding the minima than other methods. Another major advantage is that, SGD-C requires less memory (RAM) than other linear classifier methods due to it only using a single sample at a time.

D. License Plate Localization

In the detection phase, the standard scaler and the trained model is loaded. The image is also loaded. The sliding window with a set window and step size slides along the entire image and extracts the features from the cropped images in the windows. The extracted features are first scaled and then sent to the classifier for proper classification. If the classifier classifies the instance as a license plate position, a bounding box with the same size as the windows is drawn in that position and its co-ordinates are registered. If the classifier classifies the instance as non-license plate region, then the sliding windows slides to the next window position by adding the step size. When adding window size with the current window position results in a co-ordinate that is out of the bound or range of the image size, then the sliding window process is terminated and the merging process of probable parts of

license plates begin. The overlapping bounding boxes are concatenated to form a larger bounding box. Multiple license plate regions can be generated in a single image. However, license plates follow special regulations the dictate the shape and area of the license plate. The area and the aspect ratio of the bounding boxes of probable regions are calculated. Areas that have greater aspect ratio than 3.0 or lesser aspect ratio than 1.0 are discarded. The largest area is then selected, bounding box is generated and output as license plate region. Figure 3.3 presents the flowchart of the detection and localization process.

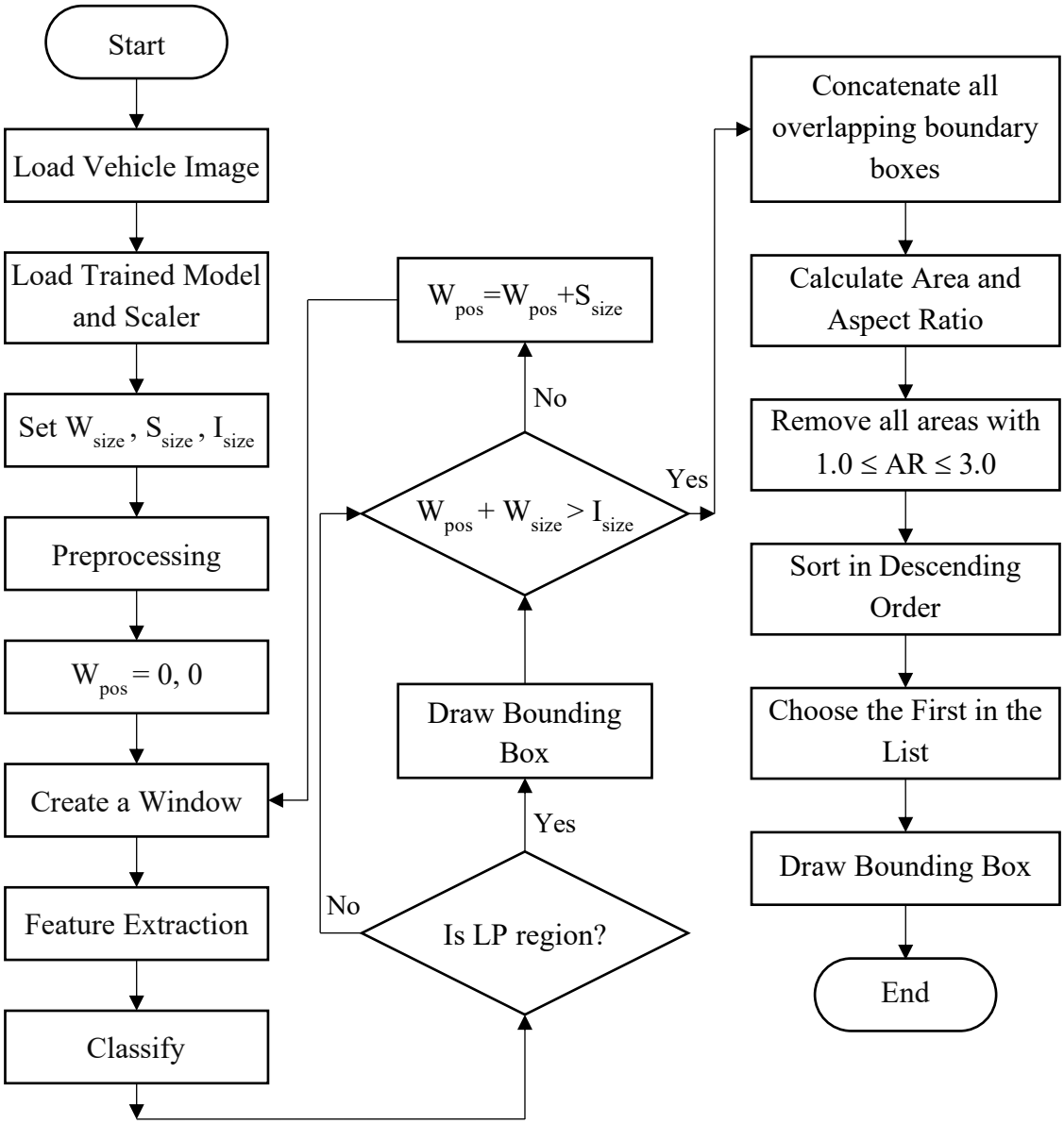


Figure 3.3: Flowchart of the detection and localization process.

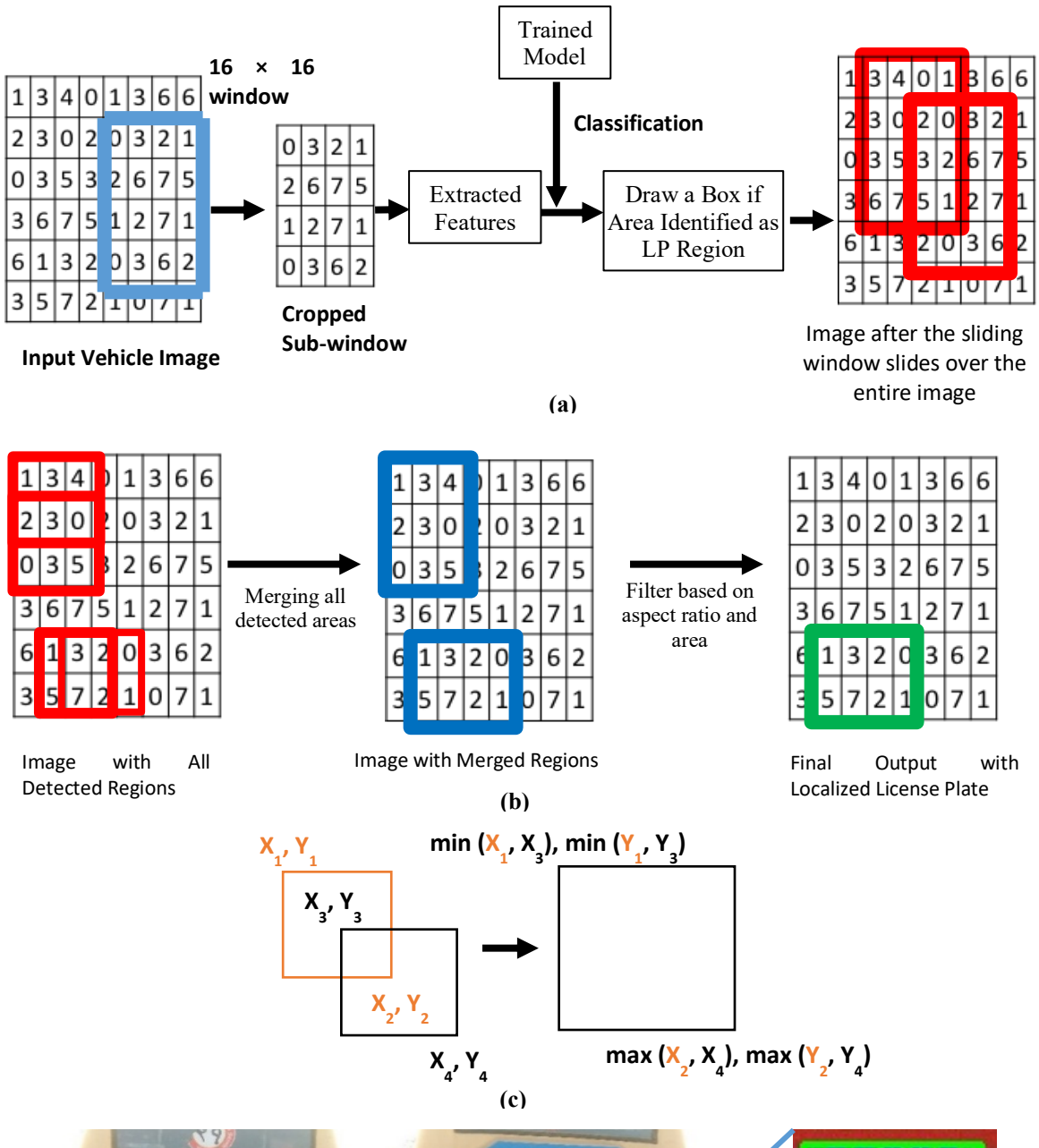


Figure 3.4: An illustration of the detection and localization process.

Figure 3.4 (a) and (b) illustrates are visual representations of the localization process. In the input image, the area under the blue rectangle is the current position of the sliding windows. This region is cropped and the features are extracted. The extracted features are then sent to the classifier for proper classification. If the classifier classifies the cropped region as a part of a license plate, the co-ordinates of the current window is saved. The areas under the red rectangle are the probable parts of a license plate region. Multiple such overlapping probable parts of license plates can exist as seen from the initial image in Figure 3.4 (b). The formula presented in Figure 3.4 (c) is then used to merge the probable parts of license plates and general probable full license plate regions. If the coordinates of the top left corner and bottom right corner are known for each of the two overlapping bounding boxes, coordinates for a larger bounding box can be calculated that contains the object detected by the initial small bounding boxes. Probable LP regions are generated by merging the red bounding boxes. Probable license plate regions are presented by blue bounding boxes. After filtering the probable LP regions based on the Aspect Ratio and area, the final license plate region is selected. The final license plate region is represented by the green bounding box. In Figure 3.4 (d), the left image is the input image. In the zoomed inset, the blue bounded regions are the probable license plate parts. The green bounding box is generated by using the formula in Figure 3.4 (c) to merge the probable parts.

3.2.2 License Plate Recognition

In this recognition phase at first, we need to segment the characters, then the characters are recognized by using classification algorithm. In the segmentation process, at first the image of license plate has been resized into 200 × 400 pixel. Then we have converted the RGB image into gray scale. Then we have checked if the license plate is rotated or not. To check this, we have detected the Canny edge. Then the Hough line detection has been used to detect the border line of the license plate. If the license plate is rotated, then it need to be straight. Then the noise is removed using Gaussian blur. Adaptive threshold has been used to make the image binary. As the Bangladeshi license plate contains two lines, we need the line segmentation to segment the character portion and the number portion. Horizontal histogram projection has been used to segment the line. After line segmentation morphological operation has been done to remove small noise and grow the size of the characters. This helps to ease the next step of segmentation and to fill small gaps within character boundaries.

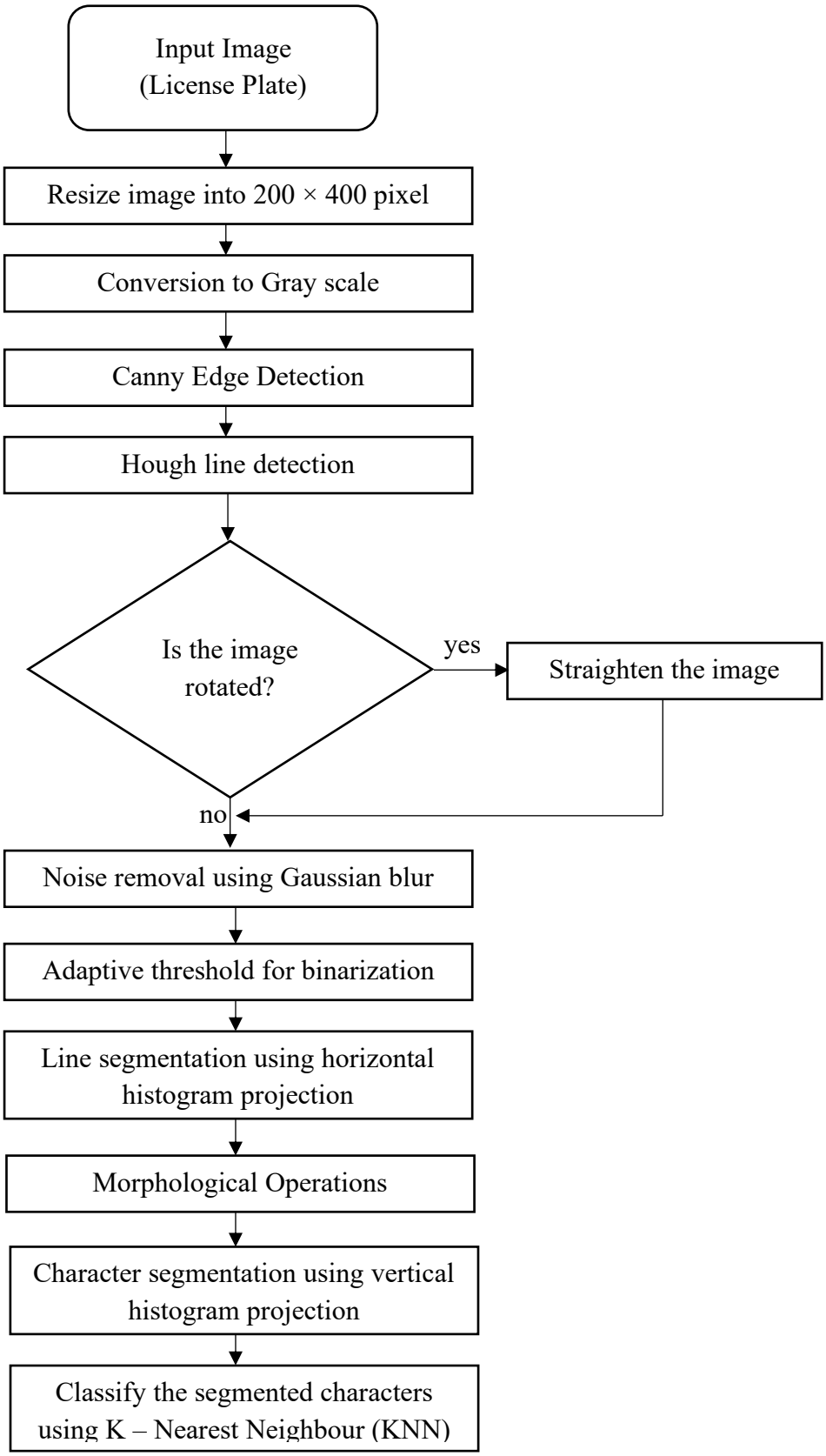


Figure 3.5: Flowchart of the Character Segmentation and Recognition Process.

(a) 200 × 400 resized image (b) Greyscale image (c) Canny edge detection

Figure 3.6: Canny edge detection output from a sample input LP image.

Then the characters have been segmented by using vertical histogram projection. Then the segmented characters have been classified by using k – nearest neighbor (KNN). Figure 3.5 depicts the steps of character segmentation and recognition process. The following steps elaborate the working procedure behind the initial license plate character segmentation using horizontal and vertical histogram projections and the subsequent character recognition using KNN algorithm.

A . Preprocessing

To get better result for segmentation and recognition pre-processing is very essential. At first the original image of license plate is converted into a height of 200 pixels and width of 400 pixels. The processing of 24 bit RGB image is time consuming. So the RGB image is converted into gray scale. Figure 3.6(a) shows the 200 × 400 resized license plate image and Figure 3.6(b) shows the image of license plate after gray scale conversion.

I . Canny Edge Detection [40]:

The edge detection of an image decreases redundant data, preserves the necessary structural properties and filters out the useless information of the image. In our proposed method, we have used Canny edge detection which is widely used in image processing tasks to detect a wide range of edges in noisy images. In our proposed method, the minimum value and maximum value of Canny edge detection is 50 and 150 accordingly and the aperture size of Sobel kernel is 3. Figure 3.6(c) represents the Canny edge detection from the gray scale image.

II. Hough Line Detection [41]:

Some images of license plate are slightly rotated. So, we need to make them straight before segmentation. At first, we need to find out in which angle they are rotated. To find the angle, we need to detect the border line of the license plate.

(a) Detected Hough lines (b) Rotation of License Plate (c) Main Portion of License Plate

Figure 3.7: Extracting the main portion of License Plate.

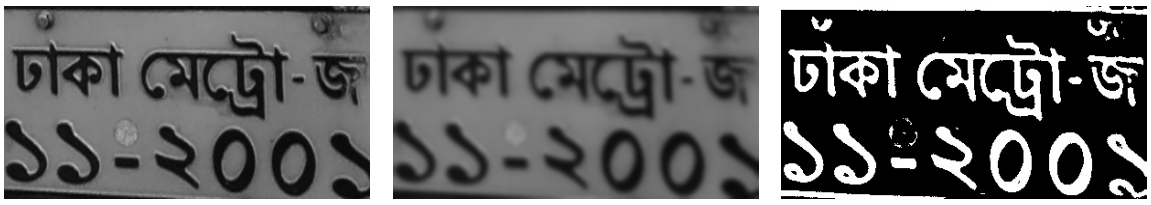
To detect the line, we have used Hough line detection. The Hough Line Transform detects the straight lines. To apply the Transform, first an edge detection pre-processing is desirable and it is already done by Canny edge detection described before. After detecting the Hough line, the angle in which the license plate is rotated is found. Then the license plate is again rotated in this angle to make it straight. Then the main portion of the plate is extracted. Figure 3.7(a) shows the detected Hough lines, figure 3.7(b) shows the license plate after rotation and Figure 3.7(c) shows the extracted main portion of the license plate.

III. Gaussian Blur:

The Gaussian blur is a type of image-blurring filter that uses a Gaussian function for calculating the transformation to apply to each pixel in the image. In our proposed method, we have used a 15 × 15 Gaussian kernel to blur the image, where $\boldsymbol{\sigma} = 2.6$. Figure 3.8(b) shows the blurred image after applying the 15 × 15 Gaussian filter on a sample greyscale image.

IV. Adaptive Threshold:

Thresholding is a useful technique used to segment the foreground and background parts of images. This is done by setting the values of the pixels surpassing a threshold value to a foreground value and the remaining pixels to a background value. While general thresholding methods use a single threshold value for the entire image, adaptive thresholding uses dynamic thresholds dependent on individual areas of images. This helps in cases where lighting or reflection is uneven. Adaptive thresholding can adapt to different lighting conditions. As a license plate has differences in lighting in different areas, Adaptive thresholding gives better results. Adaptive thresholding takes a greyscale image as input and outputs a binary segmented image. This method calculates a threshold value for each pixel in the image.



(a) Original image (b) Blurred image (c) Threshold image

Figure 3.8: Blurred image using Gaussian blur and binary output image after adaptive thresholding.

Similar to the general thresholding method, if a pixel intensity value is less than the threshold value, it is set to background value, otherwise it is set to foreground value.

There are many ways to calculate the threshold in adaptive method. Two of them are:

- Threshold value is set as the mean of the pixel intensities of the neighborhood area.
- Threshold value is set as weighted sum of pixel intensity values of neighborhood pixels where the weights are based on a Gaussian window.

In our proposed method, the Gaussian of 11 × 11 neighborhood has been used to apply the adaptive threshold. In this case, $\boldsymbol{\sigma} = 2$. The output is shown in figure 3.8(c).

B. Line Segmentation

As we described before that, the Bangladeshi number plate contains two lines in general. The upper line contains the characters and the lower line contains the numbers. At first, we need to segment the character portion and the number portion. So, for this line segmentation, we have used horizontal histogram projection to determine where we should segment.

I. Horizontal Histogram Projection:

After converting the color image into binary image there are only black and white pixels in the image. The white pixel indicates the foreground pixels and black indicates the background.

A histogram represents the frequency of pixel intensities of an image. In the recognition process, horizontal histograms have been used for separating the two major horizontal portions from the license plates. The first horizontal portion consists of characters whereas the second horizontal portion contains the numbers.

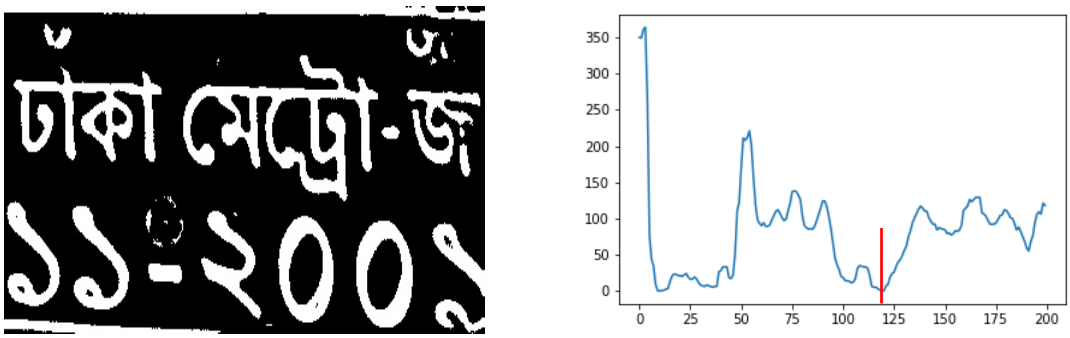


Figure 3.9: Horizontal histogram projection of binary image.

(a) Character portion (b) Number portion

Figure 3.10: The segmented parts after line segmentation.

Horizontal histogram of a specific pixel intensity for an image presents the frequency of the specified pixel in each row of the image. In the binary image of license plate, rows which contain space (Background pixels) between lines have lower peaks and rows which contain text (Foreground pixels) has higher peaks in the corresponding histogram plot. Thus, a valley between two massive peak groups can be attributed to the blank portion between horizontal lines. Figure 3.9 shows the horizontal histogram projection for white pixels of the binary image. Now we select all the rows which have lower peaks in the locality for line segmentation to separate out the line between the character portion and number portion. The red separator line shows the segmentation points. Figure 3.10 shows the segmented parts of the license plate after line segmentation.

C. Character Segmentation

We have already segmented the whole image into two portion using horizontal histogram projection. Now we have to segment characters from the segmented images containing a single line and sequence of characters. Vertical histogram projection has been used to segment the characters.

I. Morphological Operation [42]:

Morphological operations are primarily used to remove the imperfections affecting the shapes, noise, and textures of images.

(a) Eroded Image with 2 × 2 structuring element

(b) Dilated Image with 4 × 4 structuring element

Figure 3.11: Images and their output after erosion and dilation.

Dilation and Erosion are the most common morphological operations. The dilation operation is used to increase size of objects inside images and to fill the holes (missing pixels) in a continuous object. The erosion operation is used to decrease the size of objects inside images and to remove noisy connections among objects. Morphological operations are done on the segmented parts of the license plate to remove the unnecessary things and for better result for character segmentation. A 2 × 2 structuring element has been used for erosion to remove the small noises. Figure 3.11(a) shows the outputs for erosion. Then a 4 × 4 structuring element has been used for dilation for filling the small holes and growing the shape of the characters. Figure 3.11(b) shows the output for dilaton.

II. Vertical Histogram Projection:

In the vertical histogram projection method, the number of foreground pixels (white pixels) along the columns of an image are counted and the length of the resultant histogram array is equal to the width of the image. Figure 3.12 shows the vertical histogram projection of the character portion and the number portion of the license plate. In the resultant histogram, the columns which are part of characters have higher peaks and columns which are part of the space between the characters have lower peaks. Thus, the columns that contain lower peaks corresponding to spaces between characters are selected as potential segmentation points.

However, license plates contain a lot of noise such as random vertical scratches or discolorations. Thus, a minimum threshold for the segment width is necessary to ensure proper character segmentations. Figure 3.13(a) shows the segmented characters and Figure 3.13(b) shows the segmented numbers from the license plate.

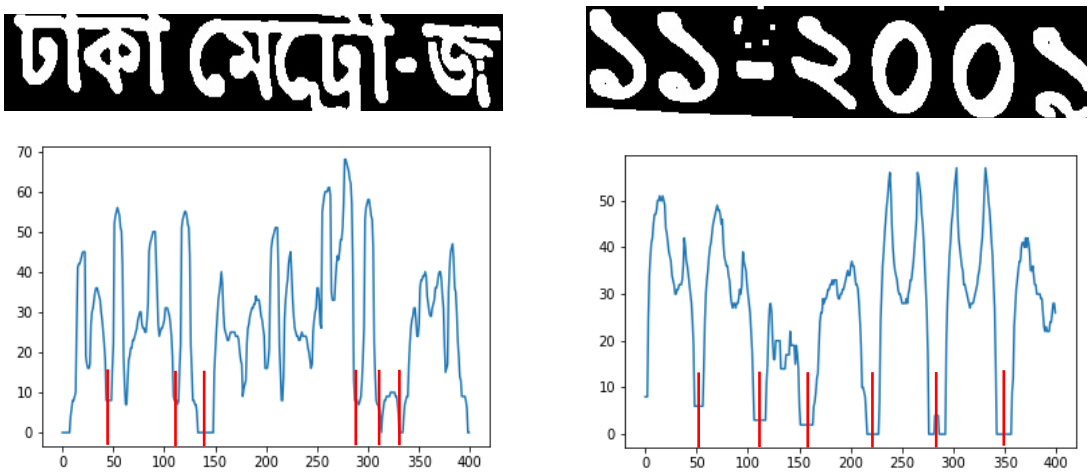


Figure 3.12: Vertical histogram projection of the segmented images.

(a) Segmented Characters

(b) Segmented Numbers

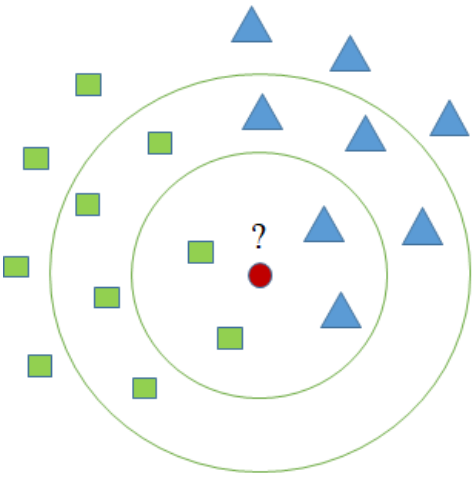
Figure 3.13: Segmented Characters and Numbers.

D. Character Recognition

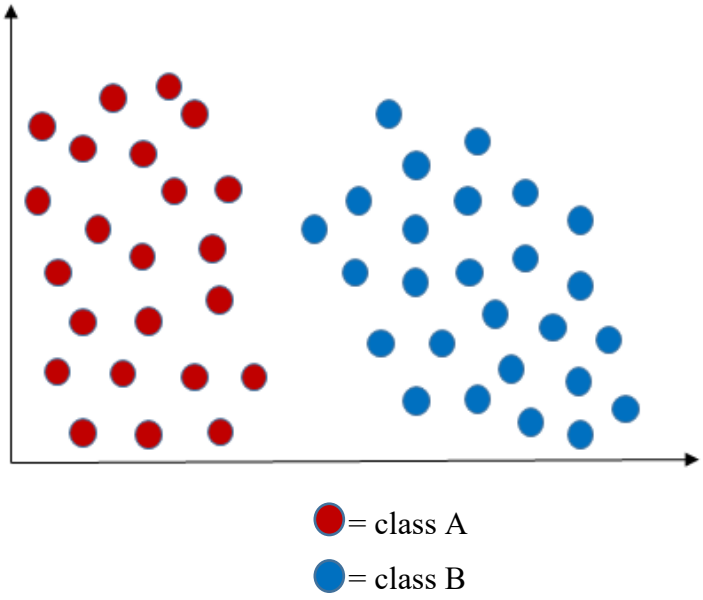
In our proposed method, we have used k - nearest neighbor algorithm to classify the characters. The model achieved high levels of performance.

I. K – Nearest Neighbor Algorithm (K-NN) [43, 44]:

The k - nearest neighbor algorithm (K-NN) is a classification technique which classifies objects by calculating distance measures from its neighbors. K-NN is widely used due to its lesser computational cost, simplicity, and high effectivity. Objects are classified based on plurality votes by its neighbors, represented by k. The output can generally be interpreted as class membership. K-NN is sensitive to local structures of data. Figure 3.14(a) shows the K-NN classification diagram. While training a K-NN classifier on image data, at first, multiple images are from each class are brought to the training stage to be stored.



(a): K-NN classification diagram



(b): Feature Space

Figure 3.14: KNN classification diagram and feature space.

This storage is basically the feature space. In the feature space, every object is presented as a numerical vector. Thus, the feature vector itself is multidimensional vector. At first, the features are stored in separate co-ordinates based on their features. At first, the features are scattered across the feature space. However, the features that are similar to some extent are grouped or clustered together. While testing images, they are compared to the training images using the K-NN algorithm. The test image data is firstly moved to the feature space and its position in the feature space is determined based on its features. After acquiring the position in the feature space, the neighbors related to it are determined and comparison occurs. The test image is then classified as part of the neighboring class with highest votes.

Figure 3.15: Some samples from training dataset.

The test image is then classified as the neighboring class with the highest amount of votes. Figure 3.14(b) shows a sample feature space. In our proposed method, 425 images for characters and 168 images for numbers have been used as training dataset. Every image is resized into 20 × 30 pixels and the intensity values of 600 pixels have been used as the feature vector. Figure 3.15 shows some samples from the training dataset.

The segmented characters of the test image are then resized into 20 × 30 pixels and label is given for each particular class. Then the minimum distance is estimated between the feature vector of the test image and the vectors stored in the library. In our experiment, the minimum distance has been measured by using different kind of distance: Euclidean distance, Squared Euclidean distance, Manhattan distance, Cosine Similarity, Bhattacharyya distance etc. Then the segmented characters are classified into appropriate class according to the minimum distance and number of votes. As the model is simple and lightweight, it is computationally faster.

3.3 Conclusion

In this chapter, the methodology behind the entire proposed automatic license plate detection and recognition system has been described. The license plate detection system depends on feature extraction and machine learning algorithms. The recognition part uses histograms for character segmentation and KNN algorithm for character recognition.

CHAPTER IV

Experimental Results

4.1 Introduction

In this chapter, the results from all the methods that were pursued along the course of this thesis work is represented. The chapter is divided into multiple sections and subsections. The rest of the chapter is organized as follows: Section 4.2 and subsequent subsections 4.2.1 and 4.2.2 present useful information about the used hardware, and the software, respectively, for future reproducibility. Section 4.3 provides a detailed overview of the used dataset. Section 4.4 and subsequent subsections 4.4.1 and 4.4.2 represent the various Evaluation metrics used in the model training phase and the detection phase, respectively. Section 4.5 illustrates the input sample images from each type in the dataset, and their respective outputs. Section 4.6 and subsequent subsections 4.6.1, 4.6.2, 4.6.3, 4.6.4, and 4.6.5 present information on the performance of the two main experimental feature extraction methods: Grey Level Co-occurance Matrix (GLCM) on RGB color channel, Grey Level Co-occurrence Matrix (GLCM) on HSI color channel, Color Histogram with MinPool and MaxPool on RGB color channel, Color Histogram with MinPool and MaxPool on LAB color channel, and Color Histogram with MinPool and MaxPool on YCbCr color channel, respectively. Section 4.7 concludes this chapter.

4.2 Experimental Setup

In this section, useful information about the hardware used in the capturing and model training process as well as the software information regarding the toolkit versions, OS information, etc. are discussed.

4.2.1 Hardware

In this subsection, useful information about the hardware used in the database creation and later processing and detection tasks are provided.

- **Image Capturing Devices**
  - Xiaomi Redmi Note 4
    - Sensor: 13 MP, Sony Exmor RS.
    - Aperture: f/2.0.
    - Pixel Size: 1.12µm.
    - Flash: CMOS BSI with dual-LED (dual tone).
  - Xiaomi Redmi 5A
    - Sensor: 13 MP, Sony Exmor RS.
    - Aperture: f/2.2.
    - Pixel Size: 1.12µm.
    - Flash: LED flash.
  - Xiaomi Redmi Note 8 Pro
    - Sensor: 64 MP, Samsung ISOCELL Bright GW1.
    - Aperture: f/1.8.
    - Pixel Size: 0.8µm.
    - Flash: Dual-LED (dual tone).
- **Personal Computer**
  - Processor: 2.8GHz dual-core Intel Core i7 (Turbo Boost up to 3.3GHz) with 4MB shared L3 cache.
  - RAM: 16GB 1600MHz DDR3L.
  - ROM: 1TB APPLE SSD SM1024F.
  - GPU: Integrated Intel Iris 5100 with 1536 MB VRAM.

4.2.2 Software

In this subsection, useful information regarding the used OS information, programming language information, toolkit versions, etc. are provided.

- OS: Mac OS Sierra v10.12.6.
- Programming Language: Python 3.7.
- Application type: Python Script.
- IDE: Spyder v3.3.0
- Libraries: Scikit-learn v0.20.0, Scikit-image v0.15.0, OpenCV v3.4.2, Pandas v0.23.4, Numpy v1.15.3, Seaborn v0.9.0, jupyter notebook v5.6.0.

4.3 Dataset

The goal of our thesis is “Automatic License Plate Detection and Recognition”. Due to the lack of open-source datasets of images of Bangladeshi vehicles with license plates, we created a dataset. All the images of the dataset were captured with express permissions of the owners of the vehicles. Smartphones were mostly used for capturing the devices. More information on the capturing devices can be found on subsection 4.2.1.

A total of 630 images were captured. We have also used the previously created private car dataset. The private car dataset contains 1043 images captured from the roadside, shopping mall, houses, beside market, parking lots etc. and in different lighting conditions [22]. For ease of processing, the captured images were divided among 6 types:

1. Back.
2. BackLeft.
3. BackRight.
4. Front.
5. FrontLeft.
6. FrontRight.

Figure 4.1 presents sample images from each category in the dataset. A standard (70%-30%) train-test split was performed on the dataset for future model training and evaluation purposes. The data were cleaned to remove images containing multiple vehicles, or wrong vehicle types. Each sample in the dataset contains only one license plate. The images in the dataset have diverse backgrounds. The frequency distribution of the images in each type after the split is shown in Table 4.1.

4.4 Evaluation Metrics

For measuring accuracy or to find that our detection and recognition rate we use some metrics like IOU, Precision, Recall. Miss Rate etc. For finding the accuracy we have used two types of approaches one Localization Based and other is Detection based.

For evaluating the results of our detection output, we have used two major type of approaches: Localization Based and Detection based. Some general statistical measure such as Accuracy, Precision, Recall, F1 Score was used to evaluate the detection performance as well as the model performance.

(a) Back

(b) BackLeft

(c) BackRight

(d) Front

(e) FrontLeft

(f) FrontRight

(4.1-1) Images from the commercial vehicle dataset.

(4.1-2) Images from the private vehicle dataset.

Figure 4.1: Sample image for each type in the dataset.

4.4.1 Localization Based Approaches

In these kinds of approaches, at first ground truths are generated. Ground truths are areas detected with the help of a supervisor. The outputs of the machines are then compared with the generated ground truths and based on a set Intersection over Union threshold, it is determined whether successful detection has occurred.

Table 4.1: Number of the images in each type after train-test split.

Type	Number of Training Images	Number of Testing Images	Total Number of Images
Back	114	48	162
BackLeft	66	30	96
BackRight	43	18	61
Front	113	50	163
FrontLeft	57	24	81
FrontRight	48	19	67
Total	441	189	630

IoU: Intersection over Union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset.

It is the ratio between the intersection of Ground Truth and Machine predicted value to union of Ground Truth and Machine predicted value (illustrated in figure 4.1(a)).

Precision: It is the ratio between the intersection of Ground Truth and Machine predicted value to Machine predicted value.

Recall: It is the ratio between the intersection of Ground Truth and Machine predicted value to Ground Truth value.

4.4.2 Detection Based Approaches

Detection based approaches uses four ideas: TP, FP, TN, FN to calculate statistical measures such as accuracy, recall, precision, and F1 score. These can also be used to evaluate performance of a trained machine learning model.

True Positives (TP): True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True).

Ex: The case where the system correctly detects a license plate region in an image.

True Negatives (TN): True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False).

Table 4.2. A generalized view of a confusion matrix.

Actual / Prediction	**Positive**	**Negative**
Positive	True Positive	False Positive
Negative	False Negative	True Negative

Ex: The case where the system detects no number plates, and there was no number plate in the input image also.

False Positives (FP): False positives are the cases when the actual class of the data point was 0(False) and the predicted is 1(True).

Ex: The case where the system detects the number plate in an incorrect region.

False Negatives (FN): False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False).

Ex: The case where the system incorrectly concludes that there are no number plates in an image whereas there was one.

The representation of these 4 ideas are known as a confusion matrix. Table 4.2 depicts a generalization of the confusion matrix:

Accuracy: Accuracy is a measure that tells how much accurate the result is. It is expressed in:

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}} \tag{4.1}$$

Precision: Precision gives a measure that tells how much data objects are correctly and positively classified out of the all positively predicted data objects. It is expressed as:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{4.2}$$

Recall: Recall gives a measure that tells how much data objects are correctly and positively classified out of the all actually positive data objects. It is expressed as:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{4.3}$$

F-measure: The F measure (F1 score or F score) is a measure of a test's accuracy and is defined as the weighted harmonic mean of the precision and recall of the test.

$$F_1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

4.5 Results

The results from all the experimented methods namely the license plate detection and recognition system are presented in this section.

4.5.1 Experimental Results of License Plate Detection

Two separate feature extraction methods were pursued along the course of the thesis. One is GLCM based features and the other is Color Histogram with MinPool and MaxPool based features. The GLCM based feature extraction was performed on two color spaces: RGB and HSI. The Color Histogram with MinPool and MaxPool features were performed in three color spaces: RGB, LAB, and YCbCr. As Color Histogram with MinPool and MaxPool performed better than GLCM based feature extraction techniques, it was used on the private cars dataset to generate results in RGB color space. While features were generated using GLCM, the feature vector consisted of 72 features. As GLCM features were calculated for all the channels, each channel generated 24 features each. A total of four difference angles (0, 45, 135, 180) were used to calculate the GLCM features. The angles generated 6 features each. While the Color Histogram with MinPool and MaxPool features were used, the feature vector consisted of 984 features. As histograms with 256 bin size were used, along three channels, the histograms generated a total of 768 features. The MinPool and MaxPool features generated 108 features each. In both cases, a standard (70-30) % split was performed on the dataset for generating the training and testing data. Four general evaluation metrics: Accuracy, Precision, Recall, and F1 score based on set IoU thresholds were used throughout the results section. The performance of the hyper parameter-tuned models trained on the extracted features were evaluated using confusion matrix, and four statistical measures: Accuracy, Precision, Recall, and F1 Score. While generating the results, various IoU thresholds were used. At first, the performance was compared based on threshold IoU

= 30% and IoU = 50%. Then local mean IoU for each type and global mean IoU was calculated. Then the results based on the IoU = local mean and IoU = global mean were compared. Finally, the local variance for each type and the global variance was calculated and results were generated by varying variance. While calculating the performance metrics with IoU = local mean, at first the performance metrics were calculated for each individual subset of the dataset. The performance metrics were then averaged to generate the final performance metrics for the set local mean threshold. In all cases, the window size was set to 16 × 16 with 75% overlap. The windows were selected as possible license plate regions when the classification confidence was equal or greater than 90%.

A. Quantitative Results and Analysis of License Plate Detection

The following sections provides quantitative results and their analysis in the case of license plate detection in various color models and methods. The results were generated by testing the system on various color models as well as varying levels of IoU measures.

I. Results of Trained Models on Extracted Features

The extracted features of GLCM in RGB color space were split into train and test sets and several linear and non-linear classifiers were tested to figure out the best performing model. We tested three classifiers: Stochastic Gradient Descent Classifier, Logistic Regression, and Random Forest Classifier. Among these models, the random forest classifier performed best in terms of all the performance metrics: accuracy, precision, recall and F1-score.

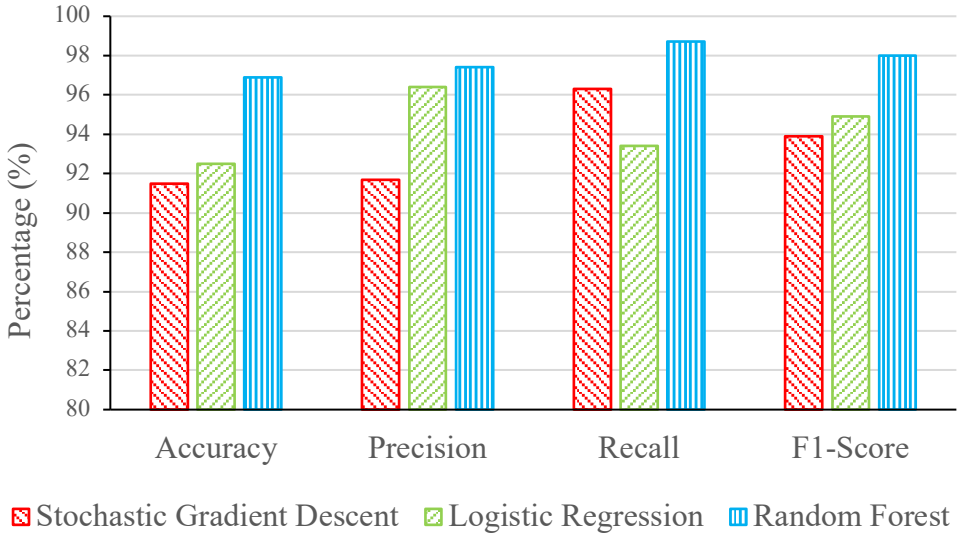


Figure 4.2: Performance metrics of different classifiers in testing of extracted features.

Table 4.3: Hyper parameters of the Random Forest model.

Parameter Name	Value
bootstrap	True
class_weight	“balanced_subsample”
criterion	“gini”
max_depth	50
max_features	“sqrt”
min_samples_leaf	5
min_samples_split	12
n_estimators	100

Figure 4.2 shows the performance metrics of the models in the test set of the extracted features. The random forest classifier model achieved 96.9% accuracy, 97.4% precision, 98.7% recall and 98% F1 score in this case. Thus, in all the detection and localization testing cases, optimized random forest classifiers were used. Table 4.3 presents the hyper parameters of the Random Forest Models. Figure 4.3 shows the performance of optimized Random Forest classifiers in the case of color histogram with minpool and maxpool features in RGB, LAB, and YCbCr color space. It is evident that the chosen color space has no direct impact on feature classification performance.

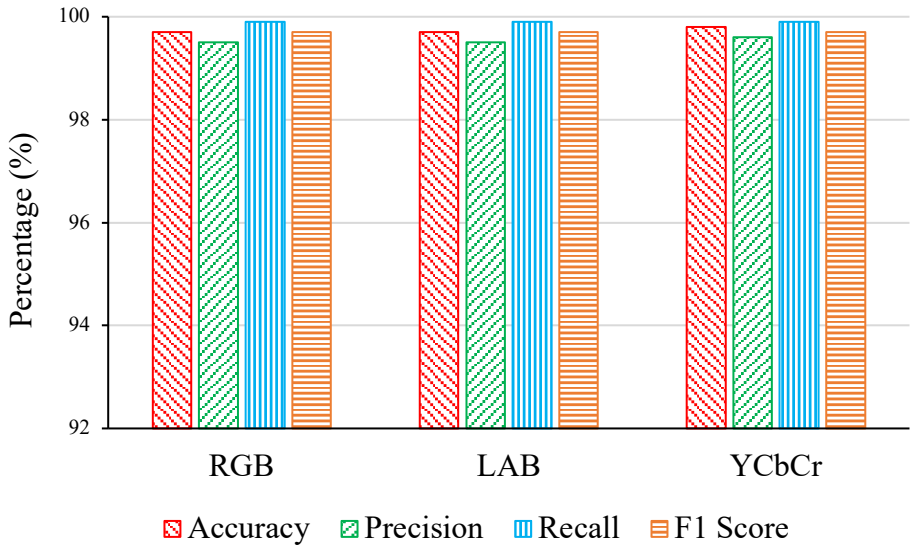


Figure 4.3: Performance metrics of Random forest classifier in different color spaces for CHPOOL features.

II. Results from GLCM in RGB Color Space

Although many models were trained in the extracted features, a decision tree classifier with max_depth = 40 performed the best, and were used for the detection and localization tasks. Various performance metrics were calculated using the optimized model. The same optimized model was used throughout the testing phase of the GLCM in RGB color space.

Table 4.4 presents the local mean IoU and variance ($\boldsymbol{\sigma}$) values. The global calculated mean IoU and $\boldsymbol{\sigma}$ was 0.341 and 0.28, respectively. The used IoU threshold was 0.621 was used when IoU was set to global mean IoU + σ. When global mean IoU - σ was used, the IoU value was set to 0.061.

Figure 4.4 compares the performance of the system when the IoU was set to 30% and 50%, local mean and global mean respectively. Lower IoU results in better results, however, the results are still below average, as the accuracy is around 40% when IoU was 30%. The precision, recall and F1 scores are also very poor in this case.

Table 4.4: Local IoU and $\boldsymbol{\sigma}$ values of corresponding types for GLCM – RGB features.

Type	Back	Back Left	Back Right	Front	Front Left	Front Right	Average
Local IoU	0.405	0.374	0.251	0.4	0.11	0.28	0.303
$\boldsymbol{\sigma}$	0.282	0.265	0.235	0.248	0.224	0.29	0.257

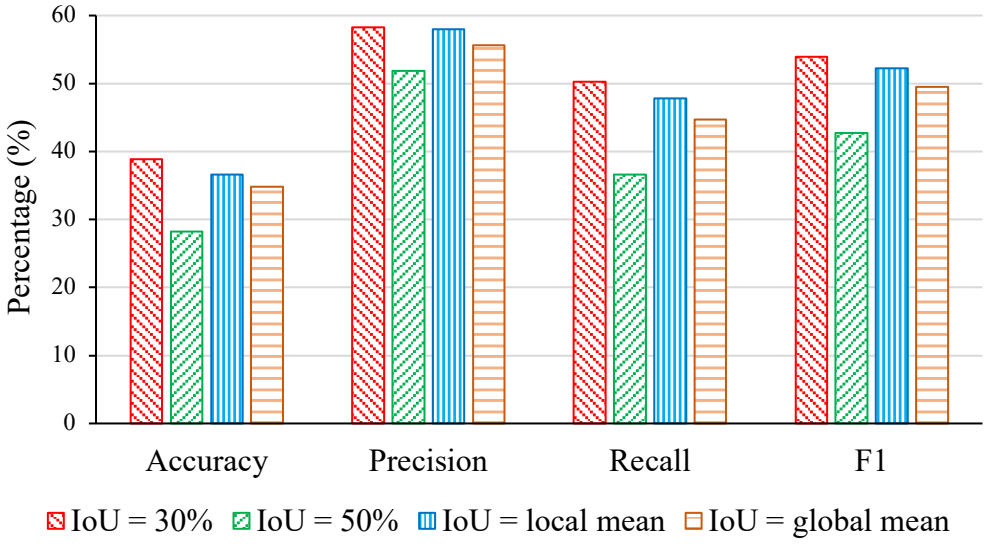


Figure 4.4: Comparative result for different IoU with GLCM – RGB features.

Figure 4.5 shows the evaluation metrics when local and global IoU were set with varying values of $\boldsymbol{\sigma}$. As the $\boldsymbol{\sigma}$ is very high, there are substantial amounts of difference between performance metrics when IoU was set with varying $\boldsymbol{\sigma}$. Althought the radnom forest models achieved high performance in separating LP and nonLP regions using GLCM features, they actually failed while testing in the testing split. A sliding window approach was used to detect and localize potential licnese plates. At first, a sliding window of size 16 in both height and weight is moved across the image. The regions inside the windows are cropped and the features are extracted. The features are then sent to the trained classifier. The classifier decides whether the cropped region is part of a license plate or not. If the cropped region is decided to be a part of a license plate, then the co-ordinates of the window is saved and the window is slid to the next location. After moving the sliding window over the entire image, possible regions of license plates are recognized. The overlapping windows that were recognized as part of the license plate are combined to form potential license plates.

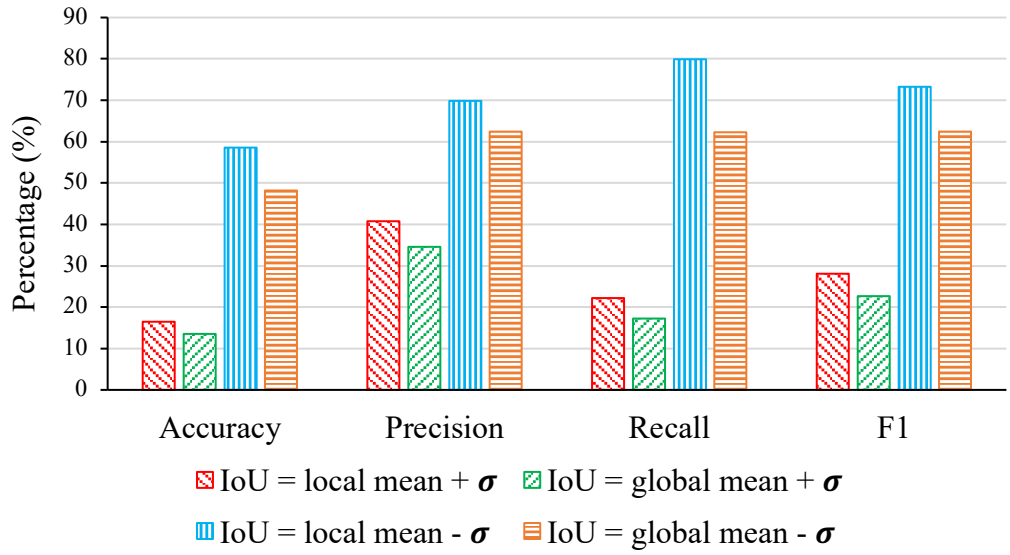


Figure 4.5: Comparative results when IoU was set with varying $\boldsymbol{\sigma}$ for GLCM-RGB features.

Table 4.5: Local IoU and $\boldsymbol{\sigma}$ values of corresponding types for GLCM – HSI features.

Type	**Back**	**Back Left**	**Back Right**	**Front**	**Front Left**	**Front Right**	**Average**
Local IoU	0.35	0.306	0.42	0.325	0.339	0.226	0.328
$\boldsymbol{\sigma}$	0.301	0.291	0.267	0.267	0.265	0.256	0.274

This merging allows to approximate the entire licesnse plate region. After calculating potential license plate regions, the final licnse plate is selected by filtering the potential regions based on their area and aspect ratio. The licnse plate region with the highest aspect ratio and maximum area is selected as the final licnese plate and output.

III. Results from GLCM Features in HSI Color Space

In this section, the results from GLCM feature extraction in HSI color space is presented. Although many models were trained in the extracted features, a random forest classifier with max_depth = 20 and n_estimators = 100 performed the best, and were used for the detection and localization tasks. Max depth represents the number of features used for the decision tress of the random forest and number of estimators represent the number of decision trees used in the random forest ensemble.

Table 4.5 presents the different IoU values that were used while varying local and global variance. Figure 4.6 compares the evaluation metrics when IoU was set to 30% and 50% local mean and global mean, respectively. The average accuracy, precision, recall, and F1 score of the license plate detection system is around 40%, 60%, 55%, and 58%, respectively when IoU = 30%. Thus, this system is also performing very poorly and is not reliable. There is a significant difference between accuracy values when IoU was set to 30% and IoU was set to 50%. The difference in performance is greater than 10%. This indicates that, in a lot of the cases, the IoU were between 30% and 50%.

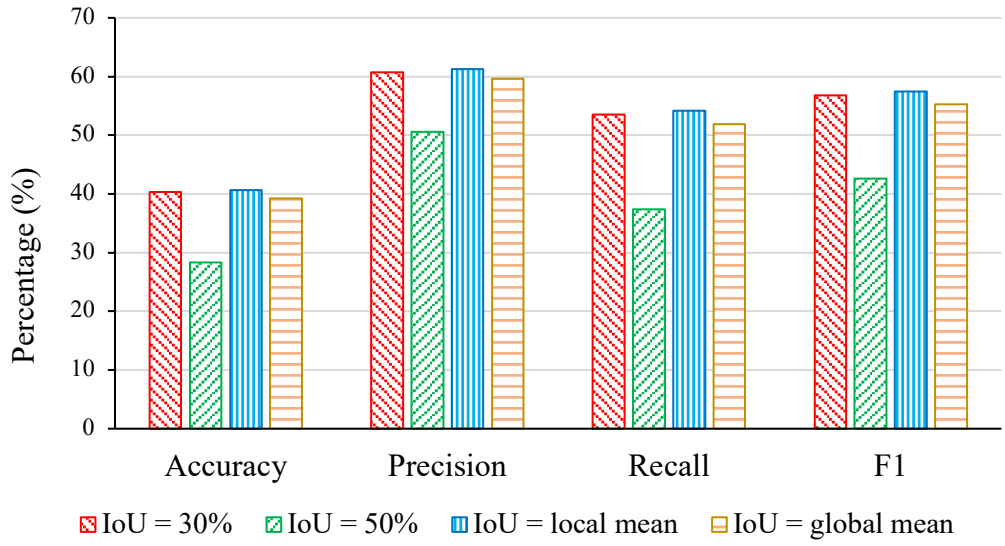


Figure 4.6: Comparative result for different IoU with GLCM – HSI features.

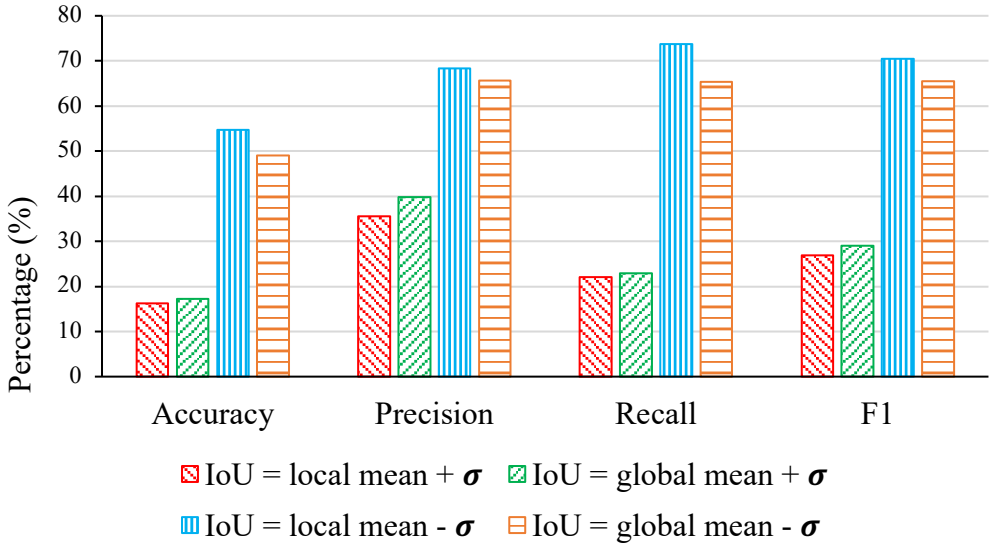


Figure 4.7: Comparative results when IoU was set for varying $\boldsymbol{\sigma}$ for GLCM – HSI features.

While comparing the performance metrics of the license plate detection system when IoU was set to local mean and global mean, respectively, it is evident that both results are similar. However, the performance of the GLCM features in HSI features is marginally better when IoU is set to local mean instead of the global mean.

Figure 4.7 illustrates the comparison of the performance metrics of the license plate detection system when IoU was set based on varying variance. In both cases, the results are similar. The global calculated mean IoU and $\boldsymbol{\sigma}$ was 0.329 and 0.283, respectively. IoU = 0.612 was used when IoU was set to global mean + $\boldsymbol{\sigma}$. IoU = 0.046 was used when global mean – $\boldsymbol{\sigma}$ was used as IoU.

IV. Results from Color Histogram with Minpool and Maxpool Feature in RGB Color Space

This subsection contains results of when the CHPOOL method was used in the bus and trucks dataset in RGB color space.

After the features were extracted and standardized, a random forest classifier with max_depth = 50 and n_estimators = 100 was used for the license plate detection and localization. Table 4.6 depicts the IoU values for each type when the IoU was set based on variance. Figure 4.8 shows the comparison of evaluation metrics of the license plate detection and localization system when IoU was set to 30%, 50%, local mean, and global mean, respectively.

Table 4.6: Local IoU and $\boldsymbol{\sigma}$ values of corresponding types for CHPOOL – RGB features.

Type	Back	Back Left	Back Right	Front	Front Left	Front Right	Average
Local IoU	0.637	0.606	0.579	0.67	0.632	0.594	0.62
$\boldsymbol{\sigma}$	0.201	0.229	0.16	0.196	0.149	0.193	0.188

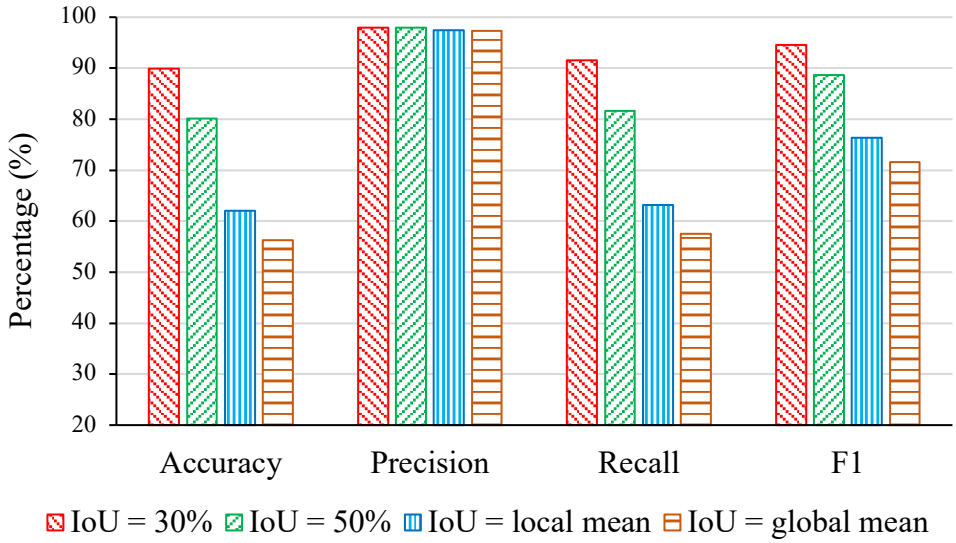


Figure 4.8: Comparative result for different IoU with CHPOOL – RGB features.

The average accuracy, precision, recall, and F1 score was around 90%, 98%, 91%, and 95%, respectively when IoU was set to 30%. This is a huge change from the GLCM feature extraction method. While comparing the evaluation metrics when IoU was set as local mean and global mean, respectively, it is evident that the precision value is identical for all cases. However, while IoU was set to global mean, the accuracy, recall, and F1 scores were the lowest. This is due to the high levels of threshold. The performance metrics was highest when the IoU was set to 30%. However, when IoU was set to 50%, the results were also very good.

The global calculated mean IoU and $\boldsymbol{\sigma}$ was 0.631 and 0.197, respectively. IoU = 0.828 when IoU was set to global mean + $\boldsymbol{\sigma}$ and IoU = 0.434 when IoU was set to global mean – $\boldsymbol{\sigma}$. Figure 4.9 depicts the comparison of the evaluation metrics of the detection and localization system when IoU was set with varying $\boldsymbol{\sigma}$. It is evident from Figure 4.9 that the $\boldsymbol{\sigma}$ value is very high. This is resulting in drastically different performance metrics.

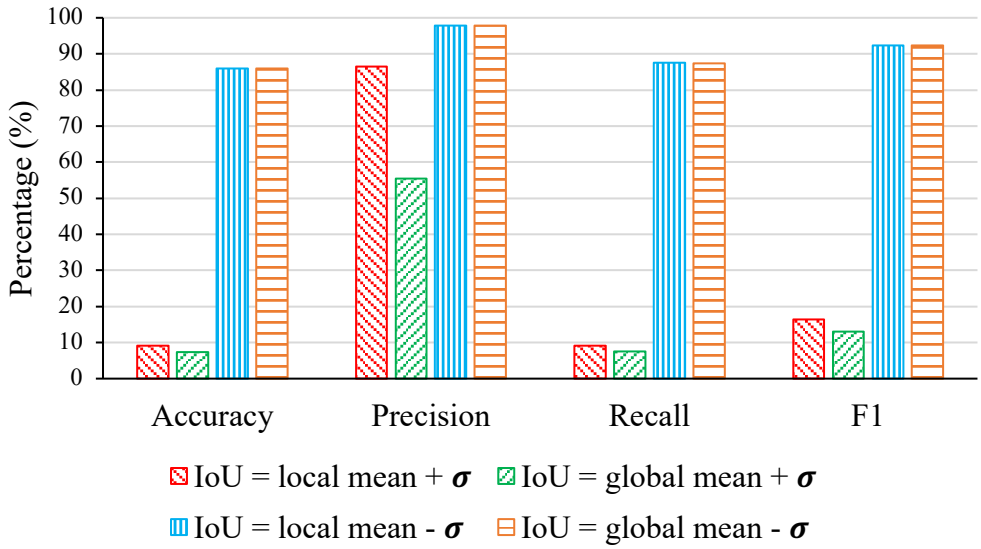


Figure 4.9: Comparative results when IoU was set with varying $\boldsymbol{\sigma}$ for CHPOOL-RGB features.

Table 4.7: Local IoU and $\boldsymbol{\sigma}$ values of corresponding types for CHPOOL – LAB features.

Type	**Back**	**Back Left**	**Back Right**	**Front**	**Front Left**	**Front Right**	**Average**
Local IoU	0.648	0.579	0.506	0.67	0.612	0.612	0.604
$\boldsymbol{\sigma}$	0.188	0.258	0.223	0.144	0.173	0.174	0.193

V. Results from Color Histogram with Minpool and Maxpool Feature in LAB Color Space

To perform the detection and localization task, a random forest classifier with maximum tree depth of 50 and number of decision trees as 100 was trained.

Table 4.7 shows the various IoU that were used in generating the evaluation metrics for the detection and localization system in Figures 4.10 and Figure 4.11. The global calculated mean IoU and $\boldsymbol{\sigma}$ is 0.622 and 0.197, respectively. IoU = 0.819 when IoU was set to global mean + $\boldsymbol{\sigma}$ and IoU = 0.425 when IoU was set to global mean – $\boldsymbol{\sigma}$.

Figure 4.10 shows the comparison of the evaluation metrics for the license plate detection system when IoU was set to 30% and 50%, respectively. The average accuracy, precision, recall, and F1 score is around 88%, 97%, 90%, 92%, respectively. Figure 4.10 also illustrates the evaluation metrics when IoU was set to local mean and global mean, respectively. Figure 4.11 illustrates the performance of the license plate detection and localization system when IoU was set with varying levels of $\boldsymbol{\sigma}$.

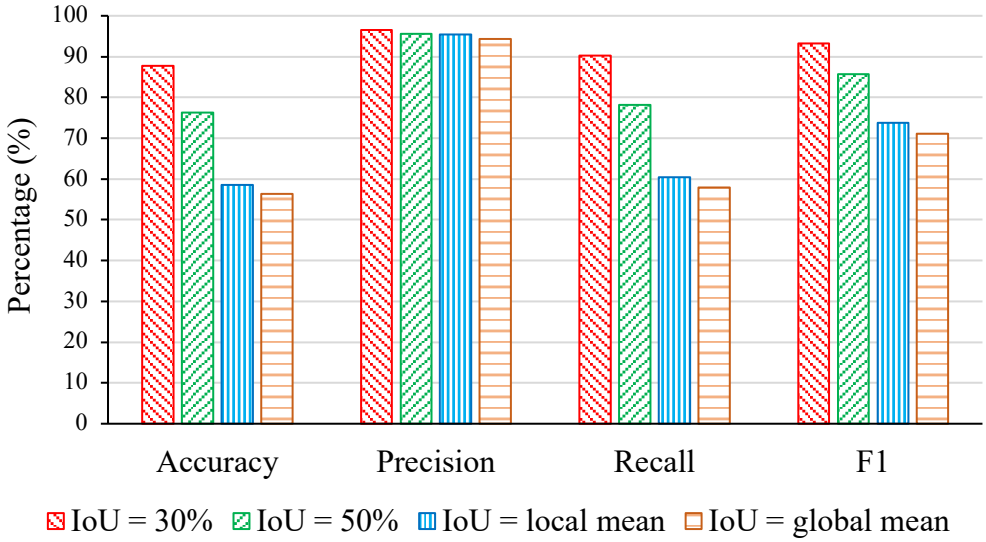


Figure 4.10: Comparative result for different IoU with CHPOOL – LAB features.

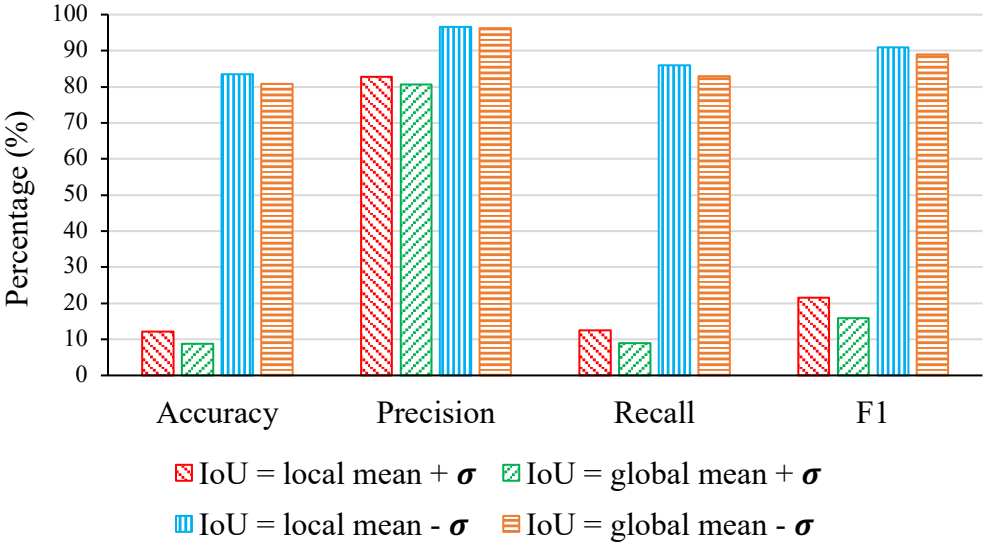


Figure 4.11: Comparative results when IoU was set with varying $\boldsymbol{\sigma}$ for CHPOOL-LAB features.

One interesting factor is that, in terms of accuracy, there is near 10% difference between the performances of the system when IoU was set to 30% and 50%. However, when IoU was set to local mean and global mean, then the difference between accuracy was 2~3%. The results are very similar to when color histogram with MinPool and MaxPool features were extracted from images in RGB color space.

VI. Results from Color Histogram with Minpool and Maxpool Feature in YCbCr Color Space

To perform the detection task, a random forest classifier with max_depth = 50 and n_estimators = 100 was trained. The optimal settings for the random forest was calculated using grid search technique. In grid search method, multiple models are trained over the dataset using separate possible configurations per model. For example, for performing a grid search for a Random forest classifier, several values for each of the random forest hyperparameters, some probable values can be set. The performance of the models is then compared and the configurations from the best performing model according to criteria are chosen for the final model. Although the grid search technique provides a rigorous way to search for hyper parameters, it is very time consuming and computationally expensive due to huge search space that increases exponentially with each new added parameter for testing.

It is evident that the model has achieved a good degree of generalization. The various IoU that were used in calculating the metrics in Figure 4.12 and Figure 4.13 are presented in Table 4.8. The global mean IoU and $\boldsymbol{\sigma}$ is 0.619 and 0.193, respectively. The average $\boldsymbol{\sigma}$ is 0.191. This indicates that the IoU values are relatively closer to one another and there are no large differences among them.

Figure 4.12 illustrates the performance of system when IoU was set to 30% and 50%, respectively. It also shows the performance metrics of the license plate detection and recognition algorithm when IoU was set to local mean and global mean respectively. There are nearly 10% difference between performance measures when IoU was set to 30% and 50%, respectively. However, in almost all of the cases, the models achieved similar levels of precision. The results are similar when IoU was set to local mean and global mean, respectively. The system achieved similar levels of accuracy, precision, recall and F1 score when IoU was set to 30%. Figure 4.13 illustrates the performance of the license plate detection and localization system when IoU was set with varying $\boldsymbol{\sigma}$.

The large difference between performance metrics when IoU was set to local mean + $\boldsymbol{\sigma}$ and IoU was set to local mean + $\boldsymbol{\sigma}$ indicate that, the variance of the IoU was too high. There is nearly 70% difference between the performance metrics when IoU was set to local, global mean – $\boldsymbol{\sigma}$ and local, global mean + $\boldsymbol{\sigma}$.

Table 4.8: Local IoU and σ values of corresponding types for CHPOOL – YCbCr features.

Type	Back	Back Left	Back Right	Front	Front Left	Front Right	Average
Local IoU	0.653	0.599	0.552	0.682	0.623	0.603	0.619
σ	0.192	0.258	0.216	0.136	0.15	0.195	0.191

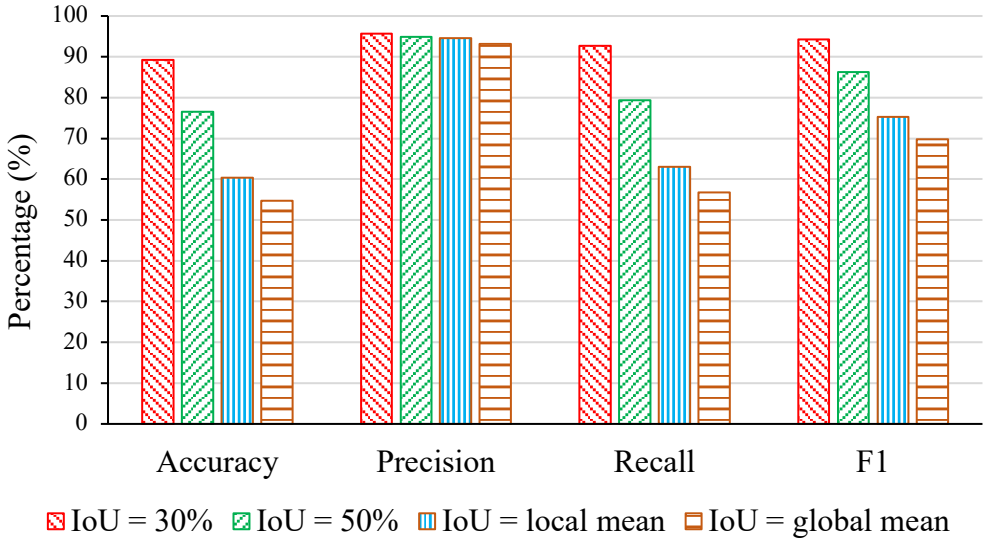


Figure 4.12: Comparative result for different IoU with CHPOOL – YCbCr features.

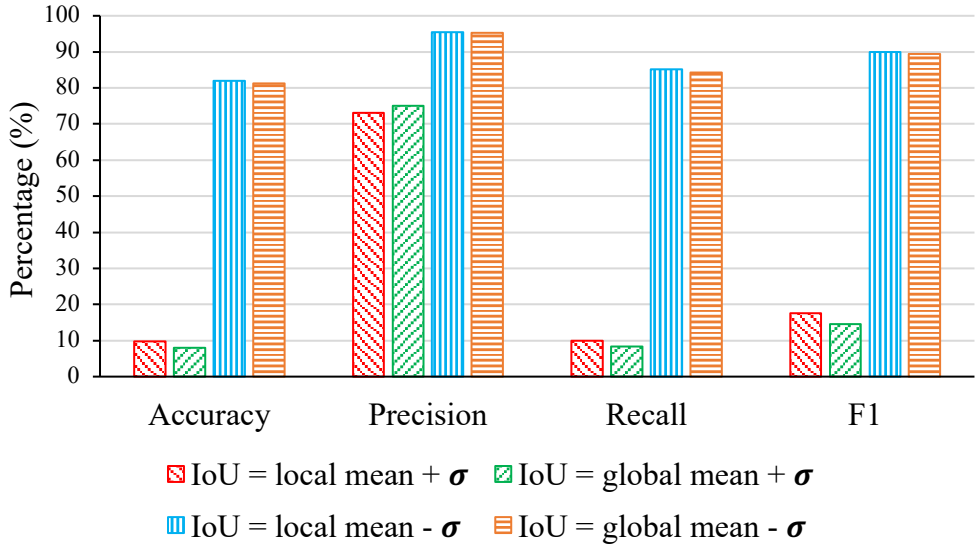


Figure 4.13: Comparative results when IoU was set with varying σ for CHPOOL- YCbCr features.

It is evident from the presented results that, the color space used does not make significant changes to the performance of the license plate detection and localization system when Color histogram with MinPool and MaxPool features are used.

VII. Results from Color Histogram with Minpool and Maxpool Feature in RGB Color Space in the Private Cars Image Dataset.

For the detection and localization purpose, a decision tree classifier with max_depth = 40 was used. The global calculated mean IoU and $\boldsymbol{\sigma}$ was 0.634 and 0.193, respectively. The IoU that were used in Figure 4.14 and Figure 4.15 are presented in Table 4.9.

Figure 4.14 represents the comparison of the performance of the system when IoU was set to 30%, 50%, local mean, and global mean respectively. It also represents the performance of the system when IoU was set to local mean and global mean, respectively.

It can be seen that, when IoU was set to 30% while using the CHPOOL features in RGB color space for private cars dataset, the accuracy, precision, recall, and F1 score was 66%, 84.9%, 74.6% and 79.4%, respectively. However, when IoU was set to 50%, the accuracy, precision, recall, and F1 score was 54.3%, 82%, 61.3%, and 70.1%, respectively. This huge difference between performance measures indicate that the overall values of IoU were pretty low and only half of the license plates were detected with greater than 50% IoU.

The performance of the detection system when IoU was set with varying levels of $\boldsymbol{\sigma}$ is presented in Figure 4.15. The highest accuracy, precision, recall, and F1 score was 68%, 85%, 77%, and 81%, respectively, when the IoU was set to global mean IoU – $\boldsymbol{\sigma}$. The performance metrics were similar when IoU was set to local mean – $\boldsymbol{\sigma}$ and global mean – $\boldsymbol{\sigma}$.

Table 4.9: Local IoU and $\boldsymbol{\sigma}$ values of corresponding types for CHPOOL – RGB features in private car dataset.

Type	Back	Back Left	Back Right	Front	Front Left	Front Right	Average
Local IoU	0.583	0.477	0.434	0.555	0.485	0.419	0.492
$\boldsymbol{\sigma}$	0.282	0.249	0.264	0.286	0.251	0.281	0.269

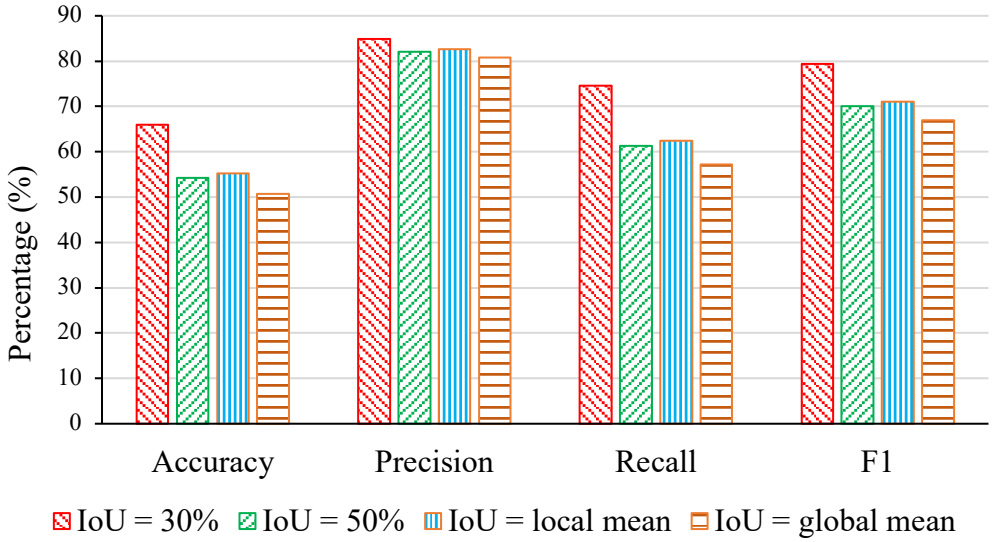


Figure 4.14: Comparative result for different IoU with CHPOOL – RGB in private car dataset.

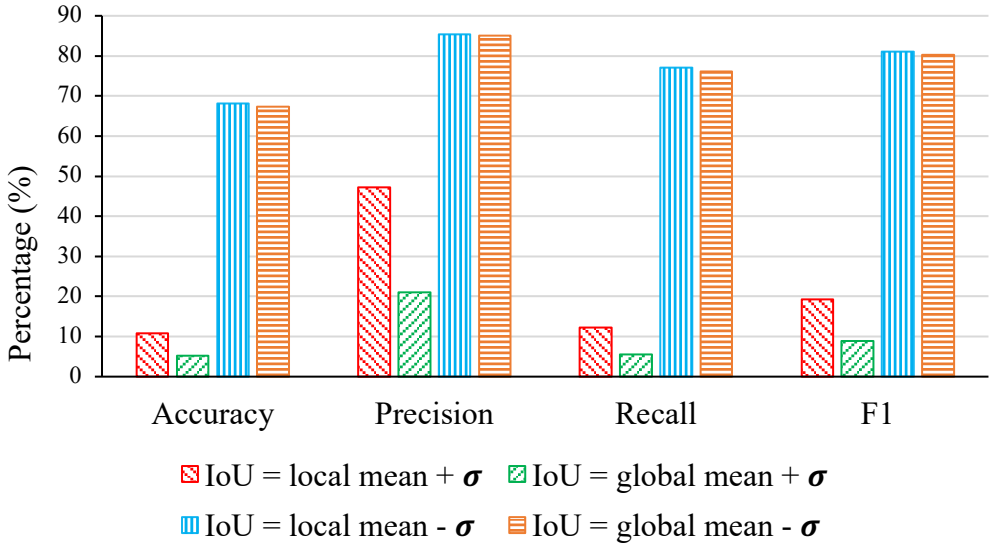


Figure 4.15: Comparative results when IoU was set with varying $\boldsymbol{\sigma}$ for CHPOOL – RGB in private car dataset.

VIII. Overall Performance and Comparison with PKU Dataset

Table 4.10 shows the overall performance of the detection and localization process while using the color histogram with minpool and maxpool features and GLCM features in RGB, LAB, and YCbCr color spaces. The CHPOOL features was experimented on RGB, LAB, and YCbCr color spaces.

Table 4.10: Overall performance measures (in %) of Detection and Localization in different color spaces for CHPOOL and GLCM features.

Color Space	Performance Metrics	Methods	
		CHPOOL	GLCM
RGB	Accuracy	80.1	28.2
	Precision	97.9	51.9
	Recall	81.6	36.6
	F1 Score	88.6	42.7
LAB	Accuracy	76.2	-
	Precision	95.6	-
	Recall	78.2	-
	F1 Score	85.7	-
YCbCr	Accuracy	76.5	-
	Precision	94.9	-
	Recall	79.3	-
	F1 Score	86.3	-

However, the GLCM feature was experimented on the RGB color space. In both cases, the results are presented when IoU was set to 50%. The results are similar across the different color spaces for CHPOOL features. Thus, it is evident that, color spaces do not directly impact the detection and localization performance in case of CHPOOL. However, the GLCM features performed very poorly in both RGB and HSI color space when the IoU was set to 50%. The performance was also very poor when IoU was set to 30% as shown in Figure 4.4. In that case, the GLCM features in RGB color space achieved only 40% accuracy, compared to the ~90% accuracy of the CHPOOL features in RGB color space. Table 4.11 shows the performance of the detection and localization process in each category of the bus and trucks dataset while using the CHPOOL features in RGB color space. The performance in BackRight category is very poor when compared to other categories. This is due to extreme angles and lower number of samples.

The PKU (Peking University) benchmark dataset [46] contains a total of 3977 images of Chinese license plates. The photos were captured under varying environmental and lighting

conditions such as daytime, nighttime, nighttime with headlights on, daylight with sunshine glare, daytime with reflective glare, etc. The dataset is divided into 5 subsets: G1-G5. While the images in G1-G4 subsets contain one license plate per image, the images in the G5 dataset contains multiple license plates per image. The resolution of the images in the G1-G3 is 1082 × 728. The images in the G4 and G5 subsets have resolutions of 1600 × 1236 and 1600 ×1200. The images were resized to 640 × 480 pixels before applying our detection system. The comparison results of our method with other state of the art with other existing state-of-the-art license plate detection methods in PKU dataset are presented in Table 4.12. Thus, it can be said that the color histogram with minpool and maxpool features in RGB color space is highly capable of properly detecting license plates in the context of Bangladesh.

Table 4.11: Categorical performance metrics for CHPOOL-RGB.

Category	**Metrics (IoU = 50%)**		
	Accuracy	***Precision***	***Recall***
Back	86.6	97.5	88.6
BackLeft	82.8	92.3	92.3
BackRight	60.0	90.0	60.0
Front	83.3	97.6	87.0
FrontLeft	91.3	100.0	90.9
FrontRight	77.7	100.0	76.4

Table 4.12:Experimental results on PKU dataset for License Plate Detection.

Methods	**Subset**					
	G1	*G2*	*G3*	*G4*	*G5*	*Avg.*
Zhou et al. [44]	95.4	97.8	94.2	81.2	82.0	90.2
Li et al. [45]	98.8	98.4	95.8	81.1	83.3	91.5
Yuan et al. [46]	98.7	98.4	97.7	96.2	97.3	97.6
Li et al. [47]	99.8	99.8	99.8	100	99.3	99.8
Selmi et al. [48]	99.5	99.4	99.4	99.6	99.1	99.4
Our System	90.5	91.1	98.6	76.5	87.9	88.9

Input Image Output Image Input Image Output Image

Figure 4.16: Inputs and outputs of sample image from every type in the dataset.

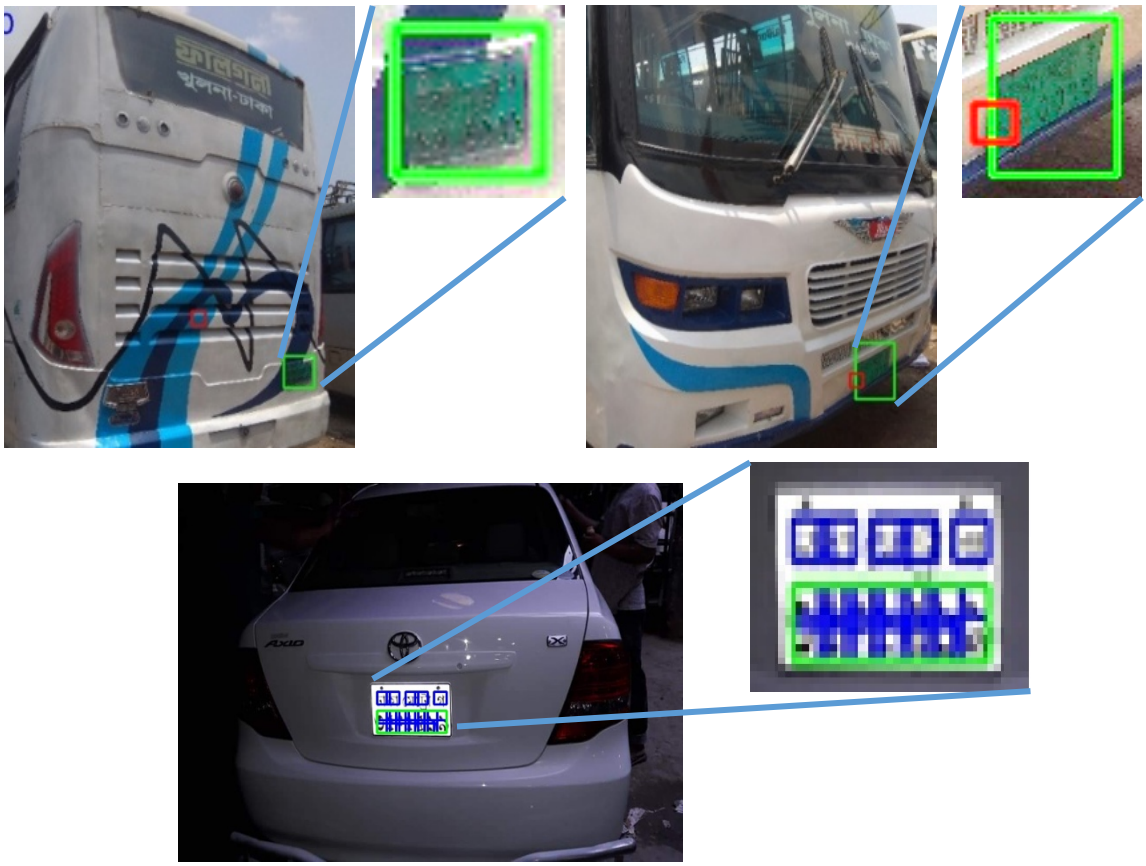


Figure 4.17: Samples of problematic images.

B. Qualitative Results and Analysis of License Plate Detection

The output of the detection system for each type in the dataset is presented in this section. Figure 4.16 shows some sample input image and the corresponding output image for each type in the dataset. Some misclassification samples are shown in Figure 4.17. In all of the samples, the misclassified images are shown and the license plate region is shown in the zoomed inset. In the bus and trucks cases, the detection and localization process failed to detect the license plates due to the extreme angles between the camera and the license plate. In the car sample, the sliding window size is small relative to the license plate size. Due to the relatively large size of the license plate, there is significant distance between the two lines in the plate. Thus, while passing through that region, the sliding window only find white regions. Thus, the detection system fails to properly identify those regions as part of license plates.

4.5.2 Experimental Results of License Plate Recognition

The following sections provide the quantitative and qualitative results related to license plate recognition. The results were obtained using different distance measures. Various performance metrics were calculated.

A. Quantitative Results and Analysis of License Plate Recognition

This section provides the quantitative results of the license plate recognition method for different distance measures.

Table 4.13: Number of license plate, characters and digits.

Type	**Bus and Trucks**	**Private Cars**
Number of plates	300	1000
Characters	3849	12972
Digits	2100	7000
Total	5949	19972

Table 4.14: Recognition result for Squared Euclidean distance.

Distance Type	**Squared Euclidean Distance**		
Symbol Type	**Plate Type** / **Metrics**	**Bus and trucks (Green background, black foreground)**	**Private cars (White background, black foreground)**
Character	Accuracy	93.27%	86.74%
	Precision	95.96%	81.70%
	Recall	93.27%	86.74%
Digit	Accuracy	96.62%	96.59%
	Precision	97.31%	96.60%
	Recall	96.62%	96.59%
Overall	Accuracy	94.45%	90.19%
	Precision	96.45%	86.72%
	Recall	94.45%	90.19%

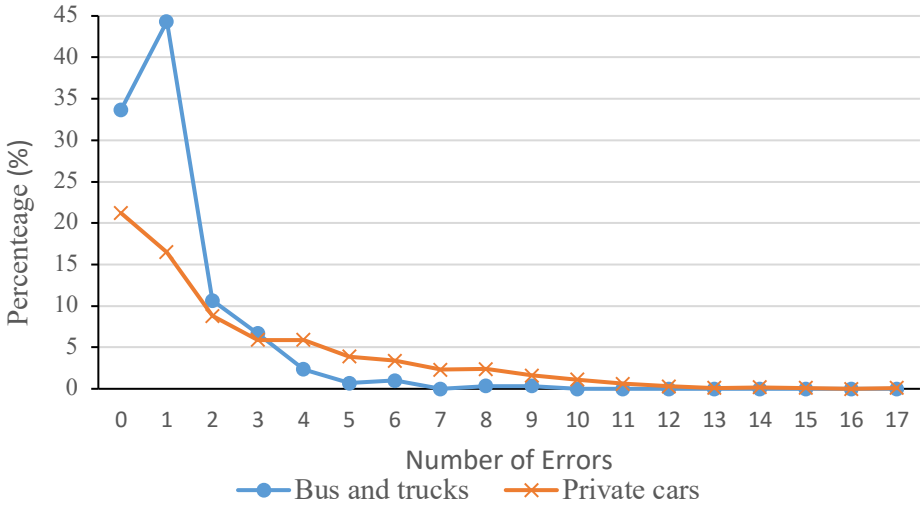


Figure 4.18: Percentile number of errors over all license plates for the Squared Euclidean distance.

Table 4.15: Recognition result for Cosine Similarity.

Distance Type	**Cosine Similarity**		
Symbol Type	**Plate Type** / **Metrics**	**Bus and trucks (Green background, black foreground)**	**Private cars (White background, black foreground)**
Character	Accuracy	91.53%	87.87%
	Precision	93.69%	81.83%
	Recall	91.53%	87.87%
Digit	Accuracy	96.04%	97.03%
	Precision	96.04%	97.04%
	Recall	96.04%	97.03%
Overall	Accuracy	93.12%	91.08%
	Precision	94.78%	86.92%
	Recall	93.12%	91.08%

We have used Manhattan, Euclidean, Squared Euclidean, Cosine similarity and Bhattacharyya distance to predict the class for each character. The performance metrics were calculated based on the overall number of characters and digits while exploring these distances and similarity measures. Table 4.13 represents the overall number of characters

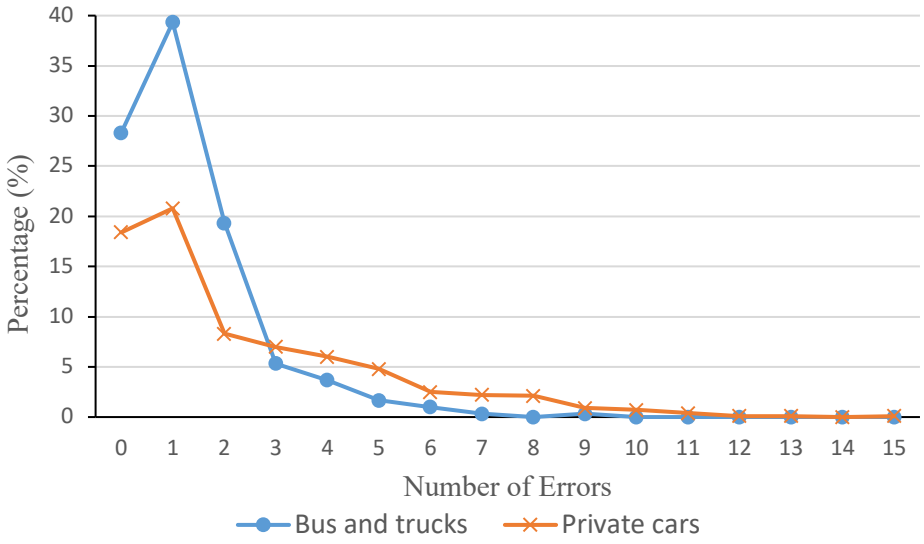


Figure 4.19: Percentile number of errors over all license plates for Cosine Similarity.

Table 4.16: Recognition result for Manhattan distance.

Distance Type	**Manhattan Distance**		
Symbol Type	**Plate Type** / **Metrics**	**Bus and trucks (Green background, black foreground)**	**Private cars (White background, black foreground)**
Character	Accuracy	93.06%	86.79%
	Precision	95.78%	81.77%
	Recall	93.06%	86.79%
Digit	Accuracy	92.81%	94.41%
	Precision	93.48%	97.43%
	Recall	92.81%	94.41%
Overall	Accuracy	92.97%	89.46%
	Precision	94.95%	86.04%
	Recall	92.97%	89.46%

and digits that were considered during the recognition process. Table 4.14 shows the recognition result for Squared Euclidean distance where overall accuracy for bus and trucks license plate is 94.45% and precision is 96.45%. The overall accuracy for private car license plate is 90.19% and precision is 86.72%.

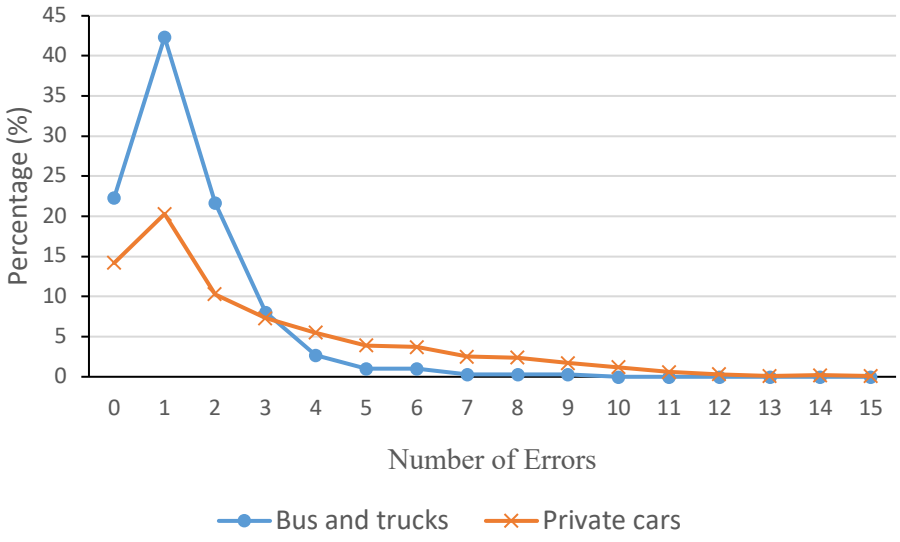


Figure 4.20: Percentile number of errors over all license plates for the Manhattan Distance.

Table 4.17: Recognition result for Euclidean distance.

Distance Type	**Euclidean Distance**		
Symbol Type	**Plate Type** / **Metrics**	**Bus and trucks (Green background, black foreground)**	**Private cars (White background, black foreground)**
Character	Accuracy	93.06%	86.79%
	Precision	95.78%	81.77%
	Recall	93.06%	86.79%
Digit	Accuracy	92.81%	94.41%
	Precision	93.48%	97.43%
	Recall	92.81%	94.41%
Overall	Accuracy	92.97%	89.46%
	Precision	94.95%	86.04%
	Recall	92.97%	89.46%

Figure 4.18 shows the percentile number of errors over all license plates for the Squared Euclidean distance where the percentage of the perfectly recognized license plate is 33.67% for bus and truck dataset and 21.2% for private car dataset. This means that, over all the tested license plates in the bus and trucks dataset, the system recognized all the characters

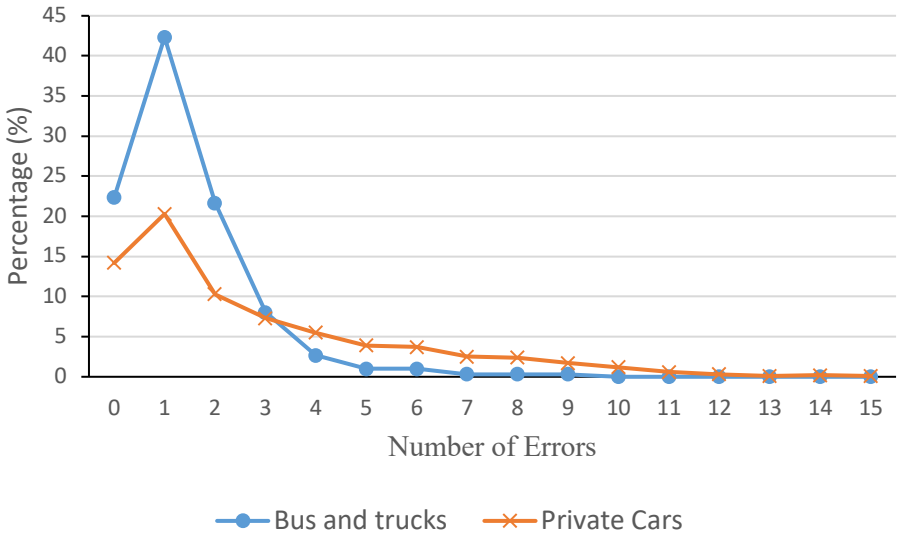


Figure 4.21: Percentile number of errors over all license plates for the Euclidean Distance.

Table 4.18: Recognition result for Bhattacharyya Distance.

Distance Type	**Bhattacharyya Distance**		
Symbol Type	**Plate Type** / **Metrics**	**Bus and trucks (Green background, black foreground)**	**Private cars (White background, black foreground)**
Character	Accuracy	77.42%	65.76%
	Precision	76.69%	59.17%
	Recall	77.40%	65.76%
Digit	Accuracy	87.24%	94.07%
	Precision	87.87%	97.08%
	Recall	87.24%	94.07%
Overall	Accuracy	80.89%	75.68%
	Precision	80.56%	70.58%
	Recall	80.89%	75.68%

without any errors in 33.67% of license plates. Similarly, the system failed to recognize 1 character in 44.33% of the tested license plates in the bus and trucks dataset. Table 4.15 shows the recognition result for Cosine Similarity where overall accuracy for bus and trucks license plate is 93.12% and precision is 94.78%. The overall accuracy for private car license plate is 91.08% and precision is 86.92%.

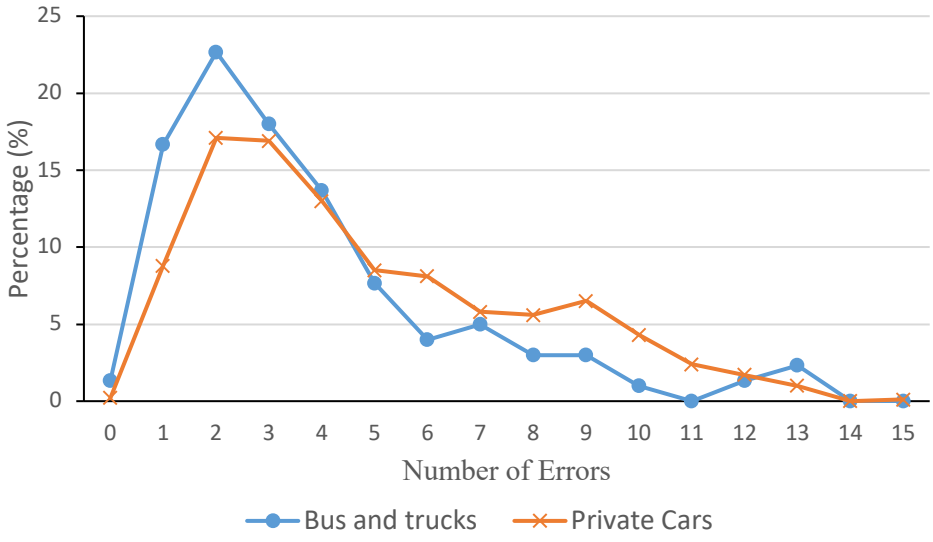


Figure 4.22: Percentile number of errors over all license plates for the Bhattacharyya distance.

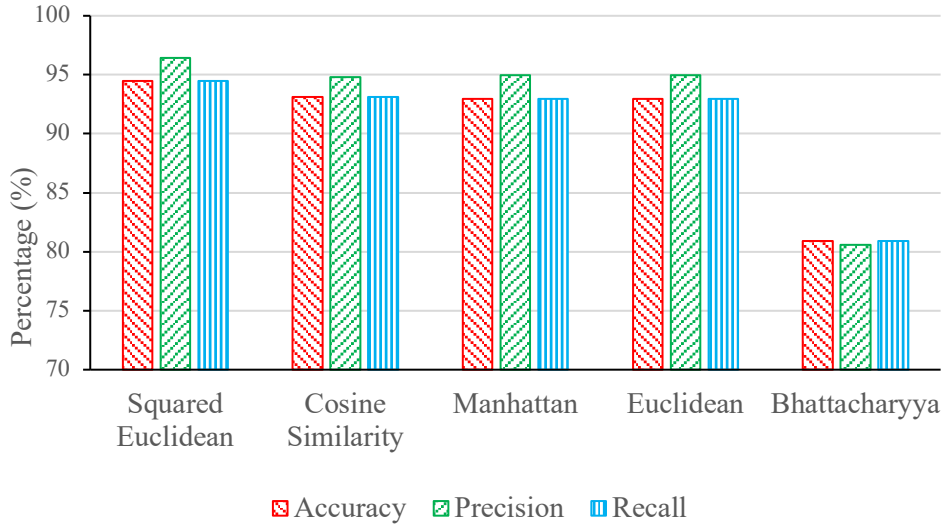


Figure 4.23: Overall performance measures for different distances (Bus and truck dataset).

Figure 4.19 shows the percentile number of errors over all license plates for the Cosine Similarity where the percentage of perfectly recognized license plate is 28.33% for bus and truck dataset and 18.4% for private car dataset. Table 4.16 shows the recognition result for Manhattan distance where overall accuracy for bus and trucks license plate is 92.97% and precision is 94.95%. The overall accuracy for private car license plate is 89.46% and precision is 86.04%.

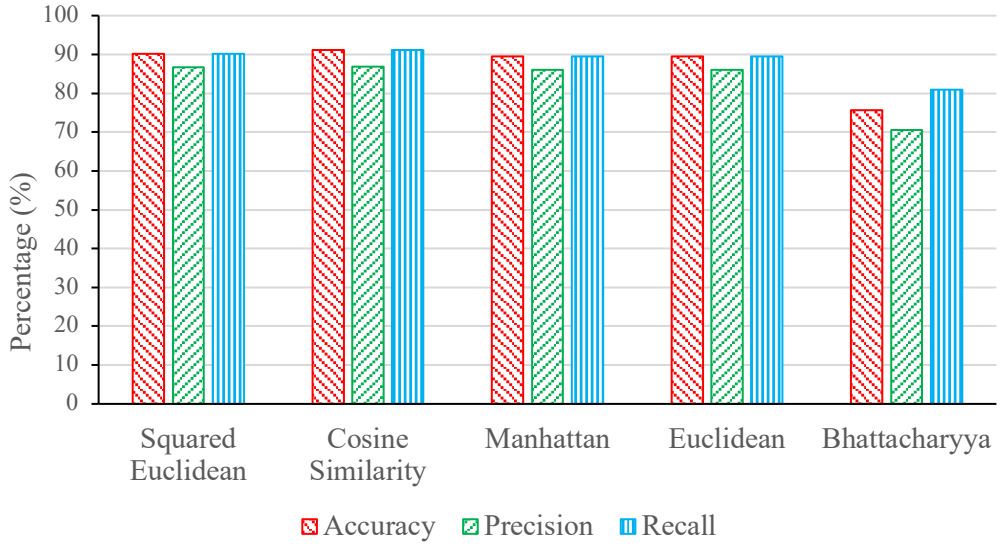


Figure 4.24: Overall performance measures for different distances (Private car dataset).

Figure 4.20 shows the percentile number of errors over all license plates for the Manhattan distance where the percentage of perfectly recognized license plate is 22.33% for bus and truck dataset and 14.2% for private car dataset.

Table 4.17 shows the recognition result for Euclidean distance where overall accuracy for bus and trucks license plate is 92.97% and precision is 94.95%. The overall accuracy for private car license plate is 89.46% and precision is 86.04%.

Figure 4.21 shows the number of errors for the Euclidean Distance where the percentage of perfectly recognized license plate is 22.33% for bus and truck dataset and 14.2% for private car dataset.

Table 4.18 shows the recognition result for Bhattacharyya Distance where overall accuracy for bus and trucks license plate is 80.89% and precision is 80.56%. The overall accuracy for private car license plate is 75.68% and precision is 70.58%.

Figure 4.22 shows the number of errors for the Bhattacharyya Distance where the percentage of perfectly recognized license plate is 1.33% for bus and truck dataset and 0.2% for private car dataset.

Figure 4.23 and Figure 4.24 shows the different results for different measuring techniques for different datasets. From the figure, we can see that highest accuracy and precision results from the Squared Euclidean distance for bus and truck dataset are 94.45% and 96.45%. For the private car dataset Cosine similarity gives the highest accuracy and precision, which are 91.08% and 86.92%. We have observed the results by varying the number of neighbors in KNN which is represented by ‘k’ for Squared Euclidean distance.

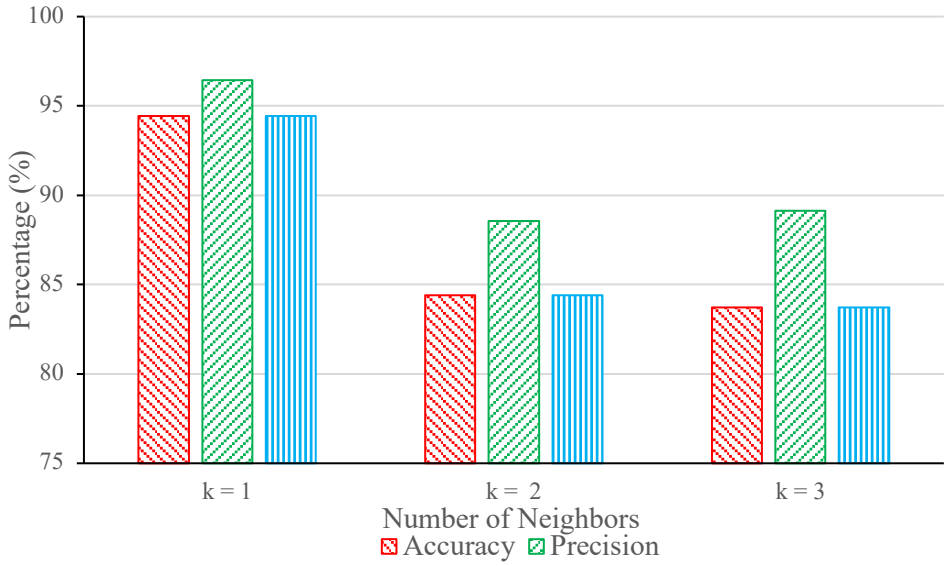


Figure 4.25: Accuracy, Precision and Recall for different number of neighbors in KNN (Squared Euclidean Distance).

Figure 4.25 shows the accuracy, precision and recall for k = 1, k = 2 and k = 3 for bus and truck dataset. From the figure we can see that, the highest accuracy, precision and recall results when the number of neighbors is 1.

B. Qualitative Results and Analysis of License Plate Recognition

As described before, the Bangladeshi license plate is composed of mainly two parts. The character parts consist of the district name, the word metropolitan, a hyphen and a single character which is used to recognize vehicle category. The other part contains the number of the license plate. In recognition phase, all the characters for bus, truck and private car license plates have been recognized. Figure 4.26 shows the sample of outputs in character recognition phase which contains both green and white background license plates. Some characters have been misclassified for noisy license plate. Figure 4.27 shows some example of missclassification for the noisy images. In the input image situated at the first row, the license plate has a black shadow in the right corner. For this reason, the last character has broken after the preprocessing steps. In the image at the second row, some characters are colored as background. So those characters have broken after thresholding. In the image at the third row, the resolution of the license plate is very low. So some characters are broken during segmentation. So the system gives the wrong result.

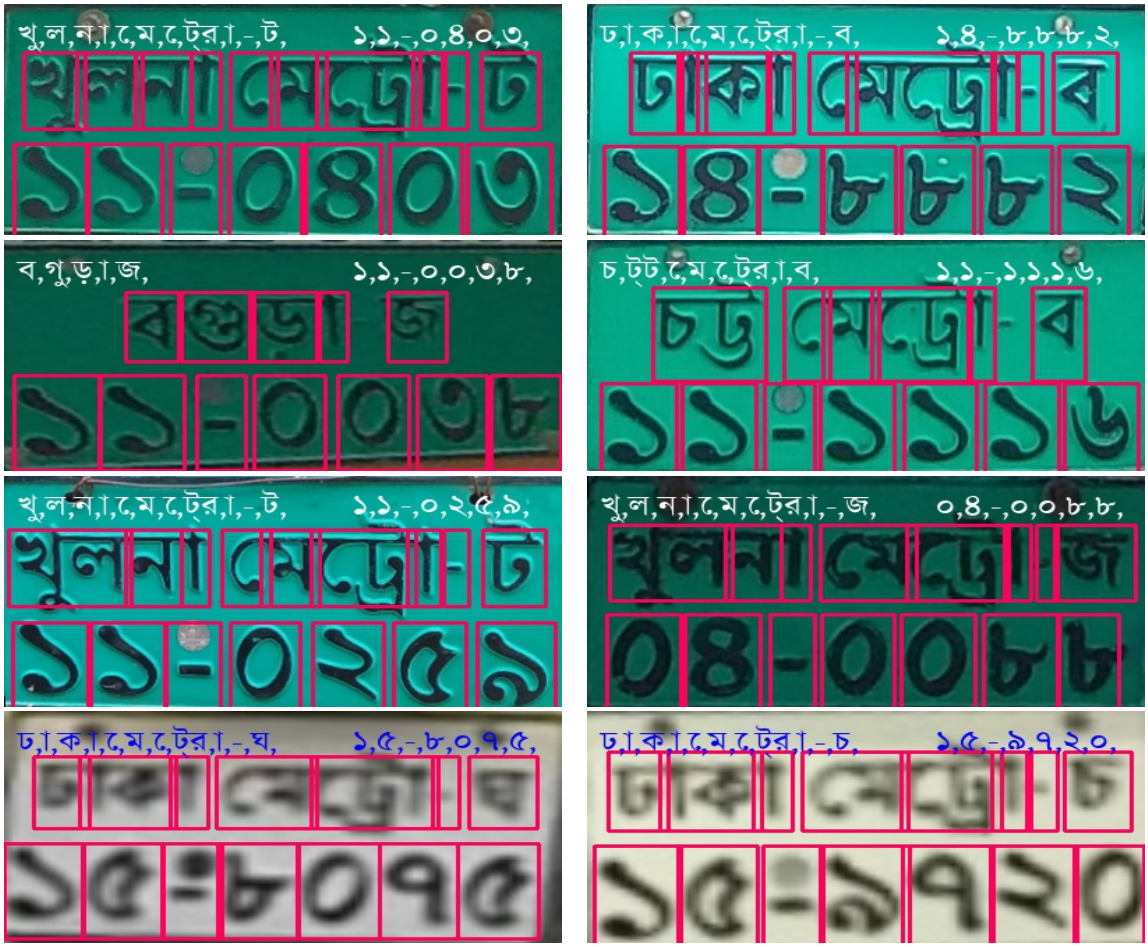


Figure 4.26: Some sample outputs from recognition phase.

Original Image Thresholded Image Final Output

Figure 4.27: Misclassification for noisy license plate images.

4.6 Conclusion

In this chapter, the result of the license plate detection task is described briefly. Overall good accuracy was found while using the Color Histogram with Minpool and Maxpool features in RGB color space. Maximum average accuracy was found to be 89.5% in case of 30% IoU. In this case, the average precision, recall, and f1 score was measured to be 98%, 91.5%, and 94.6%, respectively. From the experimental result of character recognition phase, we can see that, Squared Euclidean distance achieved the highest accuracy and precision for bus and truck dataset and Cosine Similarity achieved the highest accuracy and precision for private car dataset. The highest overall accuracy for bus and truck dataset is 94.45% and precision is 96.45%. The highest overall accuracy for private car dataset is 91.08% and precision is 86.92%. A major reason behind the good performance in the bus and truck dataset and the relatively bad performance in the private car dataset in the recognition phase is the image quality. License plates occupy a very small area in the images. Thus, for building the license plate only dataset, the license plates need to be zoomed and cropped from the original images containing the vehicle images and background objects. As the bus and truck dataset has very high quality images, the cropped license plates are generally very clear and without any noise. However, the images in private car dataset have relatively bad quality. Thus zooming and cropping the license plates results in severely noisy and blurred images in most cases. This in turn results in broken characters during segmentation process and thus negatively impacts the recognition performance.

CHAPTER V

Conclusion

5.1 Summary

Automatic License Plate Detection and Recognition (ALPDR) systems are one of the applications of Computer Vision that has revolutionized intelligent traffic control systems. Many developed countries use ALPDR systems for a multitude of purposes, from automatic centralized tolling, to law enforcement. ALPDR systems along with automated traffic control systems can help reduce the traffic jams and the vehicle related accidents in Bangladesh if utilized properly. Although, there are growing concerns about whether ALPDR systems protect the privacy of the vehicles or users. In our thesis, we have developed an automatic license plate detection and recognition system for Bangladeshi vehicles. We have created a sizeable dataset of commercial vehicle license plate images. We plan to make the dataset open source in the future. In the detection phase, we have explored two different feature extraction techniques. One is GLCM based features that are used extensively in remote sensing research. We proposed a combination of features: Color Histogram with MinPool and MaxPool for detection. Our proposed features performed well for detection and localization purposes. In the recognition phase, we have proposed using horizontal and vertical histograms for character segmentation and KNN for character recognition. Our proposed method works well in different lighting conditions and is capable of correctly recognizing Bangladeshi license plates.

5.2 Limitations

The limitations of developing an ALPDR system in the context of Bangladesh is as follows:

- In Bangladesh, although there have been rules regarding license plate standardization measures such as dimensions, color, font, etc., they are not strictly followed. A lot of the vehicles did not follow the proper standards set by BRTA. In lot of the cases, the license plates are painted, not printed. A lot of the vehicles used

laminated papers printed in black and white as license plates. The license plates also used different shades of green instead of the green standard set by BRTA. Even commercial vehicles used black and white license plates set for private vehicles. The license plates often times had huge cuts or damages which made it very hard for both detection and recognition purposes. Some number plates were broken, some were covered with mud and dust, some were rusted. There was a lack of fluorescent paint in license plates which made them harder to detect in low lighting conditions. In some license plates, the text color was same as the background color, which made them very hard to detect.

- The sheer number of unregistered vehicles in Bangladesh is also very concerning. While capturing images for the vehicle license plate dataset, we encountered a lot of vehicles that either had incomplete license plates or no license plates.
- For creating a generalized system, the size of the dataset plays a huge role. Although we collected the dataset on our own, we could not capture nearly enough images to compete with benchmark datasets in other countries. This problem could be remedied if the government agencies properly helped. We reached out to national agencies like the DMP, DMP terrorism unit, DMP cyber-crime unit, bridge authorities, etc. and were ready to provide any necessary documents. Especially large bridges that performs tolling has stationary cameras installed in the toll booth. However, the agencies told us to contact another agency and so on. The lack of support is astonishing.
- Most ALPDR systems are stationary, which reduces the complexity of the segmentation process. In stationary systems, as the background is static, the vehicles can be easily segmented using background subtraction method, which reduces the problem space.
- The resolution of the captured images is another problem. Although the captured images were high resolution, the images were down-sampled to lower resolution for faster processing. Large images would require a lot of processing power. The lower resolution oftentimes made the license plates very tiny and very hard to detect.
- Lack of standardization during the image capturing process is another limitation. As we did not capture the images with specific distances from vehicles, viewing angles

etc. in mind, our created dataset of commercial vehicles has become very challenging. In some cases, the license plates have dimensions of 16 × 48, which is very small.

- However, the private car images were captured very close to the vehicles, thus the license plate area is very large. In those cases, the 16 × 16 window size is inadequate and too small. As a result, in a lot of the cases, the windows classify the gap between the character and numbers in the license plate as non-license plate regions, thus only localizing a half of a license plate.
- As the images for the commercial vehicle dataset was captured when the vehicles were parked, a lot of green background such as grass was in the background. This is also problematic, as the green color of the grass interferes with the green background of the license plate.
- The license plate detection and recognition system is slow, thus not suitable for real-time application.

5.3 Future Work

The following tasks can be performed in the future:

- Transfer learning methods for deep learning have enabled easy training of very large and pre-processed deep learning models in low computation environments. These model weights are pretrained on huge image datasets such as the ImageNet dataset, and are very accurate. Transfer learning enables the use of these models while keeping the pre-trained weights. In the future, transfer learning approaches can be used for both detection and recognition purposes.
- Color information alone might be inadequate to make a generalized automatic license plate detection and recognition system. In the future, feature descriptors such as Speeded-up Robust Features (SURF), SIFT, HoG, Oriented FAST and Rotated BRIEF (ORB), etc. along with dimensionality reduction techniques such as PCA can be used alongside color descriptors for detection and recognition purposes.
- Custom deep neural network based architecture can be developed in the future for the automatic license plate detection and recognition system.

5.4 Conclusion

We have developed an automatic license plate detection and recognition system that works on different license plates and different lighting condition in the context of Bangladesh. For the detection and localization purpose, we have proposed using a combination of features namely Color Histogram with MinPool and MaxPool. These features are perfectly capable of detecting and localizing license plates in Bangladeshi vehicles. We have also proposed our own methodology for the character segmentation and recognition purposes namely horizontal and vertical histogram and KNN, respectively. The proposed methodology is capable of properly segmenting and recognizing the characters in a name plate. We hope the automatic license plate detection and recognition system can help move us towards a more digital Bangladesh.

REFERENCES

[1] "Bangladeshi License Plate Statistics" Accessed on: Jan. 12, 2020. https://brta.portal.gov.bd/sites/default/files/files/brta.portal.gov.bd/monthly_report/d4d56177_644f_44f8_99c4_3417b3d7b0f4/MV_statistics-bangladesh-march-18.pdf

[2] "Bangladeshi registered license plate statistics", Accessed on: Jan. 12, 2020. https://www.ceicdata.com/en/bangladesh/motor-vehicle-registered/motor-vehicle-registered-bangladesh-total,

[3] C. S. Ahn, B. G. Lee, S. S. Yang, and S. C. Park. "Design of car license plate area detection algorithm for enhanced recognition plate." *In 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), IEEE*, Malaysia, 2017, pp. 1-4.

[4] E. I. Abbas, and T. A. Hashim, "Iraqi cars license plate detection and recognition system using edge detection and template matching correlation." *Engineering and Technology Journal, IASJ*, 2016, vol 34:2, pp.257-271.

[5] J. P. D. Dalida, A. N. J. Galiza, A. G. O. Godoy, M. Q. Nakaegawa, J. L. M. Vallester, and A. R. D. Cruz "Development of intelligent transportation system for Philippine license plate recognition." *In 2016 IEEE Region 10 Conference (TENCON), IEEE*, Singapore, 2016, pp. 3762-3766.

[6] S. Das and R. Kumari, 2020, September. "Application of horizontal projection lines in detecting vehicle license plate.", *In 2020 International Conference on Smart Electronics and Communication (ICOSEC), IEEE*, India, 2020, pp. 1-6.

[7] J. Muhammad and H. Altun, "Improved license plate detection using HOG-based features and genetic algorithm.", *In 2016 24th Signal Processing and Communication Application Conference (SIU)*, IEEE, Turkey, 2016, pp. 1269-1272. IEEE.

[8] N. O. Yaseen, S. G. S. Al-Ali, and A. Sengur. "An efficient model for automatic number plate detection using hog feature from new north iraq vehicle images dataset.", *In 2019 1st International Informatics and Software Engineering Conference (UBMYK)*, IEEE, Turkey, 2019, pp. 1-6.

[9] S. Azam and M. Gavrilova, “License plate image patch filtering using HOG descriptor and bio-inspired optimization.” *In Proceedings of the Computer Graphics International Conference, ACM*, Japan, 2017, pp. 1-6.

[10] L. Xie, T. Ahmad, L. Jin, Y. Liu, and S. Zhang, “A new CNN-based method for multi-directional car license plate detection.”, *IEEE Transactions on Intelligent Transportation Systems, 19(2), IEEE,* 2018, pp.507-517.

[11] W. Wang, J. Yang, M. Chen, and P. Wang, “A light CNN for End-to-End car license plates detection and recognition.”, *IEEE Access, IEEE, 2019, vol.* 7, pp.173875-173883.

[12] N. Omar, A. Sengur, and S. G. S. Al-Ali, “Cascaded deep learning-based efficient approach for license plate detection and recognition.”, *Expert Systems with Applications, Elsevier*, 2020, vol. 149, pp.113280.

[13] S. M. Silva and C. R. Jung. “Real-time Brazilian license plate detection and recognition using deep convolutional neural networks.”, *In 2017 30th SIBGRAPI conference on graphics, patterns and images (SIBGRAPI), IEEE*, Brazil, 2017, pp. 55-62.

[14] N. Sulaiman and S. N. Hafidah, M. Jalani, M. Mustafa, K. Hawari, “Development of automatic vehicle plate detection system”, in proc. *2013 IEEE 3rd International Conference on System Engineering and Technology*, *IEEE*, Malaysia, 2013, pp. 130-135.

[15] K. B. Sathya, V. Vaidehi and G. Kavitha, "Vehicle license plate recognition (vlpr)," *2017 Trends in Industrial Measurement and Automation (TIMA)*, *IEEE* India, 2017, pp. 1-6.

[16] A. Choudhury, A. Negi, “A new zone based algorithm for detection of license plate from indian vehicle”, *in proc 2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)., IEEE,* India, 2016, pp. 370-374.

[17] J. V. John, P. Raji, B. Radhakrishnan, and L. Suresh, “Automatic number plate localization using dynamic thresholding and morphological operations”, *in proc. 2017 International Conference on circuits Power and Computing Technologies [ICCPCT], IEEE,* India, 2017, pp. 1-5.

[18] M. T. Shahed, M. R. I. Udoy, B. Saha, A. I. Khan, S. Subrina, “Automatic Bengali number plate reader”, *in proc. 2017 IEEE Region 10 Conference (TENCON)*, IEEE, Malaysia, 2017, pp. 1364-1368.

[19] O. Hommos*, A. Al-Qahtani, A. Farhat, X. Zhai, “HD Qatari ANPR System”, in proc. 2016 International Conference on Industrial Informatics and Computer Systems (CIICS), IEEE, UAE, 2016, pp. 1-5.

[20] Lubna, M. F. Khan and N. Mufti, "Comparison of various edge detection filters for ANPR," *2016 Sixth International Conference on Innovative Computing Technology (INTECH), IEEE*, Ireland, 2016, pp. 306-309.

[21] R. Islam, K. F. Sharif and S. Biswas, “Automatic vehicle number plate recognition using structured elements”, *in proc. 2015 IEEE Conference on Systems, Process and Control (ICSPC 2015), IEEE*, Malaysia, 2015, pp. 44-48.

[22] M. Rahman, M. S. Islam, “A study on Bangladeshi car name plate detection and recognition”, Thesis/Project No. CSER-19-24, KUET.

[23] A. R. F. Quiros, R. A. Bedruz, A. C. Uy, A. Abad, A, Bandala, E. P. Dadios, A. Fernando, "A kNN-based approach for the machine vision of character recognition of license plate numbers," *TENCON 2017 - 2017 IEEE Region 10 Conference, IEEE*, Malaysia, 2017, pp. 1081-1086.

[24] J. Kim, S. Kim, S. Lee, T. Lee and J. Lim, "License plate detection and recognition algorithm for vehicle black box," *in proc.* 2017 International Automatic Control Conference (CACS), *IEEE*, Taiwan, 2017, pp. 1-6,

[25] D. Bhardwaj and H. Kaur, “Comparison of ML algorithms for identification of automated number plate recognition”, *in proc. 3rd International Conference on Reliability, Infocom Technologies and Optimization*, *IEEE*, India, 2014.

[26] L. G. C. Hamey and C. Priest. 2005. “Automatic number plate recognition for Australian conditions.”, *In proc. Digital Image Computing on Techniques and Applications (DICTA '05). IEEE*, Australia, 2005, pp. 14-14.

[27] B. S. Prabhu, S. Kalambur and D. Sitaram, "Recognition of Indian license plate number from live stream videos," *in proc. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE*, India, 2017, pp. 2359-2365.

[28] M. Mondal, P. Mondal, and N. Saha, P. Chattopadhyay, “Automatic number plate recognition using cnn based self synthesized feature learning” *in proc. 2017 IEEE Calcutta Conference (CALCON)., IEEE*, India, 2017, pp. 378-381.

[29] Hall-Beyer, Mryka. "GLCM texture: a tutorial v. 3.0 March 2017.", University of Calgary, Canada. (2017).

[30] R. M. Haralick, K. Shanmugam, and I. H. Dinstein. "Textural features for image classification." *IEEE Transactions on systems, man, and cybernetics*, *IEEE*, 1973, vol. 6, pp. 610-621.

[31] B. Gérard. "Analysis of a random forests model.", *The Journal of Machine Learning Research*, vol. 13:1, pp. 1063-1095.

[32] S. Vladimir, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston. "Random forest: a classification and regression tool for compound classification and QSAR modeling.", Journal of chemical information and computer sciences, vol. 43:6, pp. 1947-1958.

[33] A. Zuhaib, and R. Ali. "Automatic Number Plate Recognition Using Random Forest Classifier." SN Computer Science, vol 1:3, pp. 1-9.

[34] H. Jr, W. David, S. Lemeshow, and R. X. Sturdivant. "Applied logistic regression". Vol. 398. John Wiley & Sons, 2013.

[35] G. David, K. Dietz, M. Gail, M. Klein, and M. Klein. "Logistic regression". New York: Springer-Verlag, 2002.

[36] M. Scott. "Logistic regression: From introductory to advanced concepts and applications". Sage, 2010.

[37] B. Léon. "Large-scale machine learning with stochastic gradient descent." *in proc. International Conference on Computational Statistics, Springer,* France, 2010, pp. 177-186.

[38] Z. Tong. "Solving large scale linear prediction problems using stochastic gradient descent algorithms." *in proc. of the International Conference on Machine Learning (ICML), ACM,* Canada, 2004, p. 116. 2004.

[39] H. Moritz, B. Recht, and Y. Singer. "Train faster, generalize better: Stability of stochastic gradient descent." *in Proceedings on Machine Learning Research (PMLR), ACM*, 2016, pp. 1225-1234.

[40] P. S. Ha and M. Shakeri. "License plate automatic recognition based on edge detection." *In 2016 Artificial Intelligence and Robotics (IRANOPEN), IEEE*, Iran, 2016, pp. 170-174.

[41] R. Duda, and P. E. Hart. "Use of the Hough transformation to detect lines and curves in pictures." *Communications of the ACM*, *ACM*, 1972, vol 15.1, pp. 11-15.

[42] S. Ravi, and A. M. Khan. "Morphological operations for image processing: understanding and its applications." In *Proc. 2nd National Conference on VLSI, Signal processing & Communications NCVSComs-2013*, *IEEE*, India, 2013. pp. 1-5.

[43] S. Roy, and M. Saravanan. "Handwritten character recognition using K-NN classification algorithm." *International Journal of Advance Research and Innovative Ideas in Education (IJARIIE)*, 2017, vol 5, pp. 1245–1250.

[44] V. Ong, and D. Suhartono. "Using K-Nearest Neighbor in optical character recognition." *ComTech: Computer, Mathematics and Engineering Applications*, 2016, vol 7:1, pp. 53-65.

[45] W. Zhou, H. Li, Y. Lu, and Q. Tian, "Principal visual word discovery for automatic license plate detection," *IEEE Trans. Image Process.*, *IEEE*, 2012. vol. 21:9, pp. 4269-4279.

[46] B. Li, B. Tian, Y. Li, and D. Wen, "Component-based license plate detection using conditional random field model," *IEEE Trans. Intell. Transp. Syst.*, *IEEE,* 2013, vol. 14, no. 4, pp. 1690-1699.

[47] Y. Yuan, W. Zou, Y. Zhao, X. Wang, X. Hu, and N. Komodakis, "A robust and efficient approach to license plate detection," *IEEE Trans. Image Process.*, *IEEE*, 2017, vol. 26:3, pp. 1102-1114,

[48] H. Li, P. Wang, and C. Shen, "Toward End-to-End car license plate detection and recognition with deep neural networks," *IEEE Trans. Intell. Transp. Syst.*, *IEEE*, 2019, vol. 20:3, pp. 1126-1136.

[49] Z. Selmi, M. B. Halima, U. Pal, and M. A. Alimi, "DELP-DAR system for license plate detection and recognition," *Pattern Recognit. Lett.*, *Elsevier*, 2020. vol 129, pp 213-223.