



VEHICLE NUMBER PLATE DETECTION AND RECOGNITION SYSTEM IN
BHUTAN



YONTEN JAMTSO

A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Master of Engineering in (Computer Engineering - (Type A 2))

2019

Copyright by Naresuan University

VEHICLE NUMBER PLATE DETECTION AND RECOGNITION SYSTEM IN
BHUTAN



A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Master of Engineering in (Computer Engineering - (Type A 2))

2019

Copyright by Naresuan University

Thesis entitled "VEHICLE NUMBER PLATE DETECTION AND RECOGNITION
SYSTEM IN BHUTAN"

By YONTEN JAMTSO

has been approved by the Graduate School as partial fulfillment of the requirements
for the Master of Engineering in Computer Engineering - (Type A 2) of Naresuan
University

Oral Defense Committee

..... Chair
(Choochart Haruechaiyasak, Ph.D.)

..... Advisor
(Assistant Professor Panomkhawn Riyamongkol, Ph.D.)

..... Internal Examiner
(Associate Professor Panus Nattharith, Ph.D.)

..... Internal Examiner
(Assistant Professor Phongphun Kijsanayothin, Ph.D.)

Approved

.....
(Professor Paisarn Muneesawang, Ph.D.)

for Dean of the Graduate School

| | |
|-----------------------|---|
| Title | VEHICLE NUMBER PLATE DETECTION AND RECOGNITION SYSTEM IN BHUTAN |
| Author | YONTEN JAMTSO |
| Advisor | Assistant Professor Panomkhawn Riyamongkol, Ph.D. |
| Academic Paper | Thesis M.Eng. in Computer Engineering - (Type A 2), Naresuan University, 2019 |
| Keywords | YOLO, Centroid Difference, xgboost classifier, Darknet, Bhutanese license plates, Flip method |

ABSTRACT

Bhutan is a small landlocked country sandwiched between two giants of the world; China in the north and India in the south, east and west. Bhutan had remained isolated from the outside world since 1970s when it was opened to foreigners. Bhutan is one of the last countries to introduce television and internet. With the rapid development and urbanization, more people have begun to migrate from rural to urban areas due to more opportunities and availability of facilities. As the number of people increases in the urban areas, there has been a problem of job opportunities, traffic congestion and human health problem. Apart from this, with the increasing number of vehicles in the country, it has become crucial to automate the traditional way of managing traffic since it creates more traffic congestion and needs more human resources to manage the traffics. The study aimed to develop an automatic license plate recognition system for a Bhutanese license plate and an algorithm for the extraction of features from the characters.

In the study, the detection of vehicle and localization of license plate was based on the latest state-of-the-art YOLO (You Only Look Once) object detector. An automatic license plate localization from the vehicle was put forward to reduce the number of false positives generated by the signboard and other objects since they look similar to the license plate. Once the license plate was extracted, several image preprocessing steps such as automatic cropping of Bhutanese scripts, noise

removal and contrast enhancement was applied to improve the quality of the image before segmenting the characters. Also, a white pixel counting with inversion method was introduced to handle the taxi (BT) license plate since it gives different output after the thresholding operation. For character segmentation, a connected component analysis (CCA) was proposed, which labels all the unique components in the binary image. After segmenting the characters, three statistical concepts: standard deviation, mean absolute deviation and modified z-score were used to remove the unwanted characters segmented during the segmentation phase. After that, the character classifier was modelled by extracting the 4948 features from 17 characters. For feature extractions, a centroid difference(CD) and flip methods along with Hu's moments were proposed to handle those characters having similar Hu's moments. The extracted features were trained and tested using XGBoost classifier.

In the license plate localization, 1050 vehicle images were given to the proposed method. From the experiments, the overall vehicle detection accuracy was 98.5%, and the license plate localization accuracy was 97.9%. The extracted license plates from the localization step were evaluated to check the performance of the segmentation method and achieved a segmentation accuracy of 96.1%. All the correctly segmented license plates were tested to monitor the performance of the recognition method and resulted in overall recognition accuracy of 92.9%.

This research is first of its kind to use Bhutanese datasets for the development of ANPR technology. The proposed method achieved high accuracy and outperformed some methods discussed in the literature.

ACKNOWLEDGEMENTS

First of all, I would like to extend my profound gratitude to His Majesty the King of Bhutan and Naresuan University, Thailand for giving me the scholarship to pursue my postgraduate study in Master of Engineering in Computer Engineering. Also, I would like to extend my sincere gratitude to the Royal University of Bhutan for granting two years study leave to undergo the Master program.

Furthermore, this thesis would not have come into this shape without the continuous support, inspiring guidance, encouragement and valuable suggestions from the advisor, Assistant Professor Dr Panomkhawn Riyamongkol and Lecturer Rattapoom Waranusast, head of NU Vision Lab. Being a member of the NU Vision Lab, I would like to thank all the lab members for creating a great atmosphere to do my thesis and also for guiding me throughout the study period. I must not forget to convey my gratitude to supporting staffs of the Faculty of Engineering for always helping me when I was in need.

The data collection was successful only because of the supports from my friends in Bhutan. So, I would like to thank them for sending me the datasets for my research.

Lastly, I would like to thank my parents and relatives for always encouraging me throughout my entire study period and of course, my friends for giving moral support.

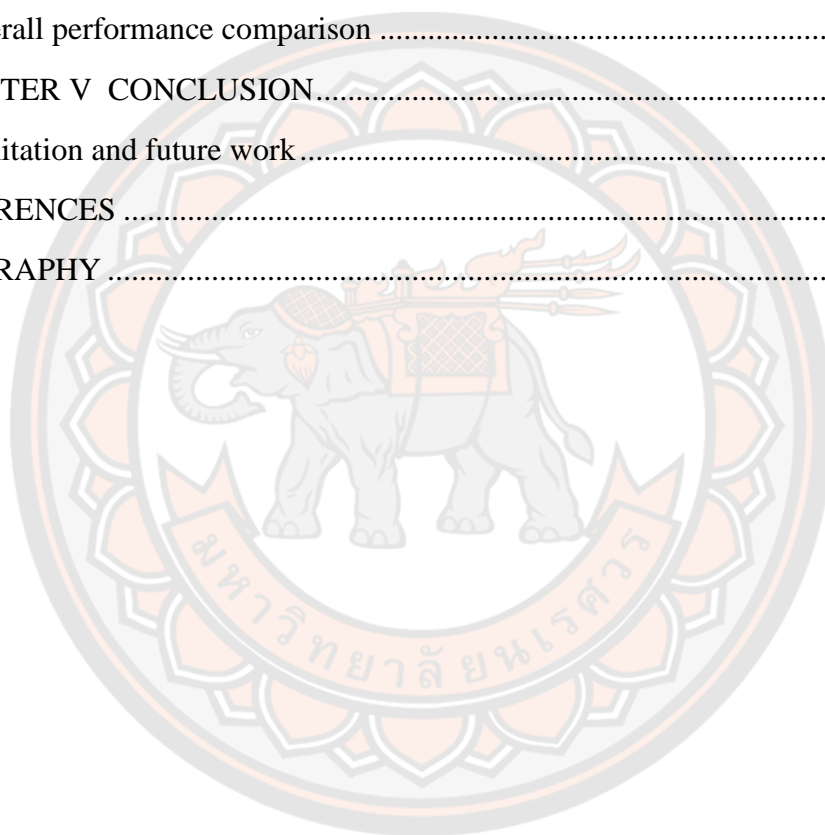
YONTEN JAMTSHO

TABLE OF CONTENTS

| | Page |
|--|-------------|
| ABSTRACT..... | C |
| ACKNOWLEDGEMENTS..... | E |
| TABLE OF CONTENTS..... | F |
| LIST OF TABLES..... | I |
| LIST OF FIGURES..... | J |
| CHAPTER I INTRODUCTION..... | 1 |
| Introduction to the study..... | 1 |
| Background of the study..... | 1 |
| Purpose of the study..... | 6 |
| Problem statement..... | 6 |
| The scope of the study..... | 8 |
| Contribution of the study..... | 9 |
| Keywords..... | 9 |
| CHAPTER II REVIEW OF LITERATURE..... | 10 |
| Introduction..... | 10 |
| Vehicle detection..... | 10 |
| License plate localization..... | 11 |
| Boundary-based approaches..... | 11 |
| Colour-based approaches..... | 12 |
| Texture-based approaches..... | 12 |
| Feature-based approaches..... | 13 |
| Character segmentation..... | 14 |
| Pixel connectivity-based approach..... | 14 |
| Projection-based approach..... | 14 |
| Character recognition..... | 15 |

| | |
|---|----|
| Template matching approach | 15 |
| Learning-based approach | 15 |
| CHAPTER III RESEARCH METHODOLOGY | 17 |
| Introduction..... | 17 |
| System overview..... | 17 |
| License plate localization | 18 |
| Working of YOLO | 20 |
| Loss function | 24 |
| YOLO version 2 | 25 |
| Data annotation..... | 27 |
| Algorithm for localization of license plate inside the vehicle..... | 31 |
| Train model in Google Colaboratory | 33 |
| Character segmentation | 33 |
| Crop upper part of the license plate..... | 34 |
| Grayscale conversion | 35 |
| Noise removal by an iterative bilateral filter..... | 36 |
| Contrast enhancement using adaptive histogram equalization..... | 37 |
| Otsu's thresholding..... | 37 |
| White pixel counting with the inversion method | 38 |
| Connected component analysis (CCA) | 39 |
| Working with CCA | 40 |
| Character recognition..... | 44 |
| Removal of unwanted characters | 46 |
| Feature extraction using Hu's moments | 54 |
| Feature extraction from Centroid Difference method | 57 |
| Feature extraction from Vertical and Horizontal flip method..... | 59 |
| CHAPTER IV RESULTS AND DISCUSSION..... | 64 |
| Introduction..... | 64 |
| License plate localization | 64 |

| | |
|--|-----|
| Performance evaluation of vehicle detection and license plate localization | 66 |
| Character segmentation | 74 |
| Performance evaluation of segmentation | 74 |
| Character recognition..... | 78 |
| Train and test the extracted features in WEKA..... | 78 |
| Implementation of XGBoost classifier on Python | 83 |
| Performance evaluation of character recognition..... | 91 |
| Overall performance comparison | 99 |
| CHAPTER V CONCLUSION..... | 101 |
| Limitation and future work..... | 102 |
| REFERENCES | 104 |
| BIOGRAPHY | 111 |



LIST OF TABLES

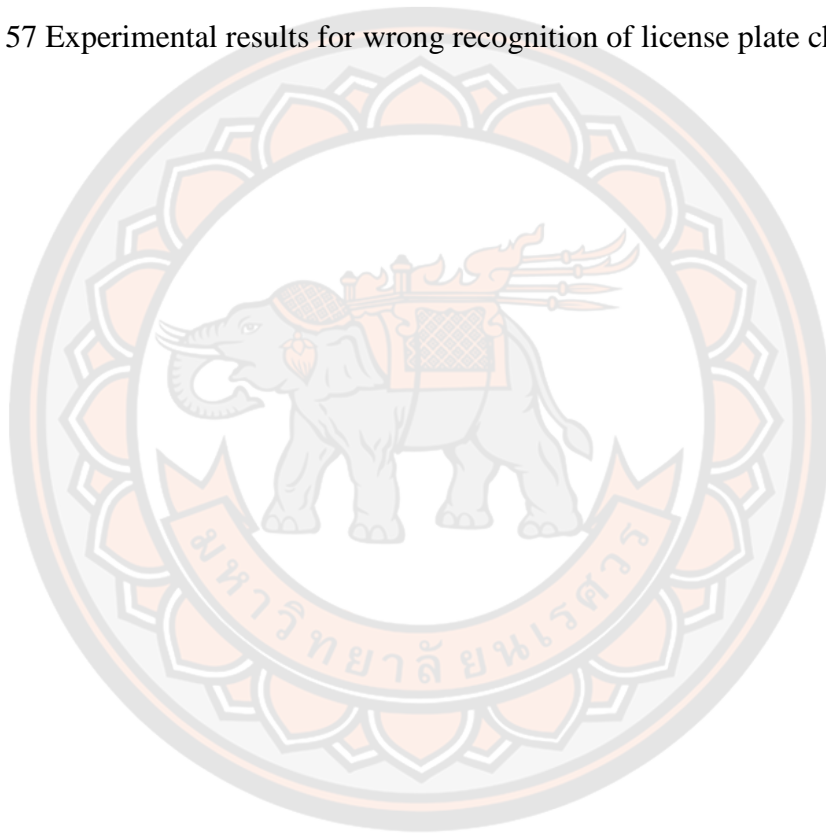
| | Page |
|---|-------------|
| Table 1 Bhutanese vehicle number plates..... | 4 |
| Table 2 Vehicle ownership with colour information | 5 |
| Table 3 Jurisdiction of the regions with region code | 5 |
| Table 4 Darknet-19 framework..... | 27 |
| Table 5 Selection of methods based on number of contours | 45 |
| Table 6 Valid license plate characters having standard deviation less than one | 50 |
| Table 7 Removal of unwanted characters using mean absolute deviation | 51 |
| Table 8 Removal of unwanted characters using modified z-score method | 53 |
| Table 9 Frequency of characters used in the classifier with its ASCII value | 54 |
| Table 10 Examples of similar Hu's moments for different characters | 57 |
| Table 11 Tabulation of average IOU and average precision of each class | 65 |
| Table 12 Results from vehicle detection and license plate localization | 68 |
| Table 13 Accuracy from character segmentation | 74 |
| Table 14 Results of classification matrix based on Hu's moments (WEKA)..... | 79 |
| Table 15 Results of classification matrix based on proposed methods (WEKA)..... | 81 |
| Table 16 Summary of parameters used in the XGBoost classifier with the accuracy | 85 |
| Table 17 Confusion matrix example..... | 86 |
| Table 18 Classification report showing precision, recall and f1-score | 87 |
| Table 19 Confusion matrix of a test set using Hu's moment..... | 89 |
| Table 20 Confusion matrix of a test set using proposed methods | 90 |
| Table 21 Results from character recognition | 92 |
| Table 22 Performance comparison with other methods | 99 |

LIST OF FIGURES

| | Page |
|--|-------------|
| Figure 1 License plate of Bhutan: a) Diplomat, b) Army, c) Police, d) Bodyguard, e) Government, f) Private, g) Taxi..... | 3 |
| Figure 2 Sample Bhutanese license plates | 4 |
| Figure 3 Traditional way of checking license plate details in the entry points..... | 8 |
| Figure 4 System overview | 18 |
| Figure 5 The YOLO architecture..... | 19 |
| Figure 6 Overview of YOLO..... | 20 |
| Figure 7 Bounding box coordinates calculation on 4x4 grid cells..... | 21 |
| Figure 8 Illustration of prediction vectors in YOLO | 22 |
| Figure 9 Prediction vectors | 23 |
| Figure 10 Rectlabel interface with the bounding box information | 28 |
| Figure 11 XML file and its normalized bounding box coordinates..... | 29 |
| Figure 12 YOLOv2 configuration files..... | 30 |
| Figure 13 Representation of coordinates | 31 |
| Figure 14 Overall architecture of character segmentation phase..... | 34 |
| Figure 15 Extracted license plates: a) Government, b) Private, c) Taxi..... | 35 |
| Figure 16 Results of license plates after cropping | 35 |
| Figure 17 Results of grayscale conversion | 36 |
| Figure 18 Results of applying iterative bilateral filter | 36 |
| Figure 19 Contrast enhancement using adaptive histogram equalization..... | 37 |
| Figure 20 Different results generated after applying thresholding | 38 |
| Figure 21 a) Binarized image with white background and black foreground, b) Pixel inverted | 39 |
| Figure 22 Pixel connectivity | 40 |
| Figure 23 Left- Binary image, Right- Label mask..... | 40 |
| Figure 24 North and west pixels corresponding to center pixel “p”..... | 41 |

| | |
|--|----|
| Figure 25 Two different labels in the same component..... | 42 |
| Figure 26 Merged blob into a single connected component..... | 42 |
| Figure 27 Bounding box drawn on the segmented license plate characters | 43 |
| Figure 28 Overall architecture of the character recognition | 44 |
| Figure 29 Unwanted characters segmented along with the license plate characters ... | 46 |
| Figure 30 Flow chart of removing unwanted characters | 47 |
| Figure 31 Illustrations of centroids in CD method | 58 |
| Figure 32 Flowchart of horizontal flip method..... | 60 |
| Figure 33 Flowchart of vertical flip method..... | 61 |
| Figure 34 a) Original image, b) Left crop, c) Right crop, d) Horizontal flip of right crop, e) Invert pixels, f) Compare pixels | 62 |
| Figure 35 Original image, b) Left crop, c) Right crop, d) Horizontal flip of right crop, e) Invert pixels, f) Compare pixels | 63 |
| Figure 36 Graph showing validation mAP and average training loss | 66 |
| Figure 37 Experimental results for detection of vehicle and localization of plate | 67 |
| Figure 38 License plate not localized inside vehicle bounding box | 68 |
| Figure 39 Experimental test results for localizing license plate in shade region..... | 69 |
| Figure 40 Experimental test results for localization of license plates under light..... | 69 |
| Figure 41 Experimental results for localization of license plate surrounded by text... | 70 |
| Figure 42 a) Signboard is detected as license plate, b) Sign board is not detected | 71 |
| Figure 43 Experimental results for localizing license plates in night view | 72 |
| Figure 44 Experimental results for localizing license plates block by the tyre and crash guard..... | 73 |
| Figure 45 Experimental results showing incomplete segmentation | 75 |
| Figure 46 Segmented characters for three types of license plates | 75 |
| Figure 47 Experimental results for character segmentation in night view | 76 |
| Figure 48 Experimental results for character segmentation of taxi license plates..... | 77 |
| Figure 49 Example of XGBoosting algorithm..... | 84 |
| Figure 50 Log loss and classification error for Hu's moments..... | 91 |
| Figure 51 Log loss and classification error for proposed methods..... | 91 |

| | |
|--|----|
| Figure 52 Experimental results for character recognition for three types of license plates | 93 |
| Figure 53 Experimental results for character recognition under shadow and light region | 94 |
| Figure 54 Experimental results for character recognition in night view | 95 |
| Figure 55 Experimental results for character recognition in skewed image..... | 96 |
| Figure 56 Experimental results for character recognition in partially blocked license plate..... | 97 |
| Figure 57 Experimental results for wrong recognition of license plate characters..... | 98 |



CHAPTER I

INTRODUCTION

Introduction to the study

This study aims to develop an ANPR (Automatic Number Plate Recognition) system using Bhutanese license plates to replace the conventional way of managing traffic on the roads. This chapter provides a brief background to the study, the statement of the problems and main contributions of the studies. The purpose and scope of the study are discussed in the chronological order.

Background of the study

Bhutan is one of the smallest Asian countries sandwiched between India and China. Bhutan is divided into 20 Dzongkhags (Districts). According to National Statistics Bureau of Bhutan (2017), they have so far recorded a population of 735,553 in which the capital city (Thimphu) accounts for 19.1% (138,736) of the total population. As the number of people increases in the town and due to rural-urban migration, there has been a problem with traffic congestion, job opportunities and human health due to pollution. As a result of an increase in the number of vehicles, the concerned agencies face a problem of monitoring the traffic, inadequate human resources and mismanagement of parking space in the town. Until 1961, because of the lack of paved roads, travel in Bhutan was made by foot or on mule track. With the starting of First Development Plan (1961 - 1966), the first 175 kilometer-long paved road was completed in 1962 linking *Phuntsholing* (nearest border town in India) and *Thimphu* (capital of Bhutan) ("Bhutan-Transportation and Communications," n.d.). Then gradually, the construction of roads in other parts of the district started in the subsequent development plan.

The number of road users has increased drastically all over the world due to the rapid development of social industrialization and urbanization. Furthermore, it has become one of the most critical problems for governments and individuals to adequately supervise and control the orderly, high-speed and safe travel of road

vehicles (Ji & Zhang, 2017). Intelligent Transport System (ITS) is regarded as the best way to guarantee the fast operation of road traffic and ANPR technology is one of the applications of ITS.

ANPR is a mass surveillance method of reading vehicle plates without human intervention through the use of the high-speed camera. In some context, the term ANPR is also known as Automatic License Plate Recognition (ALPR), License Plate Recognition (LPR), Car Plate Recognition (CPR) or Automatic Vehicle Identification (AVI) (Du, Ibrahim, Shehata, & Badawy, 2013). Since its inception in 1941 at the Police Scientific Branch (PSDB) in UK., “the ANPR technology has evolved and adapted to the times, finding new outlets and applications taking it beyond the boundaries of just policing and security” (“History of ANPR,” n.d.).

The Road Safety and Transport Authority (RSTA) acts as a lead agency who looks after the transport sector of Bhutan. Apart from this, it also looks after the enforcement of traffic rules and regulations. The RSTA is under the Ministry of Information and Communication, and its headquarters is located in Thimphu. To make the operation of RSTA more efficient, they have established five regional offices whose mandate is to give better services to the people (See Table 3).

As per the records of RSTA dated May 31, 2019, they have registered 92,162 motor vehicles (BG, BP and BT) excluding two-wheelers in the country in which *Thimphu* and *Phuntsholing* top the list among five regions. In the current situation of the country, it has become crucial to implement ANPR technology to automate the traditional way of managing traffic on the roads. The technology will be helpful to government and private sectors in delivering better services to the road users.

Successful detection and recognition of license plate details entirely depend on the performance of the hardware and software. The functions of hardware components are vehicle detection through sensors and cameras, whereas the software component does the image processing such as image preprocessing, localization, segmentation and recognition (Jin et al., 2012). ANPR is widely used by government and private sectors in the collection of electronic toll fee, monitoring traffic activity, patrolling the border and parking space management. With the rapid development in the areas of artificial intelligence and image processing technology, the detection and

identification of license plates take less than 50ms and videos of 20 frames can be processed in one second (Du et al., 2013).

The typical ANPR can be split into three stages:

1. **License plate localization:** detect the candidate license plate from the image or video frame.
2. **Character segmentation:** extract alphanumeric characters from a detected license plate.
3. **Character recognition:** recognize an individual segmented character.

“ANPR technology tends to be region-specific, owing to plate variation from place to place” (Qureshi, Asif, & Bashir, 2011). Similarly, Bhutanese license plates have different variations in terms of size and colours. The first half of the license plate contains the Bhutanese characters, and the other half contains the actual characters. In Bhutan, there are 7 types of license plates with different size, foreground and background colour, as shown in Figure 1.



Figure 1 License plate of Bhutan: a) Diplomat, b) Army, c) Police, d) Bodyguard, e) Government, f) Private, g) Taxi

Among the seven types of license plates, the study is focused on the three types of license plates since they are widely used in the country. The three license plate samples are shown in Figure 2, and the vehicle numbering scheme with the registration code is given in Table 1.



Figure 2 Sample Bhutanese license plates

Table 1 Bhutanese vehicle number plates

| Registration Code | Region Code | Registration Number Pattern |
|-------------------|---------------|-----------------------------|
| BG | 1, 2, 3, 4, 5 | 0001-9999, A0001-A9999, |
| BP | 1, 2, 3, 4, 5 | B0001-B9999 and so on |
| BT | 1, 2, 3, 4, 5 | |

Source: Road Safety and Transport Authority, 2018

The Bhutanese vehicle number starts with the registration code followed by the region code and ends with 4 numerals. For all the regions, the 4 digits begin with “0001” and end with “9999”. For that particular region, if the 4 numerals are exhausted, then the ending four numbers are prefixed by the English alphabet, as shown in Figure 2. Table 2 shows the types of vehicle ownership, and the description of each license plates with the registration code and Table 3 shows the districts under each region with its code.

Table 2 Vehicle ownership with colour information

| Registration Code | Ownership | Colour Information |
|--------------------------|------------------|---|
| BG | Government | Numbers in yellow with a red background |
| BP | Private | Numbers in white with a bright red background |
| BT | Taxi | Numbers in black with a yellow background |

Source: Road Safety and Transport Authority, 2018

Table 3 Jurisdiction of the regions with region code

| Region Code | Region | Districts under each region |
|--------------------|-----------------|--|
| 1 | Thimphu | Thimphu, Paro, Haa, Punakha, Wangdue Phodrang, Gasa |
| 2 | Phuentsholing | Chukkha, Samtse |
| 3 | Gelephu | Sarpang, Tsirang, Dagana, Zhemgang, Trongsa and Bumthang |
| 4 | Samdrup Jonkhar | Samdrup Jonkhar, Pema Gatshel |
| 5 | Monggar | Monggar, Lhuntse, Trashigang, Trashi Yangtse |

Source: Road Safety and Transport Authority, 2018

This research will be beneficial to the government and private sectors since the developed system can be used to replace the traditional way of monitoring traffic. The system can be used by the private company to manage the incoming and outgoing vehicle from/to their premises. The proposed feature extraction algorithm in this research will help in differentiating those characters having a similar geometrical shape. To the best of our knowledge, there is no research done in the field of ANPR

using Bhutanese License Plates. Hence, it is first of its kind to use the Bhutanese license plates as the datasets to develop the application of license plate recognition system. There are ANPR systems already developed in other countries, but none of the systems works in the Bhutanese context due to different vehicle laws and orders.

Purpose of the study

The main goal of this research is to develop an efficient and robust ANPR system which can be used to replace the traditional way of managing traffics by fulfilling the following purpose:

- The ANPR system is designed to help in recognition of license plate characters, which can be further integrated with the RSTA database for further information processing.
- To develop an algorithm to extract features from the license plate characters for character recognition.

Problem statement

Currently, Bhutan is trying to automate the conventional way (or manual) of providing public services through the use of Information and Communication Technology (ICT) medium whereby improving the living standard of the people. ICT may not be able to replace the basic human needs, but indeed, it can transform the way societies work and communicate with the outer world. ICT is used as a tool to accelerate social growth and economic development in the country. Like public service delivery in government and private sectors through G2C (Government to Citizens) services, the service delivery in the transport sectors in Bhutan is also improved with the introduction of E-ticketing and the display of real-time city bus location in the bus-stop (Tshering, 2017).

The number of traffic violators has increased significantly due to an increase in the number of vehicles (Road Safety and Transport Authority, 2017). Therefore, the development ANPR system is of utmost importance in the current scenario to monitor the traffic violators and remind the road users about the rules and regulations. According to Thimphu Thromde (Municipal Administration), Thimphu City is experiencing increased levels of traffic and congestion due to wide car ownership and

usage. So, to solve the problem of overcrowding, they have built two multi-level car parks, which can accommodate around 500 cars (Thimphu Thromde, n.d.). Therefore, it is an excellent opportunity to use the developed system for parking space management, since it will keep the record of incoming and outgoing vehicles.

In Bhutan, the traffic police personnel physically check the license plate details of every car, whereby it is time-consuming and requires lots of human resources. This way of visual inspection creates more traffic congestion, and it is not feasible to employ an officer who will be doing full-time work in checking the license plate details. Handling a large volume of traffic may be more efficiently done if automated than done by humans. Apart from this, deployment of traffic police personnel on the roads to physically check the details of the license plates might result in the biased checking due to the personal attachment. This research aims to help the government in reducing the costs associated with the operation, increase the processing time and minimise human errors. With the exponential growth in the number of vehicles in the country, now it is high time to automate the management of traffic in the core town areas and various security checkpoints. The proposed system can be used for automatic registration of vehicles at the entry points, which reduces the manual labour in maintaining the register to keep the records. Moreover, the ANPR technology can be used by government and private companies to manage the incoming and outgoing of vehicles in their premises based on the user's identity and type of membership group.

So far, no one has researched on ANPR technology using Bhutanese license plates due to a smaller number of vehicles on the roads and traffic well managed by humans. However, with the development of the country and with the increase in the number of vehicles, it has become essential to develop a system which will help the traffic law enforcement agency to give better services to road users. Moreover, it will also help to reduce journey time and traffic congestion. An ANPR system of one country will not work for another country, because different countries have different vehicle laws which mandate to have its license plates in terms of varying size, character and colour. So, in Bhutan, there are different license plates with different colour and size. Therefore, the development of reliable ANPR for Bhutanese license plates is currently in high demand, which can help to minimize the problems faced by

the current system. Figure 3 shows the traditional way of checking license plates in security checkpoints.

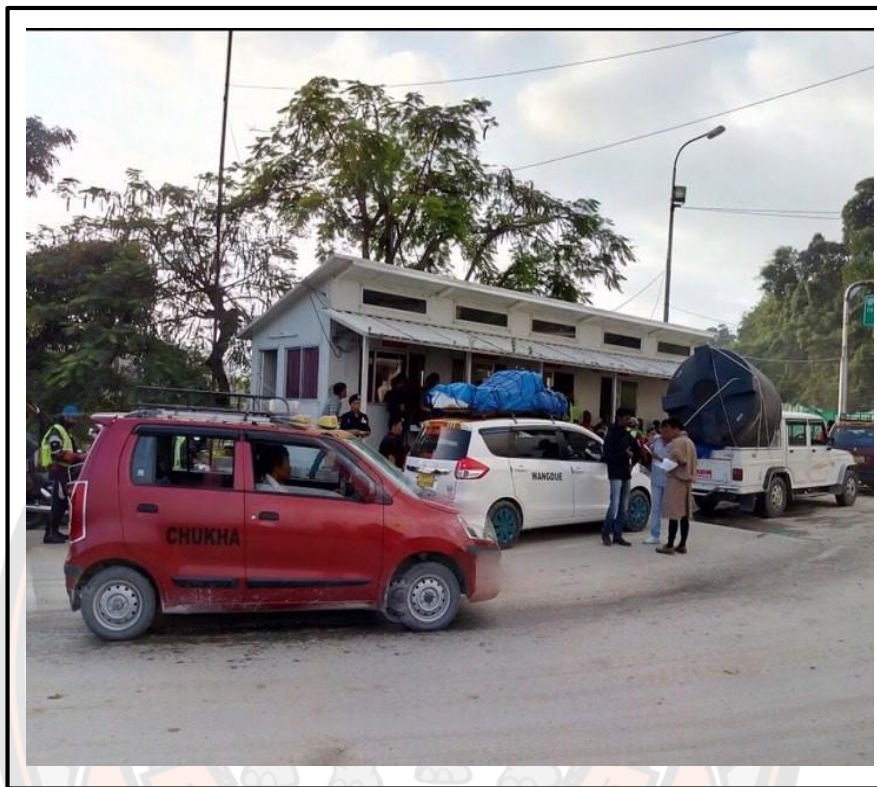


Figure 3 Traditional way of checking license plate details in the entry points

The scope of the study

The scope of the study is to develop an ANPR technology that can identify and recognize the license plates used in Bhutan. All the 20 districts use the same format of license plates. Therefore, the developed system can be used in all the districts. The system expects to detect and recognize the Bhutanese license plates with high accuracy. Although there are seven types of license plates currently used in Bhutan, the study will focus on three types of vehicle number plates since these plates are used widely. The proposed system uses still license plate images instead of real-time videos.

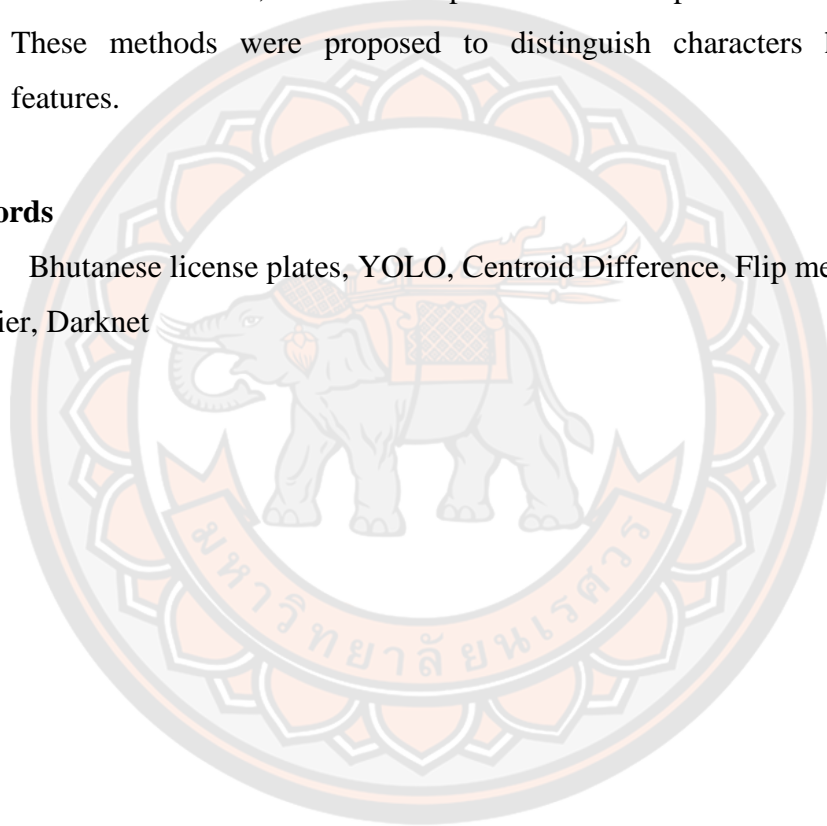
Contribution of the study

The main contribution of the thesis has been listed below:

- A single convolutional neural network (YOLO) was proposed to localize the license plate after detecting the vehicle automatically. This is basically done to reduce the false positive generated by signboard and other objects since they look similar to the license plates.
- For extraction of feature from the characters, three new methods namely centroid difference, horizontal flip and vertical flip methods were proposed. These methods were proposed to distinguish characters having similar features.

Keywords

Bhutanese license plates, YOLO, Centroid Difference, Flip methods, xgboost classifier, Darknet



CHAPTER II

REVIEW OF LITERATURE

Introduction

In this chapter, the overview of the literature review in the field of ANPR is reflected. The literature review of each phase is discussed separately with the limitation of each method. With the advancement in the field of computer vision, many new approaches are introduced in the area of ANPR to reduce the processing time and eliminate false positives (Du et al., 2013). This section provides the approaches used in the four main stages of ANPR: vehicle detection, License plate localization, Character segmentation and Character recognition.

Vehicle detection

Vehicle detection is one of the problems related to object detection, where we intend to find the position of an object in the video frame or image. Automatic vehicle detection from the videos plays a vital role in the development of the self-driving car and intelligent transportation systems (X. Hu et al., 2018). Vehicle analysis in images or video frames plays a crucial role in smart systems such as driver assistance, self-driving car, and to monitor the vehicle. Over the past decade, a lot of researchers has proposed algorithms in the field of efficient detection of moving vehicle. Researcher Vasu (2010) proposed a background subtraction to detect a moving vehicle from the static stored image. The difference in the image generated from the subtraction was used as the vehicle regions after applying thresholding. Deep learning is extensively used in industry and research centers due to high performance. Soin and Chahande (2017) have suggested a deep neural network (DNN) trained on CIFAR-10 (Krizhevsky & Hinton, 2009) data which uses gradient descent with momentum and a learning rate of 0.001. Researchers Khalid, Mazoul, and Ansari (2011) proposed a vehicle detection algorithm using a corner detector and AdaBoost classifier. In their approach, a Shi and Tomasi (Jianbo & Tomasi, 1994) corner detector was used to detect the corners of the cars. After identifying all corners, the AdaBoost classifier was applied to validate the features.

License plate localization

Among three main steps of ANPR, the license plate localization is one of the crucial steps since the accurate detection of a license plate from the car image directly affects the accuracy of the segmentation and recognition phase. Therefore, to solve the problem of license plate localization, several new algorithms were designed to cater to the needs of the different formats of a license plate.

Boundary-based approaches

One of the distinguishing features of a license plate is its rectangular shape with a known aspect ratio. Usually, the width of the license plate is greater than its height. Edge information is one of the methods to find all the possible rectangles from the input image. Zheng, Zhao, and Wang (2005) have suggested an edge detection method using vertical Sobel operator along with the image enhancement to find the vertical edges and remove the weak gradients on the plate area caused by the car shadow. In their approach, the Sobel mask is multiplied with the image to get the vertical gradient. Then, the non-maximum suppression is applied in the horizontal direction to obtain the Sobel map. The overall detection rate of their approach is 99.7%, but the algorithm needs the same dimensions of the input image. Sarfraz, Ahmed, and Ghazi (2003) also suggested a vertical Sobel operator along with the seed filling algorithm to filter out the unwanted objects from the image. The width to height ratio of the license plate is used to match the vertical edges to find the correct license plate regions. The system is tested with excellent quality images, and the overall detection rate from their approach is 96.66%.

Koval, Turchenko, Kochan, Sachenko, and Markowsky (2003) have applied the Hough Transform (HT) method to detect lines in the binary image. In the first phase, the coloured image is converted to a binary image (0-black, 1-white). Then using the HT method, the horizontal and vertical lines are extracted to construct a rectangular region. In the last stage, the aspect ratio of the constructed shape is compared with the actual aspect ratio of the license plate to confirm the correct plate region. According to Duan, Duc, and Du (2004), the HT method can detect straight lines up to $\pm 30^\circ$ inclination, but it consumes much time and memory.

Colour-based approaches

Since most of the countries have license plates with different background and foreground colour, the colour information is used to extract the license plates from the image. They use different colours to differentiate the types of vehicles and the jurisdiction that it belongs to. The drawback of using the colour information to detect the license plate is when there is a similar colour between the license plate and the car body. In most of the approaches, the coloured image is converted into a grey-scale or binary image for license plate localization, but in colour based approach, the colour features are used as a piece of essential information for localization. Ashtari, Nordin, and Fathy (2014) have used colour features to detect a Persian license plate by defining the salient and standard template. The blue colour rectangle that is found on the side of the license plate is selected as the template. Then the algorithm scans the image to locate the templates for detection. The detection rate reported in their research is 96.6%, but license plate needs some unique features to be defined as the template. In the Chinese number plate format, Shi, Zhao, and Shen (2005) proposed that all pixels in the input image were classified using HLS (Hue, Lightness, Saturation) colour model into 13 categorizes to have robustness. After the detection, the plate is verified by comparing its Width to Height Ratio (WHR).

Texture-based approaches

Texture-based approaches are mostly used to check the presence of characters in the license plate, by checking the transition of grey-scale value between the character colour and the license plate background colour (Du et al., 2013). Usually, using texture-based methods to locate the license plates on the motor vehicles need evident character. According to Hong-ke, Fu-hua, Jia-hua, and Huan-sheng (2005), the scan-lines method was used to scan the grey-scale levels in the scan line. When the number of lines meets a specific condition, the area of that scan line is treated as the potential license plate. However, to use this method, the characters on the license plate should be relatively clear. Hsieh, Juan, and Hung (2005) have introduced the Haar scaling function for Wavelet transform to extract the license plate. In their method, they have defined four sub-bands. The sub-band HL and LH

describes the edge information in the vertical and horizontal direction, respectively. The LH image determines the maximum change in the horizontal edges. Whereas, the vertical edges are projected horizontally below the LH image to determine the position based on maximum projection. All these methods have an advantage of locating license plates even if the boundary is deformed. However, these methods are computationally complex, especially when the background is under illumination.

Feature-based approaches

Features-based approaches are mostly concerned with the extraction of features from the license plate regions and using those features to detect the position of the license plate. Jia, Zhang, and He (2007) have proposed region-based license plate detection method. In their approach, the candidate regions were extracted using mean shift segmentation method. After the extraction of candidate regions, the features such as rectangularity, aspect ratio and edge densities were extracted to quantify the license plate regions. The number of morphological operations were performed, which might hamper the overall speed of the system. Prates, Cámara-Chávez, Schwartz, and Menotti (2014) have proposed a Histogram of Oriented Gradient (HOG) using the Sliding Window techniques to detect the Brazilian license plates. In their approach, the image was divided into a small grid and, the HOG was computed. For license plate detection, the fixed-size window from top to bottom with a fixed stride was used to scan the image. In each region, the HOG was computed. Finally, after the extraction of features, the SVM (support vector machine) classifier was used to check the possible license plate region. Using HOG to detect the license plate is still expensive to be deployed in the real-time application.

Character segmentation

After the localization of the license plates, the extracted plates are passed to the segmentation module to isolate the license plate characters from the unwanted objects. Any error made during the segmentation process will directly affect the accuracy of character recognition.

Pixel connectivity-based approach

The connected component analysis (also known as connected component labelling, blob extraction or region labelling) is an application of graph theory used to find the connectivity of “blob” – like regions in a binary image. According to Nukano, Fukumi, and Khalid (2004), the segmentation is performed by labelling all the connected pixels in the binary image. In their study, the segmentation accuracy was 60%. The approach of connected component analysis is easy to implement, but it fails to segment the characters that are joined or broken.

Projection-based approach

The characters used in the license plates differ from country to country. Some countries use English alphabets and Roman numerals, whereas some use their country scripts. In the projection-based approach, it assumes that the character and the background of the license plate have different colours which give different values after binarization. For Saudi Arabian license plate, the vertical projections with pixel counting method in each column were proposed to segment the characters (Sarfraz et al., 2003). The input image was converted to binary from the RGB format, and the vertical projection was applied. Finally, the number of black pixels were counted in each column, and the character segmentation was done based on the transition of a crest to its respective trough.

Whereas for the Vietnamese license plates, Duan, Du, Phuoc, and Hoang (2005) have suggested horizontal and vertical projections to segment the characters. In their approach, the horizontal projection method was first applied to segment the number of rows into two since the license plate characters were organized into two rows. Then the minimum position from the horizontal projection was taken as the starting point in the license plate. Finally, the vertical projection was applied to the

projected row to segment the individual characters. The projection-based approach is independent of character position and can deal with some rotation, but it requires prior knowledge of the number of characters in the license plate (Du et al., 2013).

Character recognition

After the segmentation of the characters, the last stage in the ANPR system is the character recognition phase. Character recognition in the ANPR system faces some difficulties due to camera zoom and unequal dimensions of extracted characters (Miyamoto, Nagano, Tamagawa, Fujita, & Yamamoto, 1991). Therefore, resizing the characters into one dimension helps to overcome this problem.

Template matching approach

Among different recognition approaches, template matching is one of the simple and straightforward method. In the template matching approach, the segmented characters were normalized into a fixed dimension, and the templates were stored in the database. The matching between the segmented character and the template was done by calculating the minimum distance. Some of the methods used to find the minimum distance were hamming distance approach (Sarfraz et al., 2003), Jaccard value (Lee, Kim, & Kim, 1994) and Hausdorff distance (Shuang-tong & Wen-ju, 2005). Template matching is simple to use, but each font requires a template with a fixed size, and it does not work for broken or rotated characters.

Learning-based approach

The learning-based approach uses the machine learning algorithm to recognize the characters based on the features. Patel, Patel, and Brahmabhatt (2013) have suggested zoning and regional features extraction methods. In the zoning feature extraction method, a character was usually divided into a zone of predefined blocks. In each box, the sum of distances of black pixels was divided with the total number, and the box feature was obtained. Whereas in the regional feature extraction method, features like Euler number, Eccentricity, Extent, Orientation, ConvexArea, FilledArea, MajorAxisLength and MinorAxisLength were computed. After the feature extraction, machine learning classifier can be used to recognize the characters.

C.H. Moreno, N. Trejo, M. Soto, and B. M. Montiel (2018) proposed the modelling and characterization of Mexican number plate characters using the technique of Hu's moments, Fourier Descriptors and Cross-correlation factor. In their research, the misclassification of number 6 and 9, M and W, A and V could not be addressed since they relied on the Hu's moments only. The number of characters used in the recognizer was relatively small, which affects the robustness of the system.



CHAPTER III

RESEARCH METHODOLOGY

Introduction

In this chapter, the detailed methodology used in the research is presented. The rationale for using the proposed method is highlighted to fulfill the aim and objectives of the study. This section discusses the steps and the methods to be used in the Vehicle Number Detection and Recognition System in Bhutan. The typical ANPR has three phases: License plate localization, Character segmentation and Character recognition. In this research, the vehicle detection phase is introduced as a first step to extract the vehicle from the image for further processing. Each stage is essential for the next phase since failure to detect the license plate would lead to failure in segmentation and then to the recognition phase.

System overview

The typical overview of the proposed system is shown in Figure 4. The digital camera acquires the input image of the system, and the expected license plate characters are displaced on the vehicle image. In the overview, the vehicle is detected first before localizing the license plate. The detection of a vehicle is done to remove the background noise, which might hamper the localization of the license plate. Apart from the removal of background noise, it is also to reduce the number of false positives generated by signboards, as they look similar to the license plates. After the detection of the car, the license plate localization is done inside the detected vehicle. The extracted license plate is given as the input to the segmentation module for the isolation of individual characters from the unwanted objects. Then the features from the segmented characters are extracted and passed to the character recognizer for the classification of characters. The features from the first license plate character “B” is not extracted since all the three types of license plates contains “B” as the first character.

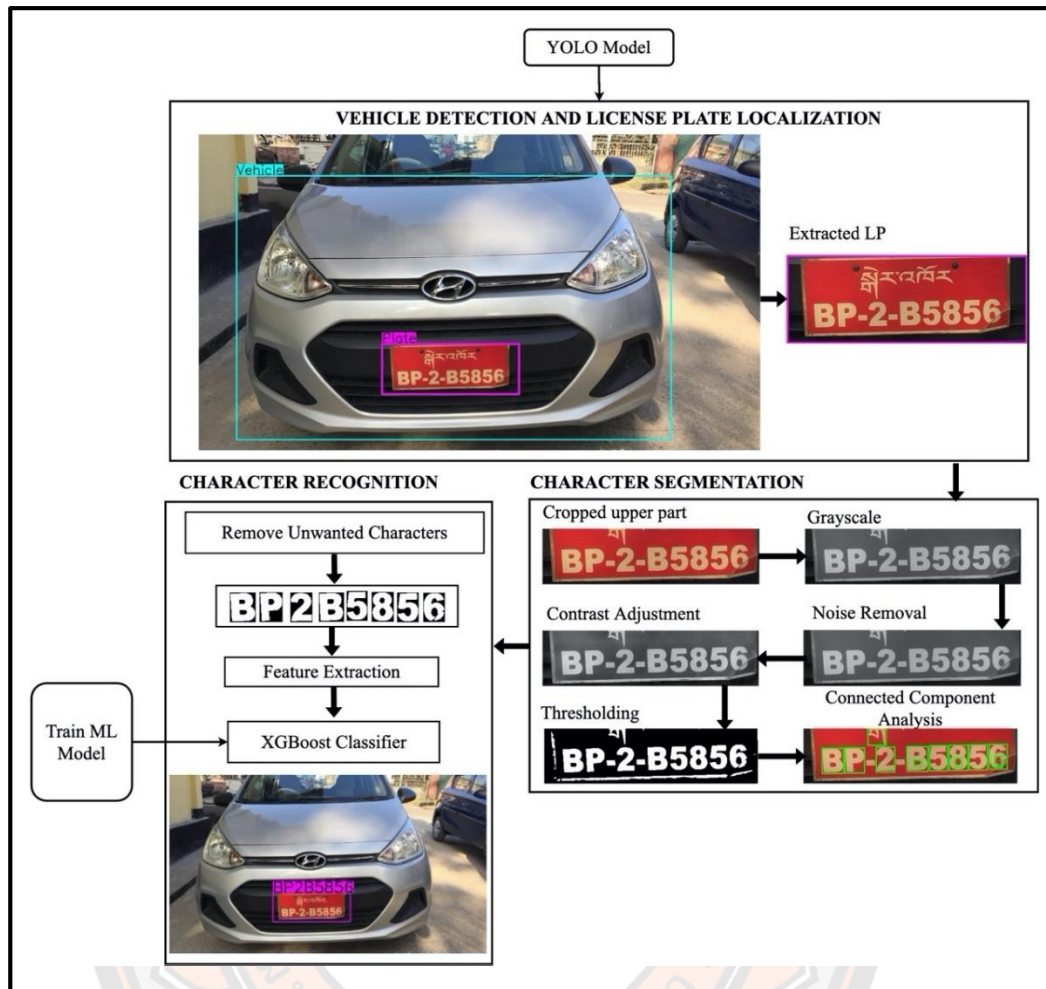


Figure 4 System overview

License plate localization

The accurate localization of a license plate from the image significantly contributes to the precise segmentation and the recognition of characters. The main task of this phase is mostly related to object detection. Usually, the Bhutanese license plates are placed in the front and back of the vehicle. The main objective of this stage is to locate the position of the license plate after detecting the vehicle. The traditional approaches like edge detection, morphological operation and sliding window methods seem to show good results when the license plates are in good quality. With the advancement in computer vision technology, the number of deep learning approaches were proposed. In our proposed system, the study was mainly concerned with YOLO v2 (You Only Look Once) state-of-the-art license plate detector. Before explaining

the YOLOv2, we will go through the background and working principle of the YOLO algorithm.

The concept of YOLO was first coined by Joseph Redmon, Santosh Santosh Divvala, Ross Girshick and Ali Farhadi. According to Joseph Redmon, Divvala, Girshick, and Farhadi (2016), YOLO is “a single convolutional neural network that simultaneously predicts multiple bounding boxes and class probabilities for those boxes”. YOLO gives a bounding box around the object, and it can detect multiple objects at the same time. Unlike R-CNN and Fast R-CNN, which uses region proposal to localize the object, YOLO scans the entire image during the training and testing, whereby, the conceptual information about the classes are encoded. The model is based on the concept of regression where the object detection and classification take place during the single scan of the image. Therefore, this type of algorithm is mostly used in real-time object detection. The network architecture of YOLO is inspired by GoogleNet, which consists of 24 convolutional neural layers followed by two fully-connected layers (Joseph Redmon et al., 2016). The 20 layers are trained on ImageNet and then converted to perform detection by adding 4 layers and two fully connected layers with randomly initialized weights. The typical YOLO architecture and overview of YOLO is shown in Figure 5 and Figure 6.

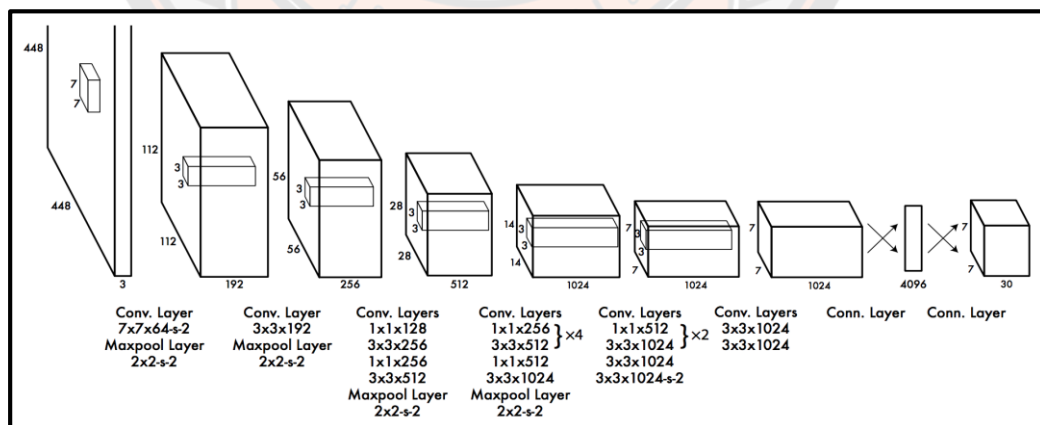


Figure 5 The YOLO architecture

Source: Joseph Redmon et al., 2016

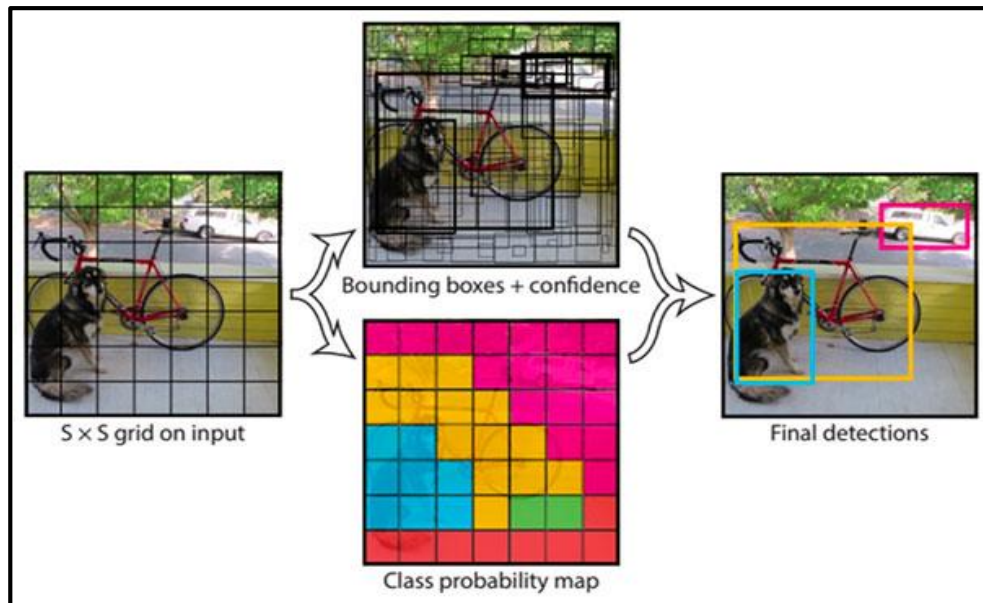


Figure 6 Overview of YOLO

Source: Joseph Redmon et al., 2016

Working of YOLO

The working of YOLO is merely different from other approaches like R-CNN, Faster R-CNN and SGD. For instance, in the sliding window technique, the window is moved across the image and thousands of predictions are generated, which leads to having a slow performance. However, in the YOLO, the input image is scanned once and gets divided into $S \times S$ grid cells in which only one grid cell will be responsible for predicting the object. The grid cell, which contains the center of the object, is responsible for predicting the bounding boxes. The output tensors of YOLO model will be a vector of $S * S * (B * 5 + C)$ where B represents the number of predicted bounding boxes with the confidence score, C is the number of classes.

During the training phase, each grid cell predicts B bounding boxes and the class probabilities. Each bounding box contains 5 components: bx , by , bw , bh and the confidence score. The (bx, by) coordinates represent the center of the object relative to the location of the grid cell and (bw, bh) coordinates represent the width and height of the bounding box corresponding to the image dimensions. These coordinates are normalized between 0 and 1 using Equation 1- 4:

$$bx = \frac{bx - cx}{cx} \quad (1)$$

$$by = \frac{by - cy}{cy} \quad (2)$$

$$bw = \frac{bw}{W} \quad (3)$$

$$bh = \frac{bh}{H} \quad (4)$$

Where (cx, cy) represents the location of the grid cell responsible for the prediction of an object and (W, H) represents the dimensions of an image. Figure 7 shows an example of how the coordinates are represented in YOLO.

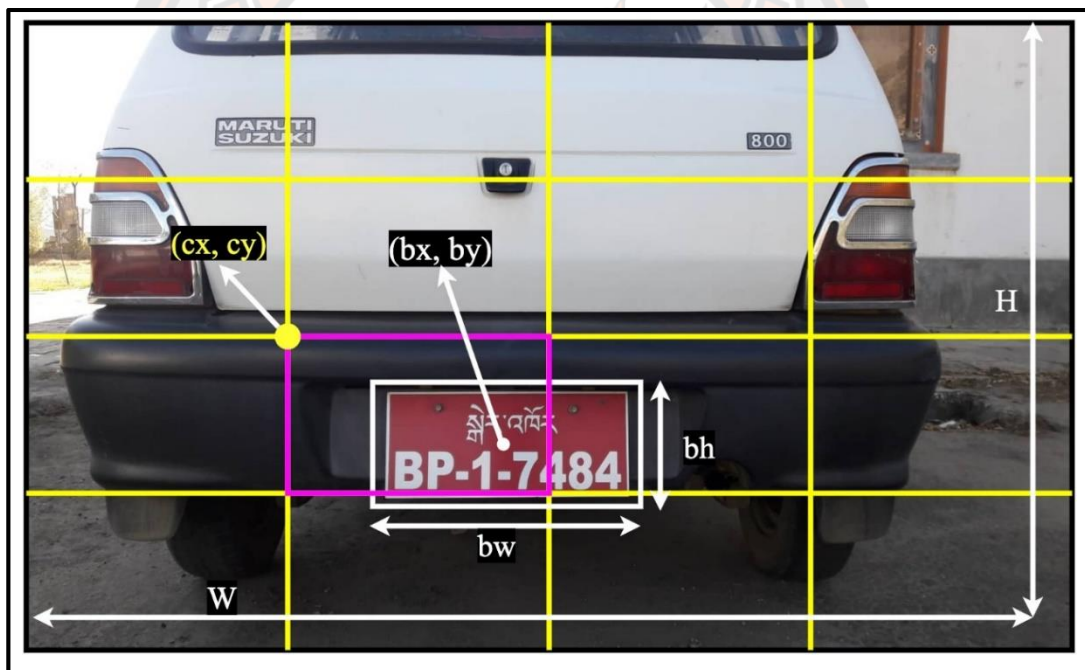


Figure 7 Bounding box coordinates calculation on 4x4 grid cells

In Figure 7, (bx, by) represented the center of the object (white colour bounding box of license plate) and (bw, bh) represented the width and height of the bounding box. The coordinate (cx, cy) represents the starting location of those grid cells (pink) which contains the center of the object.

To understand better about the concept of prediction vectors, let's take an example as shown in Figure 8.

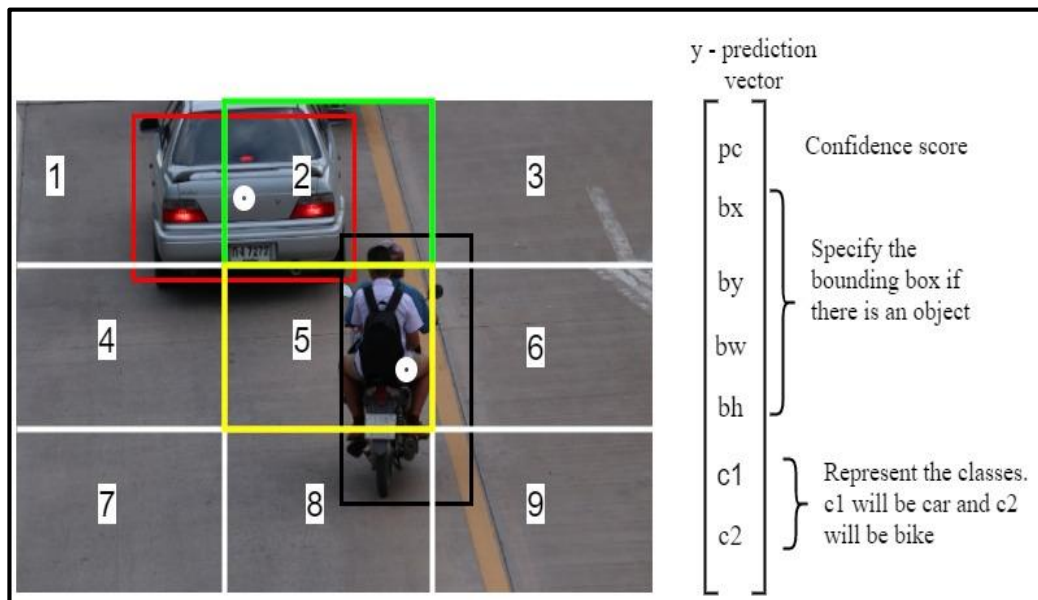


Figure 8 Illustration of prediction vectors in YOLO

In the above Figure 8, the framework divides the image into grids (say 3x3). The image classification and localization are performed in each grid cell. Then YOLO predicts corresponding bounding boxes and class probabilities for each object. For each grid cell, the label y will have a 7-dimensional vector as per the example. In the illustration, the grid cell 2 (green box) and grid cell 5 (yellow box) will be responsible for the prediction of the object (car and bike) since it contains the center of the object (see Figure 9(a-b)). The rest of the grid cells will have a confidence score of 0 since the center of the object does not fall inside those grid cells (see Figure 9(b)). The prediction vector (y) contains the normalized values obtained from Equation 1 – 4.

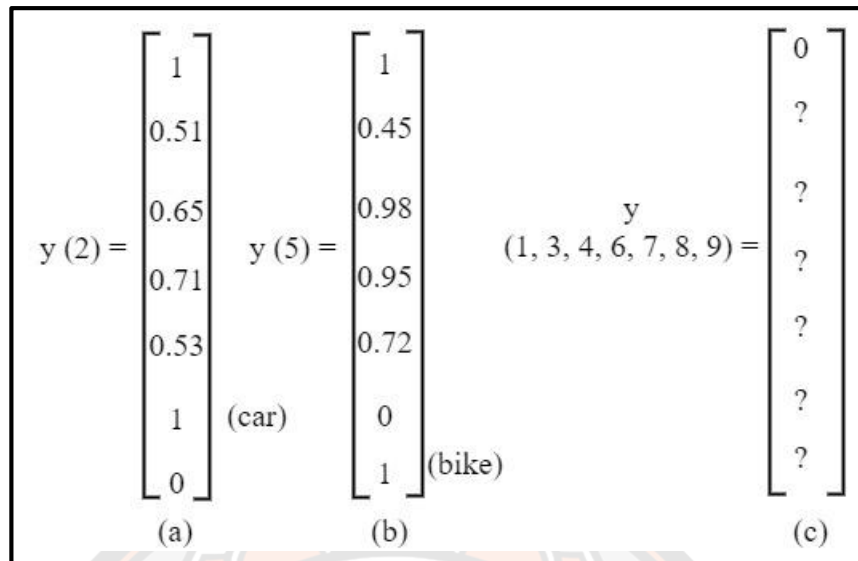


Figure 9 Prediction vectors

Along with the bounding box coordinates, the bounding box also predicts confidence score C . If there is no object in the grid cell, the confidence score is initialized to zero. The object confidence score tells the probability of the presence of an object in that grid cell. This confidence score is determined by the probability of an object that exists in the bounding box, $Pr(Object)$, and the accuracy of bounding box predicted by taking the Intersection Over Union (IOU) between the predicted truth and the ground truth, $IOU(pred, truth)$, as given in Equation 5:

$$Confidence\ Score\ (C) = Pr(Object) * IOU(truth, pred) \quad (5)$$

Where $Pr(Object)$ is the probability that the box contains an object, $IOU(pred, truth)$ represents the intersection over the union of the predicted box and the ground truth.

Since the model predicts many bounding boxes, those bounding boxes with lower object confidence score are removed under the given threshold value (Neubeck & Gool, 2006). Apart from this, each grid cell also predicts the class confidence score C , $Pr(Class_i/Object)$. The class confidence scores are conditional probability and predict one set of class probabilities. During the testing of the model, the class probability is multiplied with the individual box confidence score, which gives the specific class confidence score, as shown in Equation 6:

$$\text{Class Probability} = Pr(\text{Class}_i|\text{Object}) * \text{IOU}(\text{truth}, \text{pred}) \quad (6)$$

Where $Pr(\text{Class}_i|\text{Object})$ is the probability that the object belongs to specific class_i given that an object is present in the cell.

Loss function

YOLO predicts multiple bounding boxes per grid cell. The grid cell, which contains the center of the object is responsible for the prediction of an object. For the selection of the object, the bounding box with the highest IOU with the ground truth is selected. YOLO uses sum-squared error between the predictions and the ground truth to calculate the loss. The loss function is divided into three parts:

- 1. Localization loss:** Localization loss is the difference between the predicted truth and the ground truth as given in Equation 7:

$$\begin{aligned} \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \end{aligned} \quad (7)$$

where

- $\mathbb{1}_{ij}^{\text{obj}} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.
- λ_{coord} increase the weight for the loss on the boundary box coordinates.
- (x_i, y_i, w_i, h_i) - box definition
- S – Grid size

- 2. Confidence loss:** Confidence score is the difference between the 100% confidence that an object is present in the grid cell and the confidence score of the box, as shown in Equation 8:

$$\sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \quad (8)$$

where

- \hat{C}_i is the box confidence score of the box j in cell i
- $\mathbb{1}_{ij}^{obj} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.

If an object is not detected by the grid cell, the confidence loss is calculated by Equation 9:

$$\lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \quad (9)$$

where

- \hat{C}_i is the box confidence score of the box j in cell i
- $\mathbb{1}_{ij}^{noobj}$ is the complement of $\mathbb{1}_{ij}^{obj}$
- λ_{noobj} weights down the loss when detecting background.

3. Classification loss: Classification loss is the difference between the actual class probabilities (1,.. 0,0) and the predicted class probabilities (0.8,..... 0.01, 0.02) as shown in Equation 10:

$$\sum_{i=0}^{S^2} \mathbb{1}_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \quad (10)$$

where

- $\mathbb{1}_i^{obj} = 1$ if an object appears in cell i , otherwise 0.
- $\hat{p}_i(c)$ denotes the conditional class probability for c in cell i .

YOLO version 2

In our research, the YOLOv2 technique was used for the detection of a vehicle and Bhutanese license plate from the image. YOLOv2, also known as YOLO9000, is an improved version of the original YOLOv1 network. According to Joseph Redmon and Farhadi (2017), the first improvement was that the batch normalization was added to all the convolutional layers, whereby the converging rate is significantly improved, and overfitting is avoided. The second improvement was

with the introduction of high-resolution classifiers. In the original YOLO model, the classifier was trained at 224 x 224 resolution and increased to 448 x 448, but in YOLOv2, the model was first fine-tuned at 448 x 448 resolution for 10 epochs. This fine-tuning was done to give time to the networks to adjust its filters to work better in high-resolution images. The third improvement was with the introduction of the anchor boxes, which helps the model to predict the bounding box and use k-means clustering to define the dimensions of the input image (Yadav, 2017). Since YOLOv1 suffers from detecting smaller objects in the image, the fourth improvement brought in YOLOv1 was, the YOLOv2 predicts detection on a 13 by 13 feature map, which is smaller than YOLOv1. By doing this, it helps in identifying the smaller objects with the larger one in an effective way (Sonawane, 2018).

The YOLOv2 uses Darknet-19 architecture with 19 convolutional layers and 5 max-pooling layers with a softmax activation layers to classify the objects (Joseph Redmon & Farhadi, 2017). In Darknet-19 network architecture, the fully connected layers have been removed from the original YOLO architecture. The proposed YOLOv2 network description based on Darknet-19 architecture is given in Table 4. Darknet is a neural network framework written in C language and CUDA, and it supports both the CPU and GPU computation (J. Redmon, 2013). YOLOv2 uses 416 by 416 input size with a reduction factor of 32, which gives 13 by 13 grids cells. To make prediction more robust, we have increased the grid cell to 15 by 15 by increasing the input size to 480 by 480. Moreover, increasing the grid cell helps the model to predict smaller objects

Table 4 Darknet-19 framework

| Layers | Name | Filters | Size | Stride | Padding | Input Channel | Output Channel | Channel |
|--------|------|---------|------|--------|---------|---------------|----------------|---------|
| 0 | conv | 32 | 3 | 1 | 1 | 480 | 3 | 480 |
| 1 | max | | 2 | 2 | | 480 | 32 | 240 |
| 2 | conv | 64 | 3 | 1 | 1 | 240 | 32 | 240 |
| 3 | max | | 2 | 2 | | 240 | 64 | 120 |
| 4 | conv | 128 | 3 | 1 | 1 | 120 | 64 | 120 |
| 5 | conv | 64 | 1 | 1 | | 120 | 128 | 120 |
| 6 | conv | 128 | 3 | 1 | 1 | 120 | 64 | 120 |
| 7 | max | | 2 | 2 | | 120 | 128 | 60 |
| 8 | conv | 256 | 3 | 1 | 1 | 60 | 128 | 60 |
| 9 | conv | 128 | 1 | 1 | | 60 | 256 | 60 |
| 10 | conv | 256 | 3 | 1 | 1 | 60 | 128 | 60 |
| 11 | max | | 2 | 2 | | 60 | 256 | 30 |
| 12 | conv | 512 | 3 | 1 | 1 | 30 | 256 | 30 |
| 13 | conv | 256 | 1 | 1 | | 30 | 512 | 30 |
| 14 | conv | 512 | 3 | 1 | 1 | 30 | 256 | 30 |
| 15 | conv | 256 | 1 | 1 | | 30 | 512 | 30 |
| 16 | conv | 512 | 3 | 1 | 1 | 30 | 256 | 30 |
| 17 | max | | 2 | 2 | | 30 | 512 | 15 |
| 18 | conv | 1024 | 3 | 1 | 1 | 15 | 512 | 15 |
| 19 | conv | 512 | 1 | 1 | | 15 | 1024 | 15 |
| 20 | conv | 1024 | 3 | 1 | 1 | 15 | 512 | 15 |
| 21 | conv | 512 | 1 | 1 | | 15 | 1024 | 15 |
| 22 | conv | 1024 | 3 | 1 | 1 | 15 | 512 | 15 |
| 23 | conv | 1024 | 3 | 1 | 1 | 15 | 1024 | 15 |
| 24 | conv | 1024 | 3 | 1 | 1 | 15 | 1024 | 15 |

Data annotation

Firstly, around 1290 license plate images were collected from in and around the country for the research. Then the datasets were annotated using data annotation software before using YOLO technique to train the images. Before annotating the datasets, each vehicle image should have a common extension such as “JPG” or “PNG”. For annotation, there are numbers of data annotation software available on the Internet; both open source and paid version. In the study, the *RectLabel* software was used to annotate the license plates since YOLO needs a

ground truth for training. The software is easy to use, and it has the features to convert the XML files into YOLO format. The *RectLabel* is an image annotation tool used to label images for bounding box object detection and segmentation (Kawamura, n.d.). Figure 10 shows the interface and the bounding box drawn manually on the vehicle and license plate.

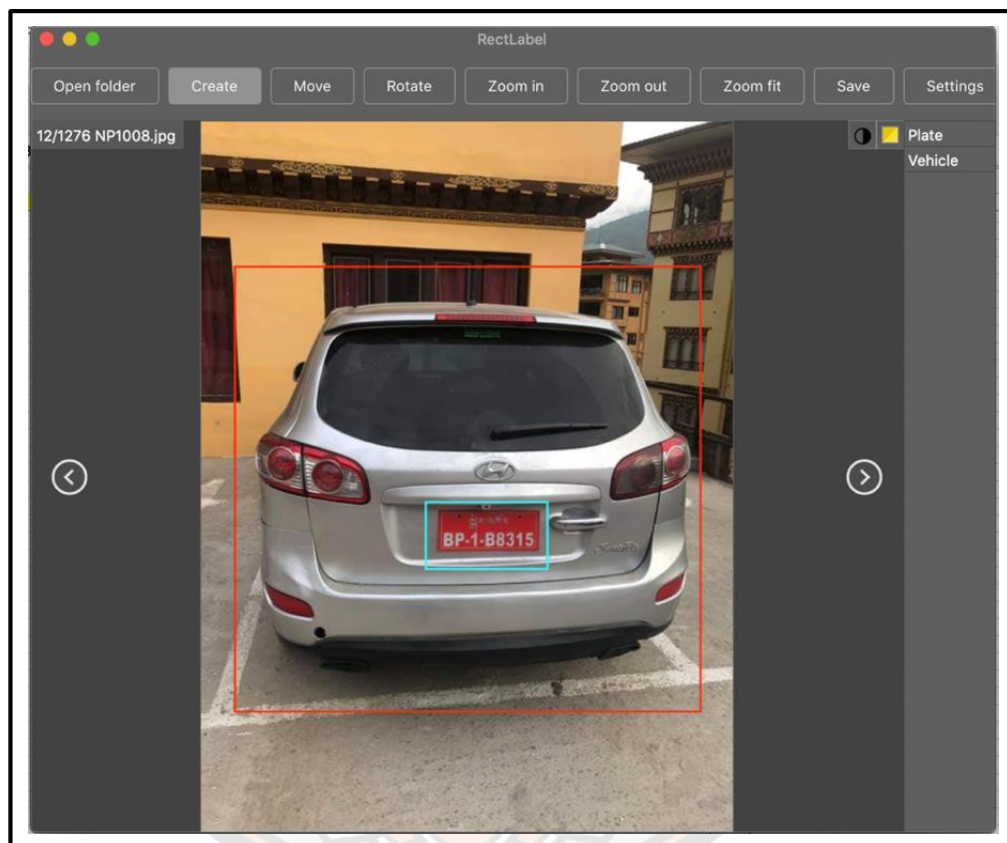


Figure 10 Rectlabel interface with the bounding box information

After the data annotation, the software generates an XML (Extensible Markup Language) file containing the bounding box information for the vehicle and license plates, as shown in Figure 11. The *RectLabel* tool gives the actual starting coordinates (x_{\min} , y_{\min}) and the ending coordinates (x_{\max} , y_{\max}) for each bounding box. The bounding box offsets need to be normalized between 0 and 1 to fit into YOLO format (x , y , w , h) using Equation 11-14:


$$x = \frac{x_{min} + x_{max}}{2 * W} \quad (11)$$

$$y = \frac{y_{min} + y_{max}}{2 * H} \quad (12)$$

$$w = \frac{x_{max} - x_{min}}{W} \quad (13)$$

$$h = \frac{y_{min} - y_{max}}{H} \quad (14)$$

Bounding box drawn using
RectLabel



XML file generated by
RectLabel

```

1 <annotation>
2 <folder>Ann</folder>
3 <filename>NP1008.jpg</filename>
4 <size>
5 <width>720</width>
6 <height>960</height>
7 <depth>3</depth>
8 </size>
9 <segmented>0</segmented>
10 <object>
11 <name>Plate</name>
12 <pose>Unspecified</pose>
13 <truncated>0</truncated>
14 <occluded>0</occluded>
15 <difficult>0</difficult>
16 <bndbox>
17 <xmin>305</xmin>
18 <ymin>515</ymin>
19 <xmax>469</xmax>
20 <ymax>605</ymax>
21 </bndbox>
22 </object>
23 <object>
24 <name>Vehicle</name>
25 <pose>Unspecified</pose>
26 <truncated>0</truncated>
27 <occluded>0</occluded>
28 <difficult>0</difficult>
29 <bndbox>
30 <xmin>48</xmin>
31 <ymin>198</ymin>
32 <xmax>675</xmax>
33 <ymax>797</ymax>
34 </bndbox>
35 </object>
36 </annotation>

```

YOLO Format

| | category number | object center in X | object center in Y | object width in X | object height in Y |
|---|-----------------|--------------------|--------------------|-------------------|--------------------|
| 1 | 0 | 0.537500 | 0.583333 | 0.229167 | 0.094792 |
| 2 | 1 | 0.502778 | 0.518750 | 0.872222 | 0.625000 |

Figure 11 XML file and its normalized bounding box coordinates

After normalizing the bounding box coordinates, the datasets were split into training and testing sets. In the research, 80-20% train-test split was used to divide the datasets into training and test sets. After splitting the datasets, the Darknet-19 framework was used as the base of YOLOv2 for the training. Before training the model, there are some configuration files inside the Darknet folder that needs to be modified to fit the requirements. The full pictorial representation of the modification of the configuration file is shown in Figure 12.

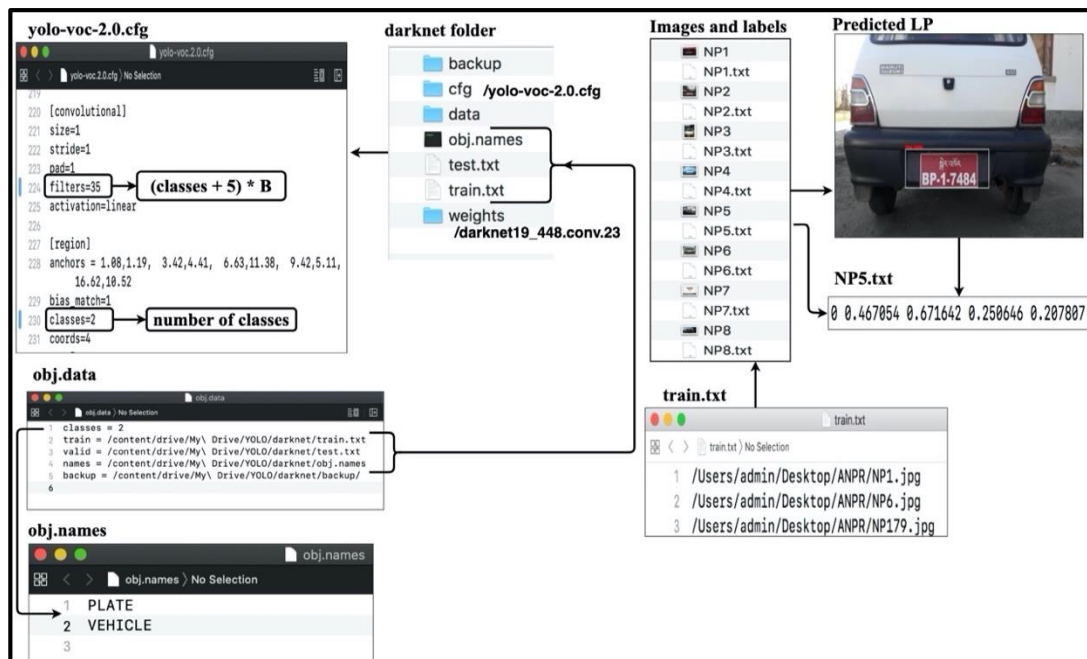


Figure 12 YOLOv2 configuration files

To run the above configuration files, the following Darknet command was executed to start the training process:

```
./darknet detector train "obj.data" "cfg/yolo-voc-2.0.cfg"
"weights/darknet53.conv.74"
```

“./darknet detector train” is a Darknet command used to train the YOLO model in the Darknet framework. The “yolo-voc-2.0.cfg” file contains the actual convolutional layers of YOLOv2 in which the last layers (filters and classes) attribute needs to be modified to match the requirement. Each grid cell in YOLOv2 predicts 5 bounding boxes ($A = 5$) having 5 components (bx, by, bw, bh, pc). Therefore, the formula to find the number of filters in the YOLOv2 is given Equation 15:

$$filters = (Classes + 5) * A \quad (15)$$

Thus, the number of filters is set to 35, and the number of classes was set to two. In the “obj.names” file, the class labels were defined, and in the “obj.data” file, the path to train set, test set, class labels and the backup directory was defined. Both the image datasets and the corresponding annotations were stored in one folder.

The “*train.txt*” and “*test.txt*” files contain the path of the datasets in which they are stored. To make the training easier, Joseph Redmon and Farhadi (2017) offer a pre-trained weight trained on ImageNet. The advantage of using pre-trained weights for training the new model is, it is faster and often end up with better models giving higher accuracy. The pre-trained weights were stored in the “*weights*” folder.

Algorithm for localization of license plate inside the vehicle

In most of the typical ANPR technology, the localization of the license plate is done without checking the presence of the vehicle. They directly try to localize the license plate from the image or video stream without checking the presence of the vehicle. Therefore, it is essential to check the presence of the vehicle to reduce the number of false positives generated by the objects having similar characteristics with the license plates. Jamtsho, Riyamongkol, and Waranusast (2019b) have proposed an automatic detection of license plate after detecting the vehicle using a single convolutional neural work to reduce the false positives. Figure 13 represents the coordinates for vehicle and license plate used in the algorithm.

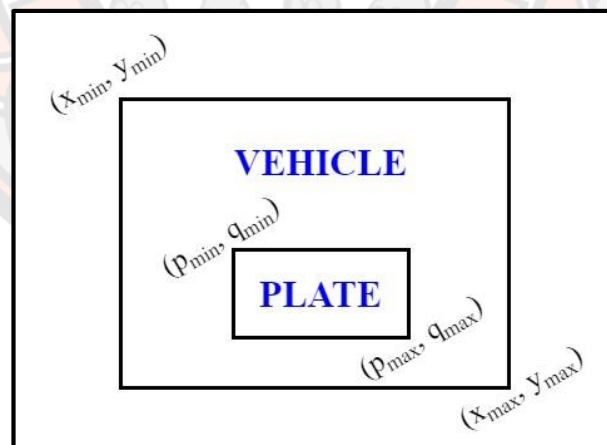


Figure 13 Representation of coordinates

In Figure 13, the coordinates represent:

- x_{min}, y_{min} = starting coordinates of the VEHICLE
- x_{max}, y_{max} = ending coordinates of the VEHICLE
- p_{min}, q_{min} = starting coordinates of the PLATE
- p_{max}, q_{max} = ending coordinates of the PLATE

The YOLO algorithm predicts center ($centerX, centerY$) coordinates of the bounding box followed by the width (w) and height (h) of the box. Therefore, Equation 16 and 17 was used to derive the top-left coordinates (x_{min}, y_{min}) of the bounding box.

$$x_{min} = centerX - (w/2) \quad (16)$$

$$y_{min} = centerY - (h/2) \quad (17)$$

After getting the starting coordinates, the ending coordinates of the bounding box was calculated using Equation 18 and 19:

$$x_{max} = x_{min} + w \quad (18)$$

$$y_{max} = y_{min} + h \quad (19)$$

The pseudo-code proposed for automatic localization of license plate inside the vehicle image is shown in Algorithm 1 (Jamtsho et al., 2019b):

Algorithm 1: Automatic license plate localization from the vehicle

Input: All detected bounding box coordinates (P and V)

Output: Bounding box of detected VEHICLE having localized PLATE

N ← Length(P)

M ← Length (V)

For i ← 0 to N **do**

For j ← 0 to M **do**

$p_{min} \leftarrow P[i][0];$ $q_{min} \leftarrow P[i][1];$

$p_{max} \leftarrow P[i][2];$ $q_{max} \leftarrow P[i][3];$

$x_{min} \leftarrow V[j][0];$ $y_{min} \leftarrow V[j][1];$

$x_{max} \leftarrow V[j][2];$ $y_{max} \leftarrow V[j][3];$

```

If ( $p_{\min} > x_{\min}$  &&  $q_{\min} > y_{\min}$ ) && ( $p_{\max} < x_{\max}$  &&  $q_{\max} < y_{\max}$ ) then
    // Get the bounding box of a PLATE
    PLATE  $\leftarrow$  ( $p_{\min}$ ,  $q_{\min}$ ,  $p_{\max}$ ,  $q_{\max}$ )
End if
End for
End for

```

Train model in Google Colaboratory

For training the YOLO model using Darknet framework, there are some prerequisites like powerful GPU (Graphical Processing Unit), Nvidia CUDA, cuDNN and OpenCV. Training of any neural networks requires high GPU to perform the training in an effective and faster way. GPU is a single-chip processor which is used extensively for Graphical and Mathematical Computation. The Google Colaboratory is a free cloud-based service and provides Tesla K80 GPU with 12GB RAM for developing deep learning applications. The advantage of using Google Colaboratory is; first, it is an open-source cloud GPU, and secondly, it has all the pre-configured prerequisites needed by the Darknet framework. Due to hardware constraints in the research, Google Colaboratory was utilized for training the YOLO model. The modified version of Darknet based on Alexey's implementation (2016) was selected since some logs generated after each epoch have been removed; keeping only the final results after each iteration. The logs generated by the original Darknet creates a long queue during the training, whereby increasing the training time.

Character segmentation

The localized license was then given to the segmentation phase to isolate the characters from the background. The correct segmentation of the actual license plate characters entirely depends on the localization of the license plates. The obtained license plate was preprocessed to have better segmentation of the characters from the license plate before using connected component analysis. Since Bhutanese license plates have different variations in the foreground and background colours, the “*White Pixels Counting with Inversion*” method was proposed to handle the different license plates. The overall architecture of the character segmentation is shown in Figure 14.

The list of preprocessing steps applied to the license plates is shown discussed in the subsequent subsections.

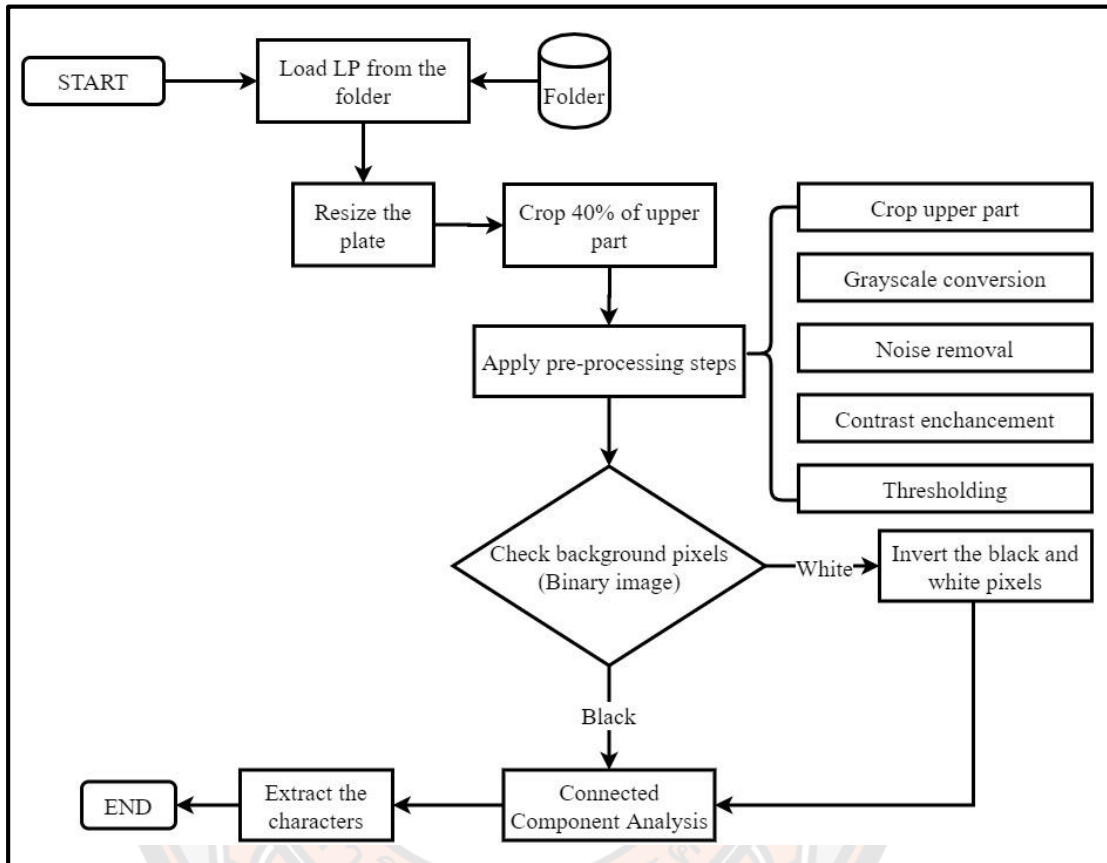


Figure 14 Overall architecture of character segmentation phase

Crop upper part of the license plate

Before cropping the upper portion of the license plate, the localized license plates were resized into fixed width and height to have common dimensions. After resizing the license plates, around 40% (upper part) of the actual height was cropped to remove the Bhutanese characters since it does not play any significant role in the identification of the plates. The 40% threshold value was chosen since the first half portion of the license plate contains the Bhutanese scripts. Apart from removing the Bhutanese scripts, it also helps in removing the bolts and screws. The original extracted license plate and the cropped license plate is shown in Figure 15 and 16, respectively.



Figure 15 Extracted license plates: a) Government, b) Private, c) Taxi

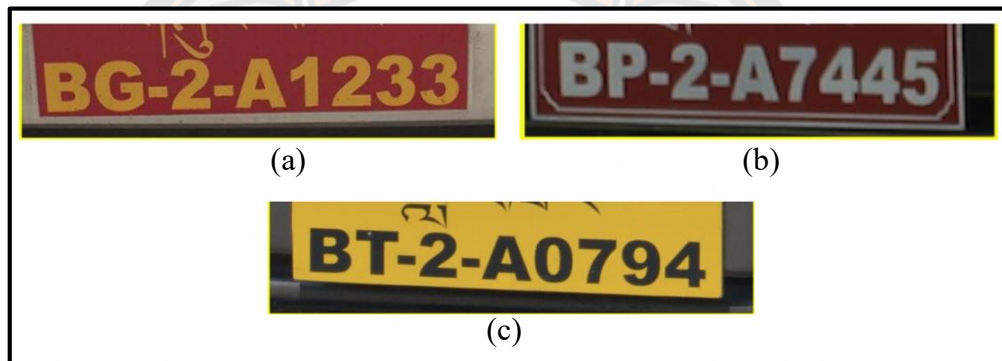


Figure 16 Results of license plates after cropping

Grayscale conversion

Since the colour information of the license plates is not taken care during the segmentation, the cropped license plates were converted to grayscale format from the RGB (red, green, blue) colour format. The grayscale images are the shades of grey whose value ranges from 0 to 255 where 0 is considered as the black pixels, and 255 is taken as the white pixels. An RGB coloured image contains 3 channels which are 3-dimensional matrix whereas the grayscale image is of a 2-dimensional matrix. By converting the image into grayscale, the noise generated by the camera can be removed easily in the subsequent steps and, the processing time is reduced. Figure 17 shows the grayscale operation performed on the license plate.



Figure 17 Results of grayscale conversion

Noise removal by an iterative bilateral filter

In the third preprocessing step, the noise from the image was reduced by using an iterative bilateral filter. The noise in the picture is usually produced by the variation of the brightness or colour information. Iterative bilateral filtering is chosen over the median filter because it provides a mechanism to reduce the noise whereby the edges of the images are still preserved (Kaur & Kaur, 2014). In this method, the intensity value of each pixel is replaced by a weighted average of intensity from the nearby pixels. Figure 18 shows the results of applying the iterative bilateral filter on the grayscale image.



Figure 18 Results of applying iterative bilateral filter

Contrast enhancement using adaptive histogram equalization

In the fourth preprocessing step, the contrast of the image was enhanced using Adaptive Histogram Equalization (AHE). AHE is an image enhancement method designed for natural and medical images with over-brightness (Pizer et al., 1987). AHE shows better contrast than the histogram equalization (HE). Histogram equalization is good when the histogram of the image is confined to a particular region, but it won't work when there is a large variation in intensities. Therefore, AHE was used in the preprocessing steps since some license plates will have considerable variation in the intensities and also to have robustness during the segmentation phase. Figure 19 shows the license plates after applying adaptive histogram equalization on the license plate.



Figure 19 Contrast enhancement using adaptive histogram equalization

Otsu's thresholding

In the final stage of the image preprocessing, the contrast-enhanced image was converted to binary format using Otsu's thresholding method. The image binarization is done mainly to isolate the interested objects from the background. Otsu's method is not only important to find the optimal threshold value but also crucial for unsupervised learning of pattern recognition application (Otsu, 1979). In a simple thresholding method, the threshold value (T) must be supplied manually, which requires to have a knowledge about the lighting conditions. In the real scenario, Otsu's thresholding is used since it automatically calculates the threshold value from the image histogram for a bimodal image. Bimodal image is an image whose histogram has two values. Otsu's method approximates the middle value from those

two peaks as the threshold value (T). Otsu's thresholding method iterates through all the possible threshold value and calculate the measure of spread for each pixel level. The aim is to find an optimal value whose sum of foreground and background is minimum. Figure 20 shows the different outputs generated by Otsu's thresholding operation.

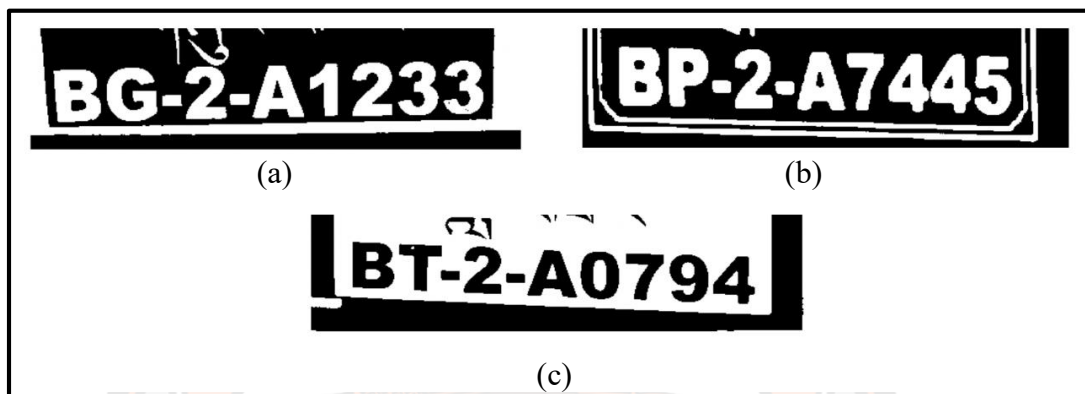


Figure 20 Different results generated after applying thresholding

White pixel counting with the inversion method

Due to the disparity of the background and foreground colours used in the Bhutanese license plates, there were different output generated after the thresholding of the license plates, as shown in Figure 20. These types of disparity will affect the segmentation process due to different thresholding outputs. Therefore, to solve this problem, the binarized license plates will have to be first checked for the presence of black background and white foreground pixels. The BG (Government) and BP (Private) license plates give a required binary image format (white foreground with black background) Still, in the case of BT (Taxi) license plate, it gives the opposite binary format, as shown in Figure 21a. This output does not conform to the required binary image, and it will create problems while finding the connected components. Therefore, a *“white pixel counting with inversion method”* was proposed to get the desired binarized output (Refer to Figure 21b).



Figure 21 a) Binarized image with white background and black foreground, b) Pixel inverted

In the white pixel counting method with the inversion method, the preprocessed license plate was first converted to a binary image using Otsu's thresholding operation, as shown in Figure 21a. Then the pixel counting method counts the number of non-zero pixels (white pixels) in the binarized image and checks with the threshold value. The white pixels from the binary image was divided by the total number of pixels and, then it was converted to a percentage. If the white pixels in the license plate is more than 40%, then a BITWISE NOT operation swap the pixel values from black to white and vice versa, as shown in Figure 21b.

Connected component analysis (CCA)

After getting the desired binary image from the preprocessing step, the binarized license plate was passed to the connected component analysis for the extraction of the blob from the image. The connected component analysis is also known by the name connected component labelling. The classical connected component analysis uses a union-find data structure and the graph theory to analyze connected components in an image (Rosenfeld & Pfaltz, 1966). The connected component analysis can be only applied to binary or thresholded images. A connected component labelling algorithm finds all the associated components in an image and assigns unique labels to all the points in the same component based on pixel connectivity. The pixel connectivity can be either 4-connected or 8-connected, as shown in Figure 22 and the example of CCA is shown in Figure 23.

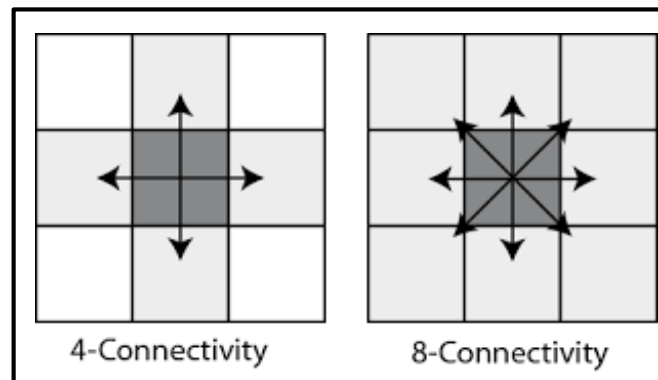


Figure 22 Pixel connectivity

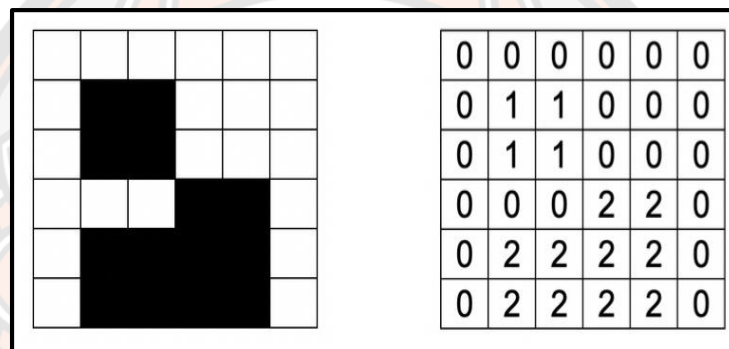


Figure 23 Left- Binary image, Right- Label mask

Working with CCA

The connected component analysis uses pixel connectivity: 4-connectivity and 8-connectivity to find the best-connected component in the binary image. For the CCA method, the binary image should have a white foreground and black background. The license plate characters should have white pixels. The connected component labelling algorithm consists of two passes. In the first pass, the algorithm loops through individual pixels " p ", and the west and north pixel are checked. This type of check is called 4 connectivity. 8-connectivity can also be performed by checking the west, north-west, north and north-east pixels. Then in the second pass, the algorithm loops over the labels generated by the first pass and merges them together that are in the same component.

1. The first pass

In this pass, every pixel in the cell is checked. For simplicity of explaining the concept, we will take 4-connectivity and check west and north pixels with respect to pixel “p”.

1.1. Step 1

In the first step, we check whether we are interested in pixel “p” or not. If the pixel is zero (background), we ignore it and move to the next pixel. If it is a foreground pixel, we proceed to step 2 and 3.

1.2. Step 2 and 3

In this step, if the pixel “p” is a foreground pixel, then we grab the north and west pixels, as denoted by W and N (Figure 24):

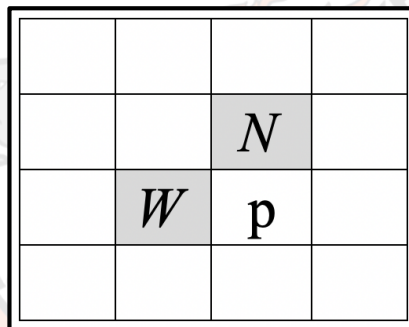


Figure 24 North and west pixels corresponding to center pixel “p”

There are two possible situations:

1. Both pixels are background (black): No labels are associated with these pixels. So, create a new label and store the label value in N and W.
2. North or/and west pixels are not a background pixel: Since one of the pixels have a label associated with it, we can proceed to Step 4 and 5.

1.3. Step 4 and 5

In this step, the center pixel “p” gets the minimum of the label value: $p = \min(W, N)$.

1.4. Step 6

In Figure 25, the north and west pixels have values X and Y , respectively. Even though these two pixels have different labels, we know that these two

components are connected by the current pixel “*p*”. To indicate that these two labels are part of a connected component, we can set the label X, as a child of Y by using the union-find data structure. It means we can store information about X and Y labels saying that they are the same even though they have different labels.

| | | | | | |
|--|----------|----------|----------|----------|----------|
| | | | | | |
| | <i>Y</i> | | | <i>X</i> | <i>X</i> |
| | <i>Y</i> | <i>Y</i> | <i>Y</i> | <i>P</i> | |
| | | | | | |

Figure 25 Two different labels in the same component

1.5. Step 7

Proceed to the next pixel and repeat with Step 1.

2. The Second Pass

In the second pass step, the algorithm visits every pixel and check the label of the current pixel. If the current label is “root”, we proceed to the next pixel. Otherwise, the tree is traversed until we find the root in the structure. After reaching the root node, we assign the value to the current pixels, as shown in Figure 26.

| | | | | | |
|--|----------|----------|----------|----------|----------|
| | | | | | |
| | <i>Y</i> | | | <i>Y</i> | <i>Y</i> |
| | <i>Y</i> | <i>Y</i> | <i>Y</i> | <i>P</i> | |
| | | | | | |

Figure 26 Merged blob into a single connected component

Once the “large connected components” have been extracted using the 8-connectivity from the binary image, the contour properties (aspect ratio and area) was used to quantify the license plate characters and to get rid of unwanted objects. The segmented characters are those bounding boxes drawn in the green colour. The unwanted characters segmented along with the license plate characters were removed during the preprocessing step of recognition phase. Figure 27 shows the results of character segmentation.

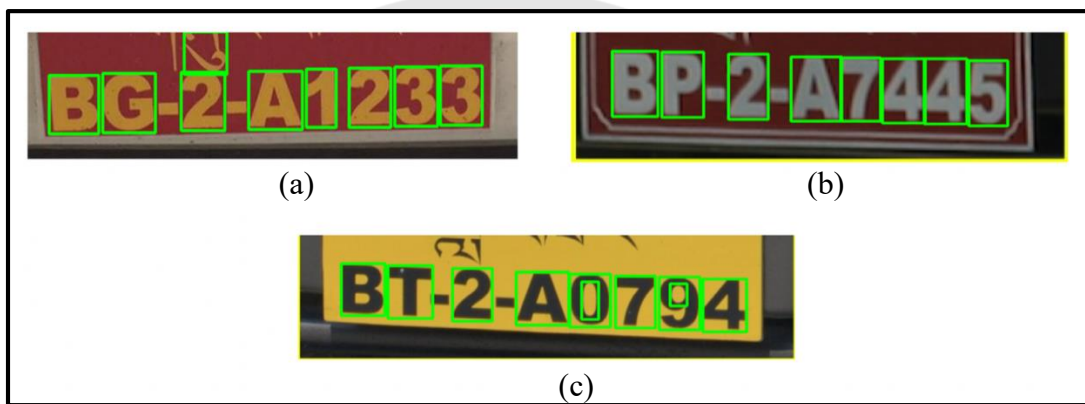


Figure 27 Bounding box drawn on the segmented license plate characters

Character recognition

The last phase in the automatic number plate recognition system is the character recognition phase. The accurate recognition of the license plate characters entirely depends on the correct segmentation of the characters in the previous phase. The overall flow chart of character recognition is shown in Figure 28.

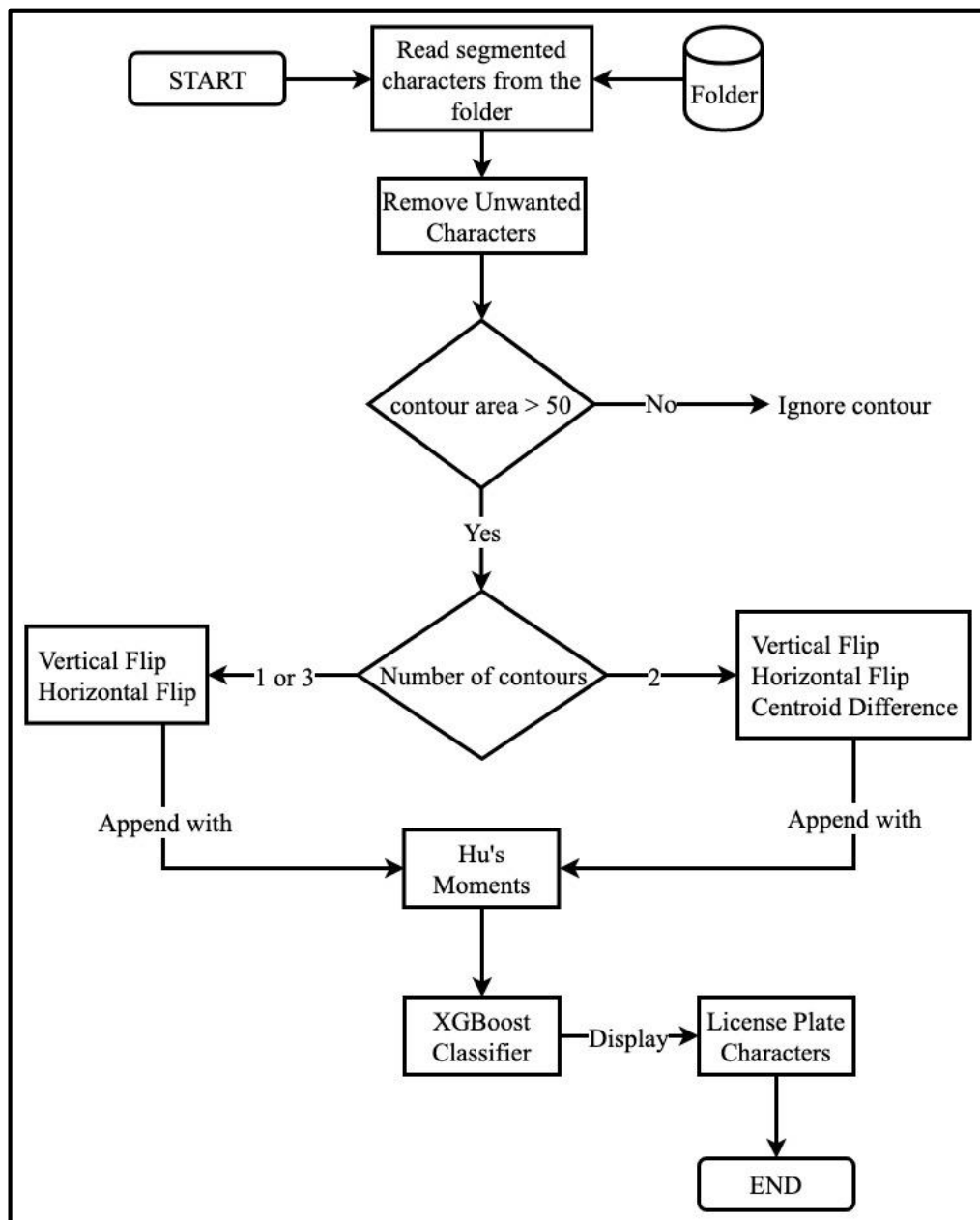


Figure 28 Overall architecture of the character recognition

In the first phase, the segmented characters from the previous step were loaded to the system, and the unwanted characters (noise) were eliminated using a proposed method given in Algorithm 2. After removing the unwanted characters, the contour area was checked with a threshold value to get rid of small unwanted contours. The presence of unwanted contours inside the characters hampers the extraction of features. The unwanted holes (contours) are usually caused by the mud or bolt on the license plate characters. After filtering out the unwanted contours, the number contours in each character were computed to select the correct proposed methods for feature extraction. After selecting the desired feature extraction method, the extracted features were appended with Hu's moments and, it was trained and tested with XGBoost classifier for character classification. Finally, the recognized characters were displayed on the input image or in the text file. Table 5 shows the selection of proposed methods based on the number of contours.

Table 5 Selection of methods based on number of contours

| Characters | # Contours | Hu's Moments | Centroid Difference | Vertical Flip | Horizontal Flip |
|------------|---------------|-----------------|------------------------|------------------|--------------------|
| 0 | 2 | ✓ | ✓ | ✓ | ✓ |
| 1 | 1 | ✓ | × | ✓ | ✓ |
| 2 | 1 | ✓ | × | ✓ | ✓ |
| 3 | 1 | ✓ | × | ✓ | ✓ |
| 4 | 2 | ✓ | ✓ | ✓ | ✓ |
| 5 | 1 | ✓ | × | ✓ | ✓ |
| 6 | 2 | ✓ | ✓ | ✓ | ✓ |
| 7 | 1 | ✓ | × | ✓ | ✓ |
| 8 | 3 | ✓ | × | ✓ | ✓ |
| 9 | 2 | ✓ | ✓ | ✓ | ✓ |
| A | 2 | ✓ | ✓ | ✓ | ✓ |
| B | 3 | ✓ | × | ✓ | ✓ |
| C | 1 | ✓ | × | ✓ | ✓ |
| D | 2 | ✓ | ✓ | ✓ | ✓ |
| G | 1 | ✓ | × | ✓ | ✓ |
| P | 2 | ✓ | ✓ | ✓ | ✓ |
| T | 1 | ✓ | × | ✓ | ✓ |

Removal of unwanted characters

There is a possibility that some unwanted characters might have also been extracted during the segmentation phase. So, this step will discuss the removal of undesirable characters (See Figure 29). The existence of unwanted characters obtained from the segmentation step might hamper the character recognition. Removal of unwanted character is one of the preprocessing steps in the recognition phase. In our approach, we do not resize the characters, but we take the actual size of the characters segmented during the segmentation phase for the recognition. During the segmentation phase, some filtering properties like aspect ratio and bounding box area were used to remove unwanted characters. However, it is challenging to define a perfect parameter: aspect ratio and the contour area, which will suffice all the characters in the license plates. Hence, we had set a smaller aspect ratio and character area to allow some of the unwanted characters to get segmented along with the license plate characters. Most of the unwanted characters have smaller height compared to the height of the valid license plate characters. Moreover, the height of the valid characters is very close to each other with a difference of 1 to 3. Therefore, the height of the characters was taken to decide whether it is noise (outliers) or not using the statistical methods. In this step, 3 statistical methods such as *standard deviation*, *mean absolute deviation* and *modified z-score* methods were used to filter out the unwanted characters. Figure 30 shows the proposed flow chart for the removal of unwanted characters.

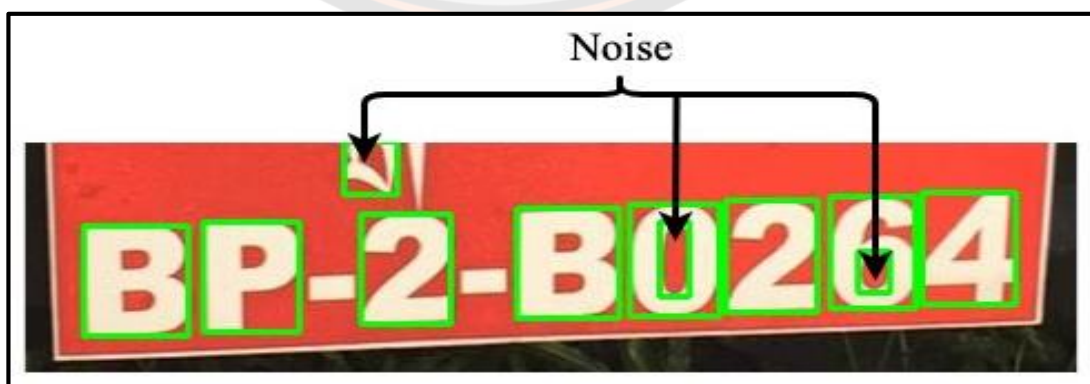


Figure 29 Unwanted characters segmented along with the license plate characters

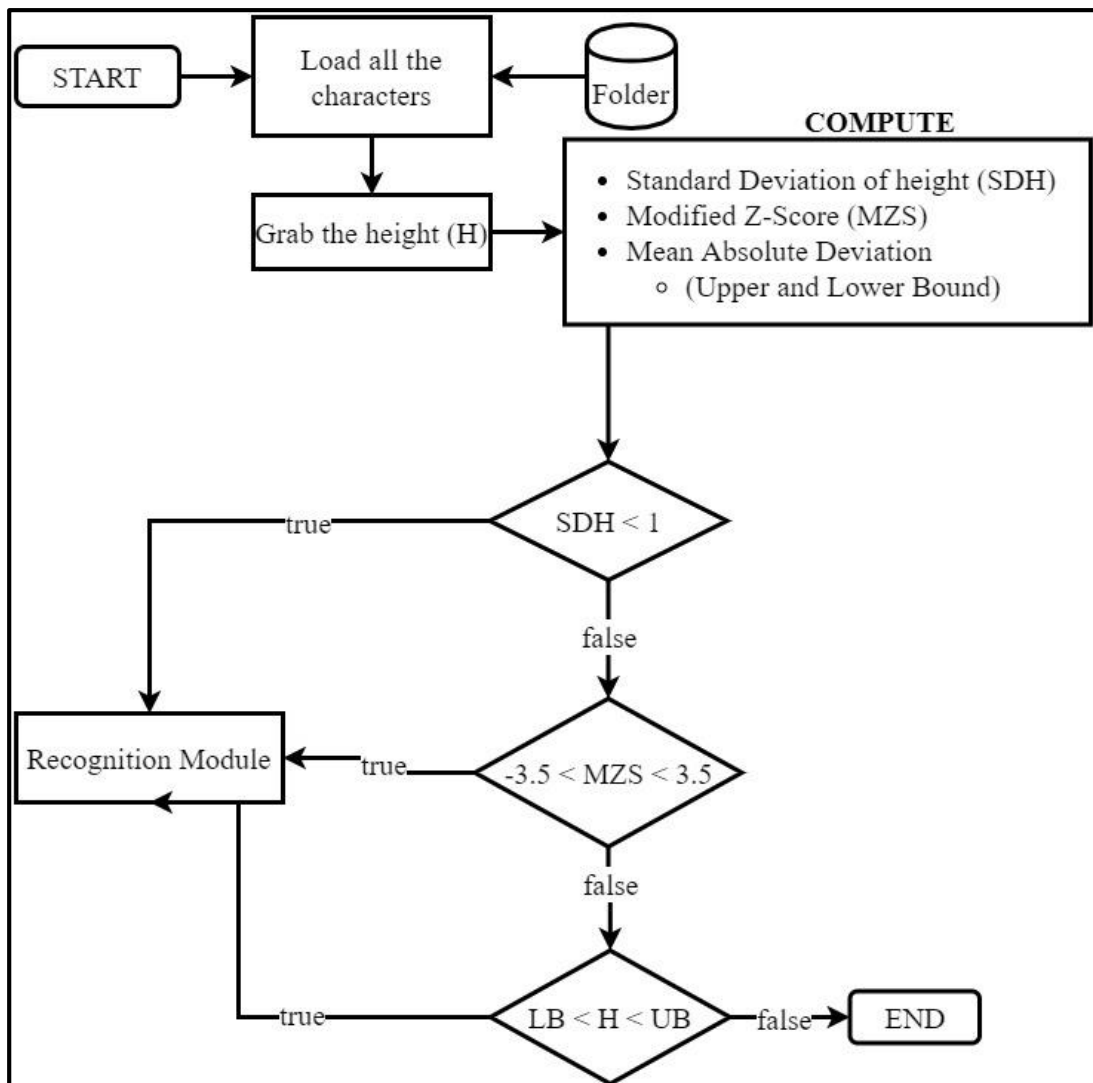


Figure 30 Flow chart of removing unwanted characters

The algorithm to remove unwanted characters from the true license plate characters is illustrated in Algorithm 2.

Algorithm 2: Removal of unwanted characters

Input: Segmented characters (Incl. unwanted characters)

Output: True license plate characters

// Get all the images from the folder

images \leftarrow Read all the images

```

// Get height of the images
allHeight ← Extract height of all the files
// Compute mean
meanH ← Mean of the height (allHeight)
// Compute median
medianH ← Median of the height (allHeight)
// Compute standard deviation
stdH ← Standard deviation of height (allHeight)
Set  $\bar{x} \leftarrow 0$ ,  $\tilde{x} \leftarrow \text{list}$ 
For image in images do
    image ← Read one image at a time
    imH ← Assign height of the image
    x ← |imH - meanH|
    y ← |imH - median|
     $\tilde{x} \leftarrow \tilde{x} + x$ 
    Append y to  $\tilde{x}$ 
End for
// Calculate mean absolute deviation, upper and lower bound
MAD ←  $\bar{x}/\text{Length}(\text{images})$ 
UB ← meanH + MAD
LB ← meanH - MAD
// Calculate median absolute deviation
medianAD ← Median of  $\tilde{x}$ 
// No unwanted characters
IF stdH < 2 then
    Save all the characters
// Modified z-score
Else if medianAD != 0 then
    For img in images do
        image ← Read one image at a time
        imH ← Assign height of the image
        ZScore ←  $0.6745 * (\text{imH} - \text{medianH})/\text{medianAD}$ 
        If ZScore > -3.5 && ZScore < 3.5 then
            Save the character
    End if

```

```

End for
// Upper and lower bound
Else
  For img in images do
    image ← Read one image at a time
    imH ← Assign height of the image
    If (imH >= LB && imH <= UB) then
      Save the character (s)
  End for
End If

```

1. Mean: Mean is an average set of a number which measures the central tendency.

The formula to compute mean is given in Equation 20:

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n} \quad (20)$$

2. Standard Deviation: A standard deviation (Equation 21) is a number that measures how far a set of numbers lie apart. The standard deviation was used when there was an absence of unwanted characters. Usually, the value of the standard deviation for VALID license plate characters was less than 1. Table 6 shows the dimensions of the extracted characters for one license plate without the unwanted characters.

$$\sigma = \sqrt{\frac{1}{N} \sum (x - \bar{x})^2} \quad (21)$$

Table 6 Valid license plate characters having standard deviation less than one

| Width | Height (x) | (x-mean) | (x-median) | Noise |
|-------------------------------|-------------|----------|------------|-------|
| 47 | 40 | 0.86 | 2 | No |
| 42 | 39 | 0.14 | 2 | No |
| 22 | 38 | 1.14 | 0 | No |
| 34 | 39 | 0.14 | 0 | No |
| 35 | 40 | 0.86 | 0 | No |
| 33 | 39 | 0.14 | 0 | No |
| 24 | 39 | 0.14 | 13 | No |
| Mean (\bar{x}) | 39.14 | | | |
| Median (\tilde{x}) | 39 | | | |
| Standard Deviation | 0.63 | | | |
| Mean Absolute Deviation (x) | | 0.49 | | |
| Median Absolute Deviation (y) | | | 0 | |
| Upper Bound ($\bar{x} + x$) | 40 | | | |
| Lower Bound ($\bar{x} - x$) | 39 | | | |

In Table 6, the modified z-score cannot be used since the median absolute deviation (y) is zero, and that will result in a divide by zero error. The mean absolute deviation (upper and lower bounds) also cannot be used since some of the dimensions are not between the given range. Therefore, the standard deviation of height was used since the value of the standard deviation is less than one.

3. Mean Absolute Deviation: The mean absolute deviation (MAD) measures the distance of all the elements from the mean of the same datasets. The MAD of a given data is an absolute average distance between each data value and the mean, as given by Equation 22:

$$MAD = \frac{\sum |datavalue - \bar{x}|}{Number\ of\ values} \quad (22)$$

Using the value of MAD, the upper and lower bound was calculated by adding/subtracting to/from the mean value (Upper Bound = MAD + Mean, Lower Bound = MAD – Mean). Table 7 shows the removal of unwanted characters using the upper and lower bounds.

Table 7 Removal of unwanted characters using mean absolute deviation

| Width | Height (x) | (x-mean) | (x-median) | Noise |
|-------------------------------|------------|--------------|------------|------------|
| 46 | 57 | 2.78 | 2 | No |
| 41 | 57 | 2.68 | 2 | No |
| 23 | 55 | 0.78 | 0 | No |
| 45 | 55 | 0.78 | 0 | No |
| 36 | 55 | 0.78 | 0 | No |
| 35 | 55 | 0.78 | 0 | No |
| 35 | 55 | 0.78 | 0 | No |
| 16 | 42 | 12.22 | 13 | Yes |
| 36 | 57 | 2.78 | 2 | No |
| Mean (\bar{x}) | 54.22 | | | |
| Median (\tilde{x}) | 55 | | | |
| Standard Deviation | 4.42 | | | |
| Mean Absolute Deviation (x) | | 2.72 | | |
| Median Absolute Deviation (y) | | | 0 | |
| Upper Bound ($\bar{x} + x$) | 57 | | | |
| Lower Bound ($\bar{x} - x$) | 52 | | | |

In the above Table 7, there is one unwanted noise with **16x42** dimensions. We cannot use standard deviation because its value is greater than one. Moreover, the modified z-score cannot be used since the value of median absolute deviation is zero and it will result in divide by zero error. Therefore, the mean absolute deviation was

proposed to compute the upper and lower bounds. Those heights which fall between the given bounds were considered as the valid characters.

4. Modified Z-Score Method (MZS): Modified Z-Score method is also used to detect outliers in the dataset. Instead of using mean and standard deviation, the modified z-score method uses Median Absolute Deviation to calculate the outliers. The median absolute deviation (Equation 23) is a measure of how spread out a set of data is. Although standard deviation and variance also measures the spread of the data, however median absolute deviation is less affected by the outliers. The Median Absolute Deviation is calculated by taking the absolute difference between the actual data and median, and then calculating the median of those differences.

$$\begin{aligned} \text{Median Absolute Deviation (mAD)} \\ = \text{median} \{|\text{datavalue} - \tilde{x}|\} \end{aligned} \quad (23)$$

If the value of median absolute deviation is equal to zero, then there are no deviations and all the values are the same. Iglewicz and Hoaglin (1993) proposed a modified z-score method to detect outliers as shown in Equation 24:

$$Mzs_i = \frac{0.6745(\text{datavalue} - \tilde{x})}{mAD} \quad (24)$$

Where 0.6745 (constant) is the 0.75th quartile of the standard normal distribution, to which the mAD converges to. As a thumb rule, the z-score values which are not between -3.5 and +3.5 are considered as the outliers. Table 8 shows the removal of unwanted characters using median absolute deviation and the modified z-score method.

Table 8 Removal of unwanted characters using modified z-score method

| Width | Height (x) | (x-mean) | (x-median) | Mzsi | Noise |
|------------------------------------|------------|-------------|------------|-----------------|------------|
| 49 | 53 | 6.8 | 0 | 0 | No |
| 26 | 17 | 29.2 | 36 | -24.28 | Yes |
| 42 | 54 | 7.8 | 1 | 0.6745 | No |
| 25 | 57 | 10.8 | 4 | 2.698 | No |
| 37 | 52 | 5.8 | 1 | -0.6745 | No |
| 16 | 38 | 8.2 | 15 | -10.1175 | Yes |
| 43 | 32 | 14.2 | 21 | -14.1645 | Yes |
| 37 | 53 | 6.8 | 0 | 0 | No |
| 40 | 53 | 6.8 | 0 | 0 | No |
| 25 | 53 | 6.8 | 0 | 0 | No |
| Mean (\bar{x}) | 46.2 | | | | |
| Median (\tilde{x}) | 53 | | | | |
| Standard Deviation | 12.32 | | | | |
| Mean Absolute Deviation | | 10.32 | | | |
| Median Absolute Deviation | | | 1 | | |
| Upper Bound = ($\bar{x} + x$) | 57 | | | | |
| Lower Bound ($\bar{x} - x$) | 36 | | | | |

In Table 8, there are three unwanted characters with dimensions **26x17**, **16x38** and **43x32**. The concept of bounds (lower and upper bound) cannot be used since some of the dimensions of the unwanted characters falls between the given range. Since, the value of median absolute deviation is greater than one, we had used the modified z-score to eliminate the outliers from the data. After removing unwanted

characters, the features from the characters were extracted to create a character classifier.

Feature extraction using Hu's moments

Feature extraction refers to the extraction of useful characteristics from the object that can be represented with fewer data compared to the original data. The success of character classification entirely depends on the quality of features used in the classifier. The selection of an appropriate feature extraction method is one of the most critical factors to achieve high recognition. In this research, the combination of Hu's moments along with the proposed methods were adopted to extract the features for the classifier. The features from 17 characters were extracted for the study since the frequency of usage is more in Bhutanese license plates. In the character classifier, a total of 4948 features were extracted from the 17 characters using Hu's moments and the proposed methods. Table 9 shows the frequency of characters and numbers used in the classifier with its ASCII value.

Table 9 Frequency of characters used in the classifier with its ASCII value

| Character | Count | ASCII | Character | Count | ASCII |
|-----------|-------|-------|-----------|-------|-------|
| 0 | 255 | 48 | 9 | 294 | 57 |
| 1 | 375 | 49 | A | 258 | 65 |
| 2 | 310 | 50 | B | 385 | 66 |
| 3 | 267 | 51 | C | 281 | 67 |
| 4 | 254 | 52 | D | 296 | 68 |
| 5 | 234 | 53 | G | 290 | 71 |
| 6 | 271 | 54 | P | 317 | 80 |
| 7 | 284 | 55 | T | 282 | 84 |
| 8 | 295 | 56 | | | |

Moments are the scalar quantities used to characterize a function and to capture its significant features. They have been mostly used in the statistics for description of the shape. Hu's Moments contains 6 orthogonal invariants and one

skew orthogonal invariants based on the algebraic invariants, which are invariant to rotation, translation and scaling (M.-K. Hu, 1962). Hu's moments are mainly used to describe, characterize and quantify the shape of an object in an image.

Image moments are the weighted average of the image pixel intensities. Moment (i, j) , of a digital image $I(x, y)$, of a size $M \times N$ is defined in the Equation 25:

$$M_{ij} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^i y^j I(x, y) \quad (25)$$

Where i and j are the order of x and y respectively. The above moments are often referred to as raw moments. The raw moments entirely depend on the intensity of pixels and the location of blobs in the image.

A central moment (Equation 26) of an image is same as the image moment but in this, the mean value \bar{x} and \bar{y} is subtracted from the moment value x and y :

$$\mu_{ij} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x - \bar{x})^i (y - \bar{y})^j I(x, y) \quad (26)$$

Where the centroid (\bar{x}, \bar{y}) of the image is given in the Equation 27:

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}} \quad (27)$$

The above central moments are *translation invariant*, which means, if two identical shapes are placed in different positions in the image, their central moments will be same. So far, the moments are invariant to translation but to make it invariant to scaling, the central moments are normalized as shown in Equation 28:

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\frac{i+j}{2}+1}} \quad (28)$$

It is not enough to have a moment that is invariant to translation. We need a moment that are invariant to translation, scale and rotation. Hu's moments are 7 sets of numbers calculated using *central moments* that are invariant to image transformations like rotation, scaling and translation. The 7 Hu's moments are defined below:

$$h_1 = \eta_{20} + \eta_{02}$$

$$h_2 = (\eta_{20} - \eta_{02})^2 - 4\eta_{11}^2$$

$$h_3 = (\eta_{30} - \eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$h_4 = (\eta_{30} + \eta_{12})^2 + (3\eta_{21} + \eta_{03})^2$$

$$h_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) \\ + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2)$$





$$h_6 = (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + 4\eta_{11}(\eta_{30} \\ + \eta_{12})(\eta_{21} + \eta_{03})$$

$$h_7 = (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) \\ - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2)$$

Zekovich and Tuba (2013) used Hu's moments to recognize the handwritten digits from the MNIST database. They have used Hu's moments to extract the features from the handwritten characters and, the overall recognition rate was around 63%. Apart from this, (Kanaparthi, 2012) proposed the use of Hu's moments for detection and recognition of U.S speed signs.

Hu's moments alone are not enough to get better accuracy in the recognition because some characters such as (6, 9), (B, 8), (A, 4) and (7, T) have similar geometrical shapes. These characters will confuse the character classifier, especially when the features are almost identical. Table 10 shows the similar moments generated by Hu's algorithm for number (6, 9) and (B, 8). This type of characters having the same geometrical shapes will affect in developing effective and robust ANPR systems. Therefore, to solve this problem, the following subsequent methods are proposed to reduce the number of misclassifications.

Table 10 Examples of similar Hu's moments for different characters

| | | License plate characters | | | |
|---------------------|-----------|---|---|--|---|
| | |  |  |  |  |
| Hu's moments | H0 | 3.05150747 | 3.008181 | 3.08095057 | 3.11452098 |
| | H1 | 7.10036403 | 6.90290468 | 7.08543011 | 7.57224065 |
| | H2 | 10.8730879 | 10.52923744 | 11.37545707 | 11.81952751 |
| | H3 | 11.6963587 | 11.29848155 | 12.7640063 | 12.6750007 |
| | H4 | -22.9890148 | -22.32389572 | 24.84757681 | 25.42984215 |
| | H5 | -15.2867478 | -14.95380059 | 16.41254984 | 16.65678325 |
| | H6 | -23.7037038 | -22.41036902 | 25.43844835 | 24.94431772 |

Feature extraction from Centroid Difference method

The Centroid Difference (CD) method is used to extract features based on the comparison of distance between the two centroids of the image (Jamtsho, Riyamongkol, & Waranusast, 2019a). The centroid of the shape is the arithmetic mean of all the points on the shape. The CD method is especially used in characters having two contours. This method was especially developed to extract the features from (6, 9) since these two characters tend to give similar Hu's moments. In this method, two centroids are calculated from the character as shown in Figure 31.

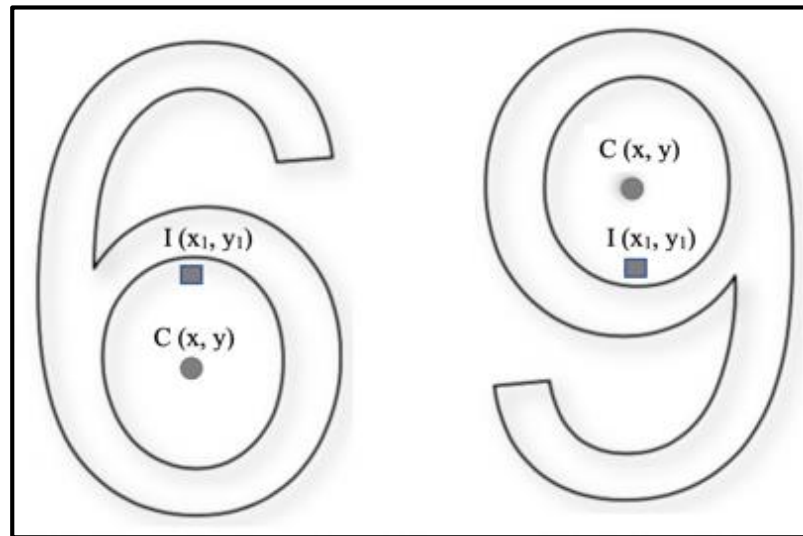


Figure 31 Illustrations of centroids in CD method

Source: Jamtsho et al., 2019a

In Figure 31, $C(x, y)$ represents the central coordinates of the small hole inside number 6 and 9, and $I(x_1, y_1)$ denotes the central coordinates of the character. The selection of the image centroid and the hole centroid was decided based on the area of the contour. If the outer area is greater than the inner area, then the central coordinates of the image was calculated; otherwise, the inner centroids was calculated. After finding the two centroids, the y coordinate of the small hole was subtracted from the corresponding y coordinate of the character, as given by the Equation 29:

$$CD_y = y_1 - y \quad (29)$$

Where y_1 is the y -coordinate of the image centroid and y is the y -coordinate of the hole centroid. After the subtraction of the corresponding y -coordinates, the difference in the y -coordinates were used as one of the features along with Hu's moments. Usually for number 6, the features were negative and positive for number 9. In CD method, we are just comparing the y coordinates of the two centroids. The same process was applied to other characters, as mentioned above. The algorithm to calculate the value of centroid difference is given in Algorithm 3.

Algorithm 3: Algorithm to compute centroid difference value

Input: License plate character

Output: Features from CD method

```

image ← Read input image
gray ← Convert to grayscale
threshold ← Convert to binary image
// Compute the centroid of the image
M ← Moments (threshold)
Ix ← int(M["m10"] / M['m00'])
Iy ← int(M["m01"] / M['m00'])
contours ← Find the contours in the image
set areaList ← list, contourList ← list
For contour in contours do
    area ← Compute area of each contour
    append area to areaList
    append contour to contourList
End for

// Compute the centroid of inner hole
If (areaList[0] > areaList[1]) then
    // Compute the centroid of the inner hole
    M ← Moments (contourList[1])
    Cx ← int(M["m10"] / M['m00'])
    Cy ← int(M["m01"] / M['m00'])
     $CD \leftarrow \frac{Iy - Cy}{imH} * 100$ 

```

Feature extraction from Vertical and Horizontal flip method

The vertical flip (VF) and horizontal flip (HF) are the second proposed method that works based on the comparison of pixel densities. Unlike centroid difference method, this method can be used to extract features from all the characters since it does not need to depend on the number of contours in the image. This method helps the classifier to differentiate (B, 8), (A, 4), (T, 7) and (0, D) characters. The flowchart of the horizontal and vertical flip methods is shown in Figure 32 and 33.

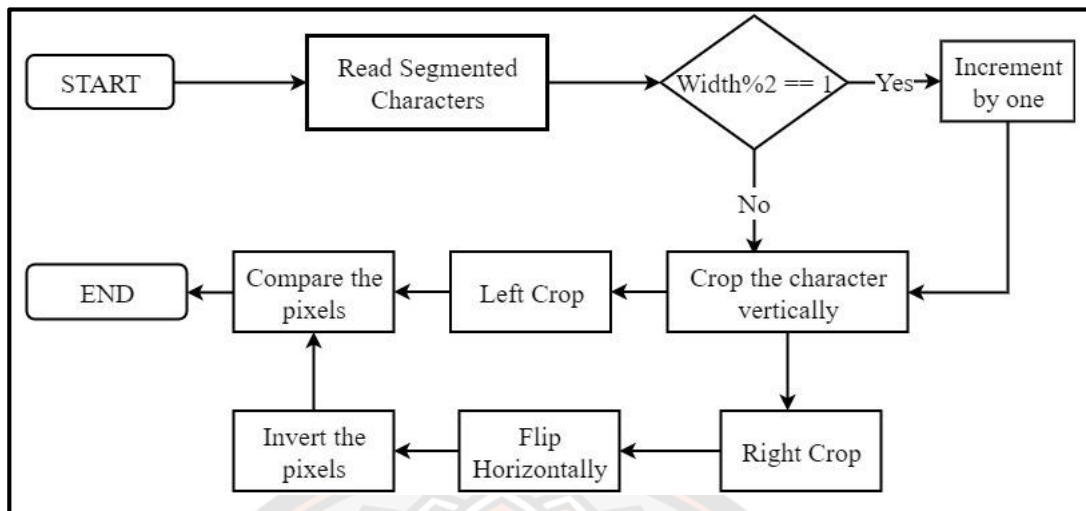


Figure 32 Flowchart of horizontal flip method

The algorithm to compute features using horizontal flip method is depicted in Algorithm 4.

Algorithm 4: Feature extraction using horizontal flip algorithm

Input: License plate character

Output: Features generated from algorithm

image \leftarrow Read input image

gray \leftarrow Convert to grayscale

threshold \leftarrow Apply OTSU's thresholding

(imH, imW) \leftarrow Get the height and width of the image

// Check whether width is even or not

If imW % 2 == 1 **then**

imW \leftarrow imW + 1

End if

// Left crop

(startY, endY) \leftarrow (0, imH)

(startX, endX) \leftarrow (0, imW/2)

LCrop \leftarrow threshold[startY:endY, startX:endX]

```

// Right crop
(startY, endY) ← (0, imH)
(startX, endX) ← (imW/2, imW)
RCrop ← threshold[startY:endY, startX:endX]
// Horizontal flip the right crop
rHorizontalFlip ← Flip the right crop (RCrop) in the horizontal direction
// Invert the pixels
not_rHorizontal ← Invert the horizontal flip image
// Find total pixels
totalPixels ← imH * imW
// Count the white pixels
nonZeroPixels ← countNonZero(not_rHorizontal)
// Compute percentage of white pixels
percent ← (nonZeroPixels/totalPixels) * 100

```

Note: In vertical flip method, we have to check whether the height is even or not and crop the image accordingly.

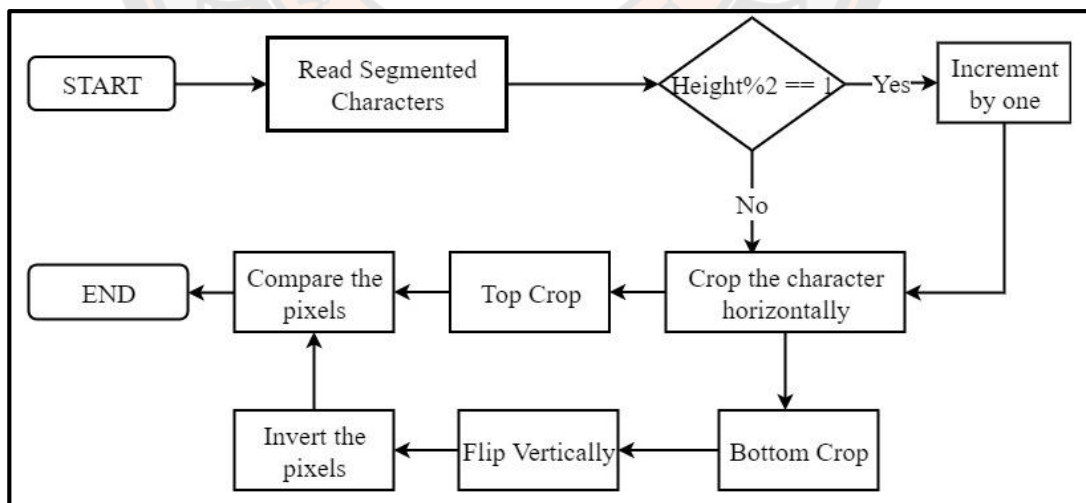


Figure 33 Flowchart of vertical flip method

In the flip methods, before cropping the character, the width/height of the character was checked to make sure that it is even in dimensions. It is basically done

to avoid a problem when we compare two portions of the characters. If the dimension is found to be ODD, one is added to the width/height depending on the flip method. After checking the dimensions, the original character image (Figure 34a and 35a) was cropped vertically/horizontally from the center into two portions (Figure 34(b, c) and 35 (b, c)). After the crop operation, the right/bottom portion of the image was flipped into the horizontal/vertical direction (Figure 34d and 35d) and, the BITWISE NOT operation was applied to invert the pixel values (Figure 34e and 35e). In the final step, the inverted image was compared with the left/top portion of the image using the BITWISE AND operation (Figure 34f and 35f). After the pixel comparison, the white pixels were converted to a percentage based on total pixels in the image and was used as the features. The HF and VF method play a vital role in distinguishing (B, 8), (A, 4) and (T, 7). Figure 34 and Figure 35 shows the sequence of steps executed to extract the features from character “8” and “B”.

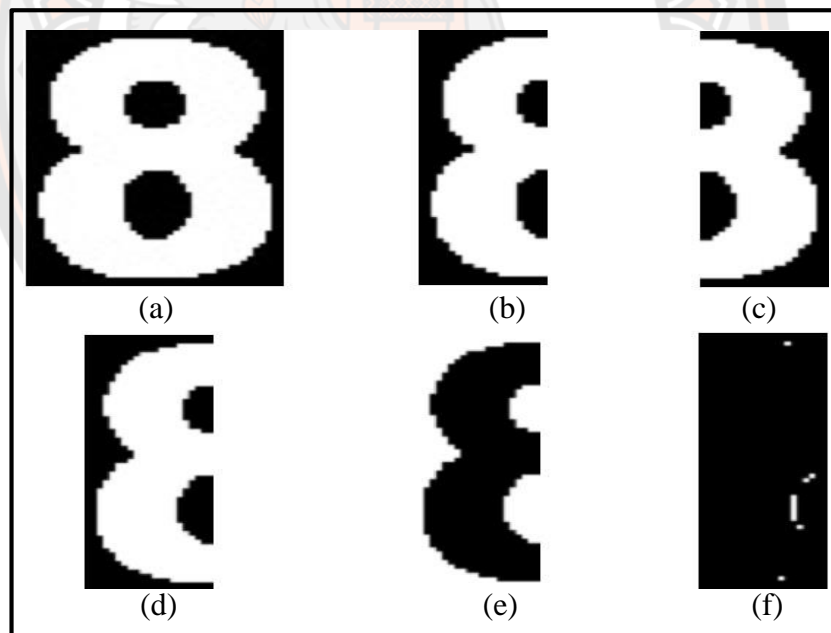


Figure 34 a) Original image, b) Left crop, c) Right crop, d) Horizontal flip of right crop, e) Invert pixels, f) Compare pixels

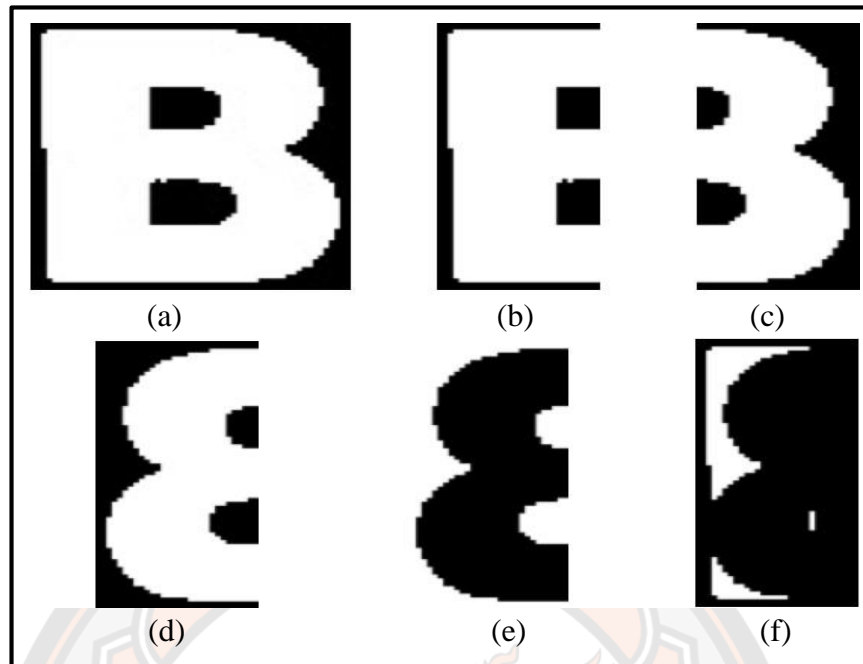


Figure 35 Original image, b) Left crop, c) Right crop, d) Horizontal flip of right crop, e) Invert pixels, f) Compare pixels

CHAPTER IV

RESULTS AND DISCUSSION

Introduction

In this chapter, the detailed simulation results of the proposed system are presented. The main aim of the study was to develop an ANPR system that can detect and recognize Bhutanese license plates and design a method to extract features from the characters. The overall accuracy of the system is also presented in this section. For the development and simulation of results, the Python programming language with OpenCV libraries was used to achieve the objectives of the study. Finally, the comparison between the proposed methods and algorithms discussed in the literature review is also presented. The proposed methods of each ANPR stage is compared with the methods discussed in Chapter II.

License plate localization

In the ANPR system, the license plate localization system is said to be one of the essential steps since all other subsequent steps entirely depend on the accuracy of the license plate extraction. By keeping in this mind, vehicle detection from the image was performed first to remove the false positives generated by the objects having similar license plate characteristics. For automatic license plate detection inside the vehicle image, a single convolutional neural network (YOLOv2) was used to accomplish the task. For the experiments, 1290 vehicle images were collected and annotated manually for training purposes. In the experiments, the model was trained for 10,000 iterations, where each iteration took 2.12 seconds. Once the training was completed, the model was evaluated to check the accuracy of the test datasets. The overall mean average precision generated by the model was 99.1% with the average training loss of 0.0261 for 64 batch size with 8 subdivisions. According to Alexey (2016), the model is said to be performing well if the average training loss is less than 0.060730. The average IOU with average precision rate generated by each iteration for VEHICLE and PLATE classes is shown in Table 11.

Table 11 Tabulation of average IOU and average precision of each class

| Iterations | Average Precision | | Average IOU (%) | Mean AP (mAP) (%) |
|-------------|-------------------|--------------|--------------------|----------------------|
| | VEHICLE (%) | PLATE (%) | | |
| 1000 | 99.9 | 98.99 | 82.6 | 99.48 |
| 2000 | 99.59 | 98.75 | 83.12 | 99.17 |
| 3000 | 99.23 | 98.64 | 83.76 | 99.93 |
| 4000 | 99.97 | 98.28 | 83.67 | 99.12 |
| 5000 | 99.97 | 98.25 | 83.67 | 99.11 |
| 6000 | 99.97 | 98.25 | 83.67 | 99.11 |
| 7000 | 99.97 | 98.26 | 83.85 | 99.12 |
| 8000 | 99.97 | 98.26 | 83.7 | 99.11 |
| 9000 | 99.97 | 98.26 | 83.88 | 99.12 |
| 10000 | 99.97 | 98.26 | 83.74 | 99.12 |

In Table 11, after training the YOLO model for 1000 iteration, the average IOU generated by the model was 82.6% with 99.48% mean average precision (mAP). The average IOU denotes the total intersection area between the predicted truth and the ground truth. In the table, there is notable average IOU between the predicted truth and the ground truth. Based on the average IOU value, we had selected the weights given after 9000 iterations since it had the highest average IOU comparing to other iterations. Figure 36 shows the average loss and validation mAP generated by YOLOv2 model during training.

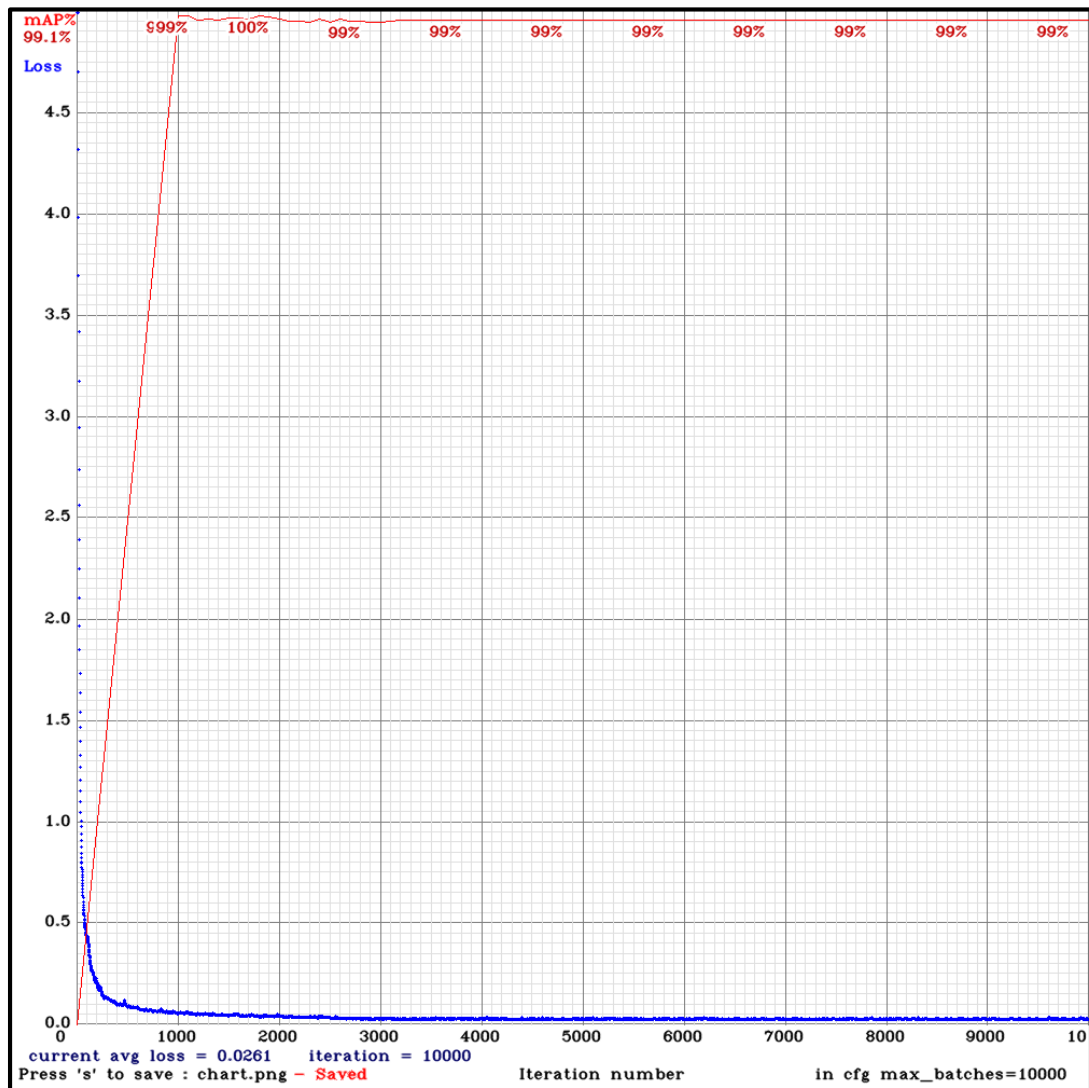


Figure 36 Graph showing validation mAP and average training loss

Performance evaluation of vehicle detection and license plate localization

To analyze the reliability of license plate recognition using YOLO algorithm, around 1050 images of Bhutanese license plates with varying dimensions were selected for the testing. All the selected images were given to the license plate localization phase for automatic detection of license plate after the detection of the vehicle. Some of the results generated by the localization method for the three types of license plate is shown in Figure 37.



Figure 37 Experimental results for detection of vehicle and localization of plate

The overall accuracy of the vehicle detection and license plate localization is shown in Table 12.

Table 12 Results from vehicle detection and license plate localization

| Total test images: 1050 | Correct | Accuracy (%) |
|----------------------------|---------|--------------|
| Vehicle detection | 1034 | 98.5 |
| License plate localization | 1012 | 97.9 |

In Table 12, out of 1050 vehicle images, only 1034 images passed through the vehicle detection phase, which results in a detection accuracy of 98.5%. The license plate localization of each image entirely depends on vehicle detection. Therefore, out of 1034 detected vehicle, only 1012 license plates got correctly localized with an accuracy of 97.9%. The reason for not localizing the license plate despite the detection of a vehicle is because the bounding box information of the license plate was not enclosed inside the vehicle bounding box, as shown in Figure 38.



Figure 38 License plate not localized inside vehicle bounding box

The system was tested to localize the license plate under shadow and light regions, as depicted in Figure 39 and 40. We can notice from the results that the proposed method for localization is reliable and efficient in detecting the vehicle and license plates.



Figure 39 Experimental test results for localizing license plate in shade region



Figure 40 Experimental test results for localization of license plates under light

Apart from shadow and lighting conditions, the performance of the proposed system was examined to detect a license plate surrounded by objects having similar license plate characters. In the original image, it can be seen that there are texts written near the license plates. Those text and license plate have a similar colour. However, the proposed system can only localize the license plates eliminating other characters. The results in Figure 41 shows that the proposed method differentiate license plate from the text.



Figure 41 Experimental results for localization of license plate surrounded by text

The performance of the proposed method was examined to localize the license plates after the detection of the vehicle automatically. The automatic detection of license plates is mainly done to avoid false positives generated by the signboard since these objects have similar characteristics as license plates. The result in Figure 42 shows detection of license plate in two ways.

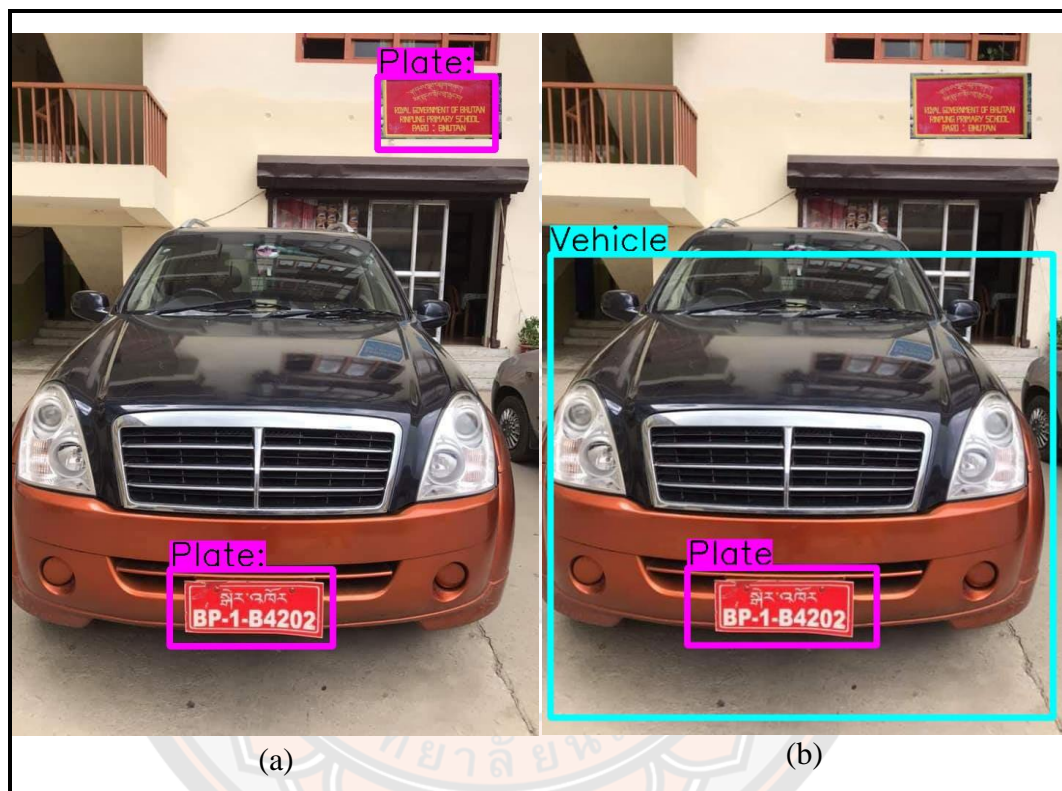


Figure 42 a) Signboard is detected as license plate, b) Sign board is not detected

In Figure 42a, the signboard from the car image was detected as a license plate when we do localization before checking the presence of a vehicle. However, in Figure 42b, incorrect detection was eliminated with the introduction of automatic license plate detection inside the vehicle image. By doing so, it helps to create a robust localization method without any false positives.

Also, the performance of the localization method was evaluated for detecting license plates in the night lighting condition. It is challenging for the traditional approaches to identify the license plate since most of the algorithms use physical characteristics of the license plate, which is hardly visible. However, the proposed method can localize the license plate during the night vision. The results in Figure 43 shows that the proposed method can localize the license plates during night vision.

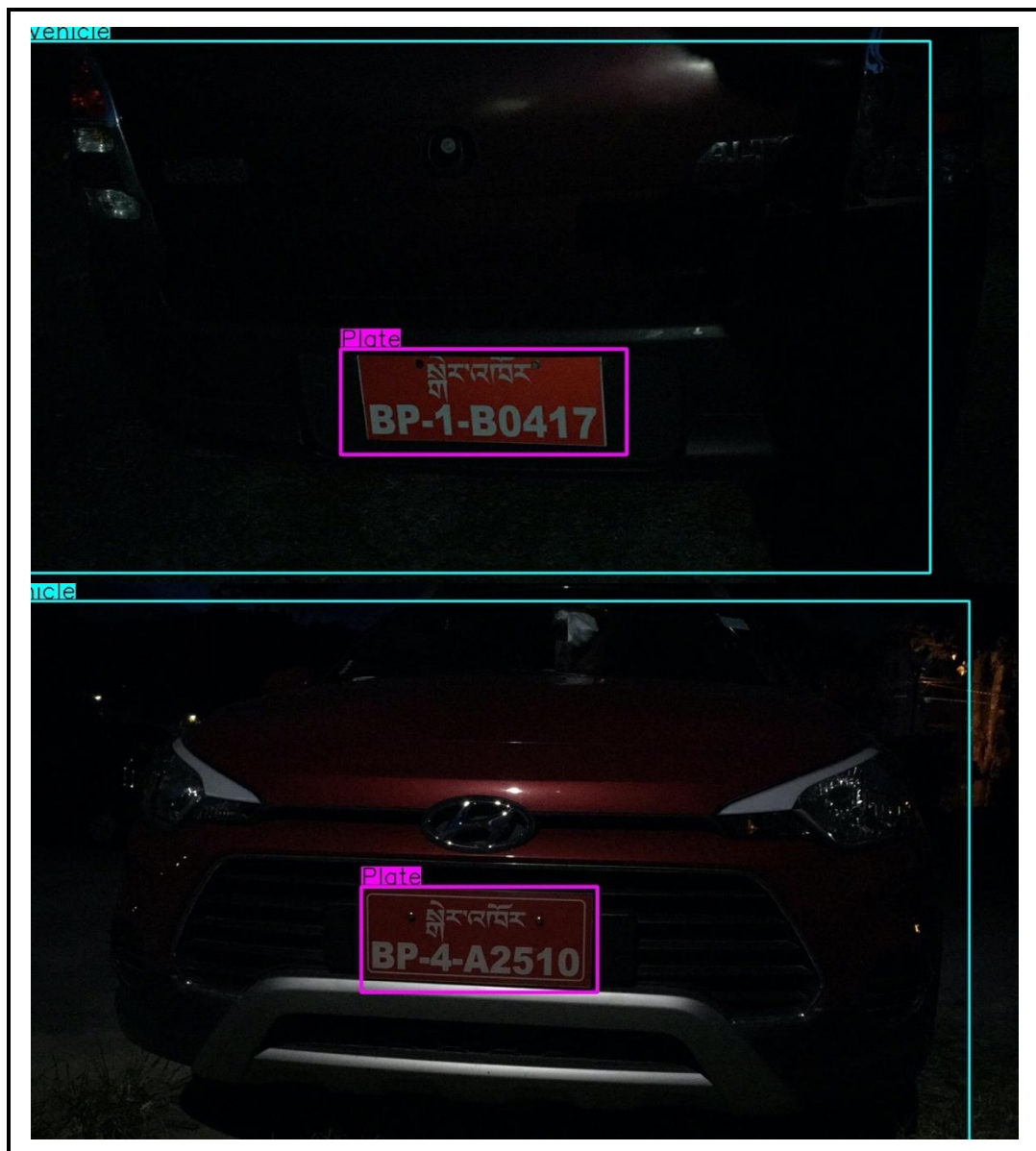


Figure 43 Experimental results for localizing license plates in night view

The other distinct performance of the proposed method is that it can localized the license plate which is partially blocked by some objects such as spare tyres at the back of the vehicle or crash guard at the front of the vehicle as shown in Figure 44. The traditional methods such as boundary information cannot be used since the upper license plate boundary is hidden by the tyre.



Figure 44 Experimental results for localizing license plates block by the tyre and crash guard

Character segmentation

In the character segmentation phase (Chapter III), the extracted license from the localization step was given as an input to isolate the characters for recognition. Before segmenting the characters, the license plate was enhanced using some of the preprocessing steps such as grayscale conversion, histogram equalization, contrast enhancement and thresholding. After that, the characters were segmented from the license plate using connected component analysis. In this section, the results and performance of the character segmentation method have been discussed.

Performance evaluation of segmentation

For the evaluation of character segmentation phase, all the extracted license plates (1012 license plates) from the localization stage were taken as the input in this stage. Before segmenting the characters, the extracted license plates were first preprocessed, as discussed in Chapter III. The overall segmentation accuracy generated by the algorithm is given in Table 13.

Table 13 Accuracy from character segmentation

| | |
|---------------------------------------|-------------|
| Total localized license plates | 1012 |
| Correct segmentation | 973 |
| Incorrect segmentation | 39 |
| Overall accuracy | 96.1% |

In Table 13, out of 1012 extracted license plates from the localization step, only 973 license plates got correctly segmented. The overall accuracy of the character segmentation algorithm was 96.1%. The error of 3.9% is due to poor resolution of license plates, faded license plate characters and some characters had joined each other as shown in Figure 45. Some of the characters were joined with the license plate boundary, hence affecting the performance of the segmentation.



Figure 45 Experimental results showing incomplete segmentation

Figure 46 shows the segmented characters for three types of license plates found in Bhutan. As we can notice from the results that some unwanted characters were also segmented along with the license plate characters. Those unwanted characters were treated as noise (outliers), and the proposed method discussed in Chapter III handles those unwanted characters. The segmented characters are shown by the bounding boxes (green colour).

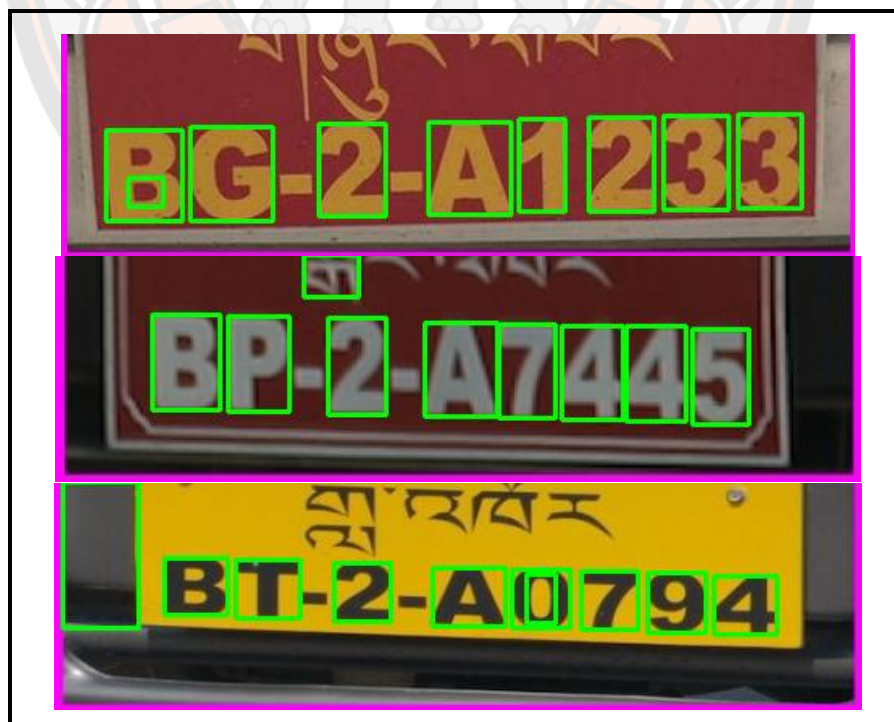


Figure 46 Segmented characters for three types of license plates

The performance of character segmentation method was also evaluated with license plate images taken during the night. The proposed character segmentation method with a preprocessing step was able to segment the characters, as shown in Figure 47.

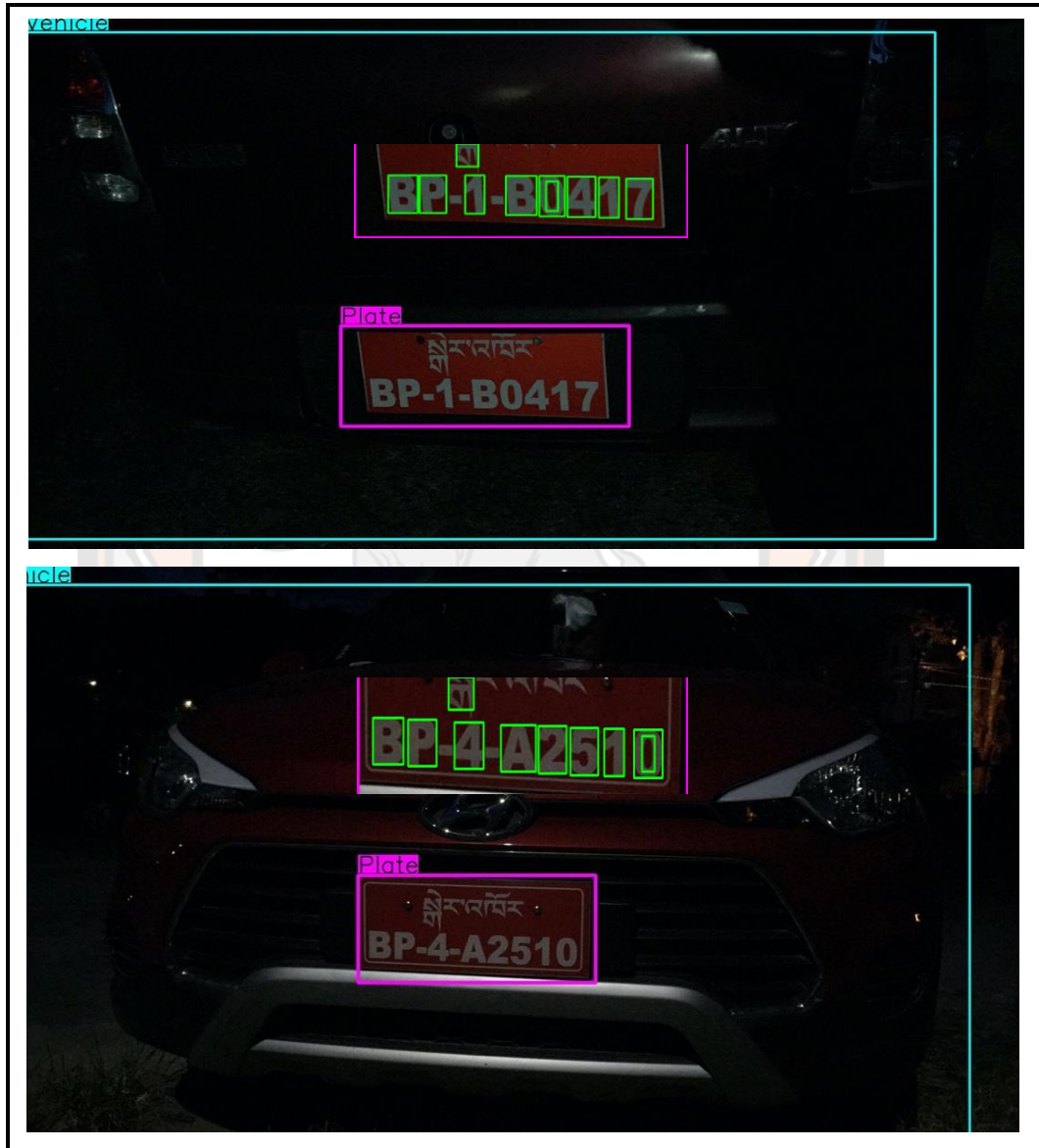


Figure 47 Experimental results for character segmentation in night view

In the character segmentation method, a “*White pixel counting with inversion method*” was proposed to especially handle TAXI (BT) since it has a yellow background with a black foreground. This method was mainly designed to generate a binary license plate having a black background with white foreground. As shown in Figure 48, the proposed segmentation algorithm correctly segments all the characters after applying “*White pixel counting with inversion method*”.

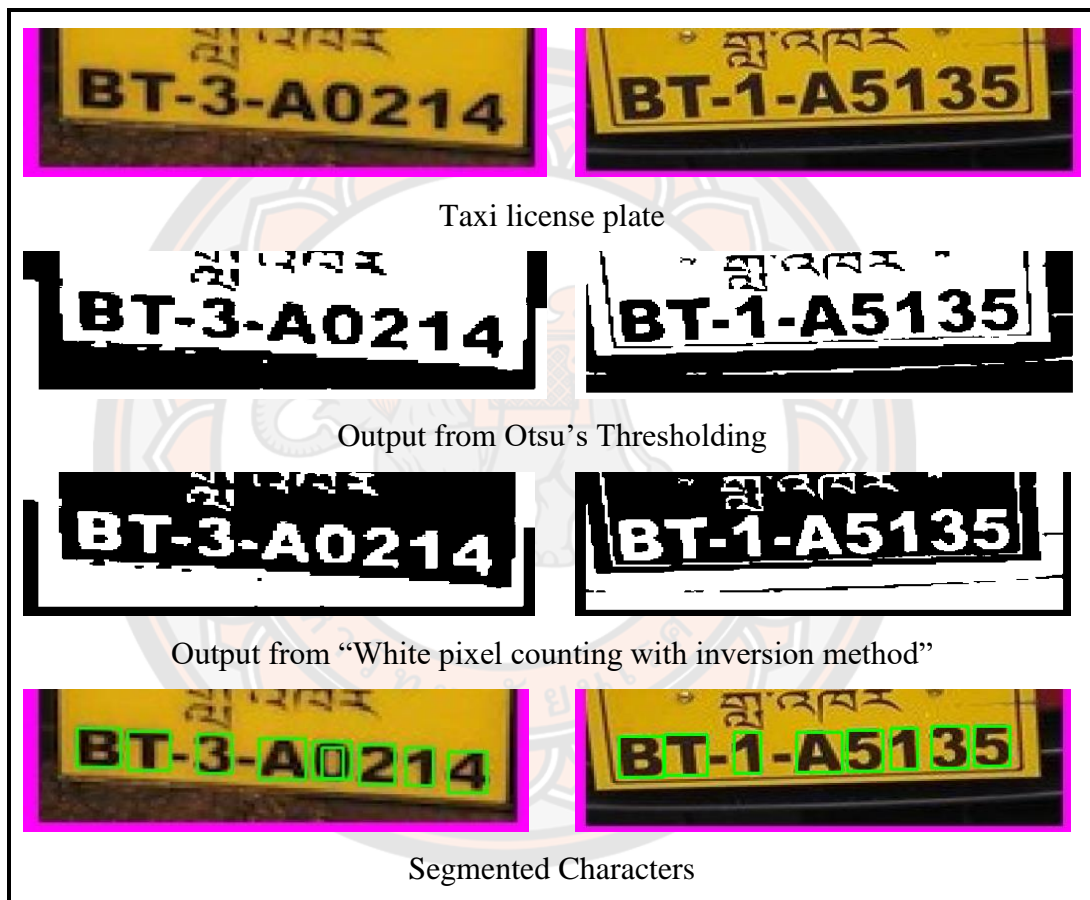


Figure 48 Experimental results for character segmentation of taxi license plates

Character recognition

Character recognition is the last stage in the ANPR technology. The accurate recognition of license plate characters entirely depends on the accurate localization of license plates followed by segmentation of individual characters. Before recognizing the characters, a character classifier needs to be modelled, which has been discussed in Chapter III. The features from the individual characters were extracted to create a classifier and the WEKA software has been used to check the effectiveness of the features.

Train and test the extracted features in WEKA

In this section, the quality of the extracted features was evaluated using WEKA program. The extracted features were trained and tested using a machine-learning algorithm. The WEKA software was used to train and test the extracted features since it contains all the machine learning algorithms. WEKA is a collection of machine learning algorithms used for data mining problems. It is helpful in data visualization, classification, regression, clustering and association (Witten, Frank, Hall, & Pal, 2016). Among many machine learning algorithms, the random forest classifier with 10 cross-fold validation performs well in the extracted features. For the evaluation of machine learning algorithm in the given features, the classification matrix generated by the Hu's moments was compared with the classification matrix obtained using Hu's moments and proposed methods. The classification matrix obtained from the trained model is given in Table 14 and Table 15.

Table 14 Results of classification matrix based on Hu's moments (WEKA)

Part I

| | | PREDICTED | | | | | | | | | |
|--------|---|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ACTUAL | 0 | 232 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 7 | 1 |
| | 1 | 0 | 358 | 4 | 0 | 6 | 0 | 0 | 6 | 0 | 0 |
| | 2 | 0 | 1 | 272 | 0 | 0 | 2 | 5 | 0 | 0 | 20 |
| | 3 | 0 | 0 | 0 | 256 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 4 | 0 | 4 | 0 | 0 | 224 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 2 | 1 | 8 | 0 | 0 | 204 | 1 | 1 | 5 | 0 |
| | 6 | 0 | 0 | 9 | 0 | 0 | 1 | 148 | 0 | 1 | 109 |
| | 7 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 274 | 0 | 0 |
| | 8 | 10 | 0 | 2 | 0 | 0 | 3 | 1 | 0 | 228 | 1 |
| | 9 | 0 | 1 | 13 | 1 | 0 | 3 | 100 | 0 | 2 | 171 |
| | A | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 |
| | B | 4 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 39 | 4 |
| | C | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 0 | 0 | 0 |
| | D | 0 | 1 | 6 | 0 | 1 | 1 | 2 | 0 | 3 | 3 |
| G | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 0 | 0 | 8 | |
| P | 0 | 2 | 0 | 0 | 4 | 0 | 0 | 1 | 0 | 0 | |
| T | 0 | 1 | 0 | 0 | 3 | 0 | 0 | 13 | 0 | 0 | |

Part II

| | | PREDICTED | | | | | | |
|--------|---|------------|------------|------------|------------|------------|------------|---|
| | | A | B | C | D | G | P | T |
| ACTUAL | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |
| | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 2 | 0 | 2 | 1 | 3 | 4 | 0 | 0 |
| | 3 | 0 | 0 | 9 | 0 | 1 | 0 | 0 |
| | 4 | 14 | 0 | 0 | 0 | 0 | 10 | 2 |
| | 5 | 0 | 6 | 0 | 3 | 3 | 0 | 0 |
| | 6 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| | 7 | 2 | 0 | 0 | 0 | 0 | 1 | 4 |
| | 8 | 0 | 45 | 0 | 4 | 1 | 0 | 0 |
| | 9 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |
| | A | 247 | 0 | 0 | 0 | 0 | 1 | 1 |
| | B | 0 | 310 | 0 | 15 | 10 | 0 | 0 |
| | C | 0 | 0 | 268 | 0 | 1 | 0 | 0 |
| | D | 0 | 4 | 1 | 264 | 9 | 1 | 0 |
| G | 0 | 7 | 2 | 7 | 261 | 0 | 0 | |
| P | 5 | 0 | 0 | 1 | 0 | 303 | 1 | |
| T | 5 | 0 | 3 | 0 | 0 | 2 | 255 | |

Note: Since the content of the table is large to fit into one page, the confusion matrix is separated into Part I and Part II. The boldface numbers are the true positive (TP) of a confusion matrix.

Table 15 Results of classification matrix based on proposed methods (WEKA)

Part I

| | | PREDICTED | | | | | | | | | |
|--------|---|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ACTUAL | 0 | 244 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 |
| | 1 | 0 | 371 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| | 2 | 0 | 1 | 302 | 0 | 0 | 3 | 0 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 | 265 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 4 | 0 | 0 | 0 | 0 | 245 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 0 | 0 | 1 | 0 | 0 | 231 | 0 | 1 | 1 | 0 |
| | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 270 | 0 | 0 | 0 |
| | 7 | 0 | 4 | 0 | 0 | 0 | 1 | 0 | 268 | 0 | 0 |
| | 8 | 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 266 | 0 |
| | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 294 |
| | A | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 |
| | B | 3 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 11 | 0 |
| | C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | D | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 3 | 0 |
| G | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| T | 0 | 3 | 0 | 1 | 1 | 0 | 0 | 6 | 0 | 0 | |

Part II

| | | PREDICTED | | | | | | |
|--------|---|------------|------------|------------|------------|------------|------------|----|
| | | A | B | C | D | G | P | T |
| ACTUAL | 0 | 0 | 7 | 0 | 0 | 1 | 0 | 0 |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| | 2 | 0 | 2 | 0 | 1 | 1 | 0 | 0 |
| | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| | 4 | 9 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 6 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 11 |
| | 8 | 0 | 15 | 0 | 3 | 2 | 0 | 0 |
| | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | A | 254 | 0 | 0 | 0 | 0 | 0 | 0 |
| | B | 0 | 350 | 0 | 13 | 5 | 0 | 0 |
| | C | 0 | 0 | 278 | 0 | 3 | 0 | 0 |
| | D | 0 | 5 | 0 | 283 | 4 | 0 | 0 |
| G | 0 | 2 | 3 | 6 | 277 | 0 | 0 | |
| P | 0 | 0 | 0 | 0 | 0 | 317 | 0 | |
| T | 0 | 0 | 0 | 0 | 0 | 0 | 271 | |

Note: Since the content of the table is large to fit into one page, the confusion matrix is separated into Part I and Part II. The boldface numbers are the true positive (TP) of a confusion matrix.

In Table 14, the number “6” was incorrectly classified 109 times as number “9”, whereas the number “9” was misclassified 100 instances as number “6”. However, with the introduction of the Centroid Difference method in Table 15, the misclassification rate was reduced to 0, respectively, for number “6” and “9”. Similarly, with the introduction of VF method, the misclassification rate of number

“8” predicted as character “B” was reduced to 15 from 45, and “B” predicted as “8” was reduced to 11 from 39. The combination use of Horizontal and Vertical Flip method helps to reduce the misclassification rate in other characters. The overall classification accuracy using the only Hu’s moments is 86.39% and, the accuracy obtained from the Hu’s moments with the proposed methods is 96.72%.

Since the accuracy of the proposed method is significantly high compared to the accuracy obtained from Hu’s moments, the features extracted using Hu’s moments along with the proposed methods were used to create a classifier for character recognition. As we can see from the classification matrix that, the proposed methods have a more significant impact on the classification of character (6, 9), (B, 8) and (A, 4). However, there are still some misclassifications generated by the random forest classifier. Therefore, to create a model for character recognition, we have proposed an XGBoost classifier to be used on our features.

Implementation of XGBoost classifier on Python

The term XGBoost stands for “eXtreme Gradient Boosting” is a decision tree-based ensemble learning algorithm developed by Chen and Guestrin (2016). The boosting algorithm aims to generate multiple weak classifiers and combine their predictions to form a strong rule. Unlike random forest classifier where the trees are built in the parallel fashion, the XGBoost algorithm built the trees in a sequential manner such that each tree aims to reduce the error of the previous tree. In the XGBoost classifier algorithm, parameter like learning rate can be tuned as per the requirement, and the early stopping can be used, which avoid the problem of overfitting. It also helps you to reduce the number of decision trees automatically. Figure 49 shows the diagrammatic representation of the XGBoost classifier.

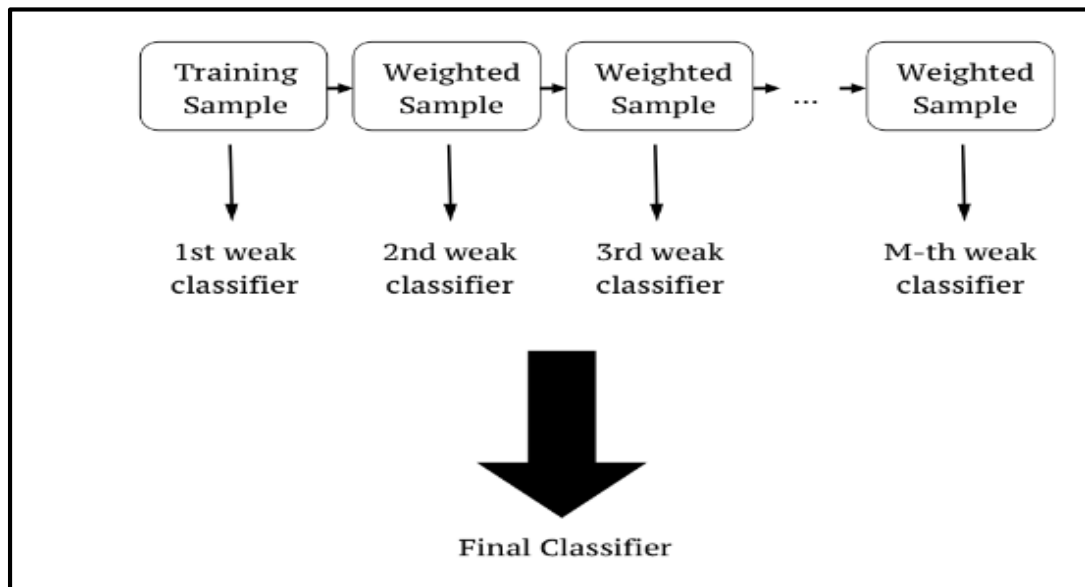


Figure 49 Example of XGBoosting algorithm

Source: Kumar, 2018

In the XGBoost algorithm, the datasets are assigned some weights which specify what is the probability of each record (row) getting selected by the decision tree during the training phase. In the first iteration, all the records in the datasets will get the same weights. After the training of the first decision tree, the model evaluates the test datasets. The misclassification generated in the first model is considered as the weak classifier. In the second decision tree, the weights of those records which were wrongly classified in the previous tree were updated. Similarly, the accuracy of the model generated by the second model was evaluated to identify the weak classifier, and the process continues depending on the number of decision trees. Finally, voting from each decision tree predicts the classes of the new data.

In the experiments, the extracted features stored in a CSV (Comma Separated Values) file were divided into training and testing sets. During the datasets split, an equal number of train-test features were taken to have an equal number of features in the training and testing sets. Since the value to be predicted (class label) contains a mixture of characters and numbers, the machine learning algorithm faces a problem during the training phase. Therefore, to create a class label in the same

format, the characters and numbers were encoded with the ASCII code. In the XGBoost classifier, a good number of decision trees defined in the classifier plays an essential role in obtaining high accuracy along with the learning rate. Along with the learning rate, early stopping features was also used, which monitors the performance of the model. The early stopping method stops the training procedure once the performance on the test dataset continues to remain same after a fixed number of iterations. By doing so, it helps to avoid overfitting and takes less time to train the model. Table 16 shows the summary of parameters used during the training of the classifier.

Table 16 Summary of parameters used in the XGBoost classifier with the accuracy

| Parameters | Feature extraction methods | |
|------------------------------|----------------------------|---------------------------------|
| | Hu's moments | Hu's moments + proposed methods |
| No. of features | 7 | 10 |
| No. of labels | 1 | 1 |
| % of train and test split | 80-20 | 80-20 |
| Number of trees (iterations) | 500 | 500 |
| Train accuracy | 98.28% | 99.92% |
| Test accuracy | 84.75% | 96.16% |
| Learning rate | 0.1 | 0.1 |
| Early stop | 275 | 194 |

In Table 16, the training accuracy generated by the classifier using only the Hu's moments is 98.28%, but the testing accuracy is 84.75%. This type of variation between the train and test accuracy will result in overfitting since the training accuracy is too high comparing to testing accuracy. Overfitting refers to a model that performs very well in the training datasets, but it fails to generalize the prediction on the new data. Therefore, the features extracted from the Hu's Moments

alone cannot be used as a character classifier to recognize the license plate characters. Hence, the combination of Hu's Moments along with the proposed methods (Centroid Difference, Vertical Flip, Horizontal Flip) gives the training accuracy of 99.92% and test accuracy of 96.16%. The XGBoost classifier trained on Hu's moments utilized 275 decision trees, whereas the proposed methods used 194 decision trees. The training time is reduced with the introduction of early stopping method in the classifier. The proposed feature extraction methods were far better than using only the Hu's Moments for the classifier. For evaluating the performance of the classifier, the confusion matrix or classification matrix is used to describe the performance of the model based on the test data in which the true values are known. The confusion matrix is used in measuring recall, precision, accuracy and f1-score. There are some terminologies to be known in the confusion matrix as shown in Table 17:

Table 17 Confusion matrix example

| | | Predicted Class | |
|--------------|-----|-----------------|----|
| | | YES | NO |
| Actual Class | YES | TP | FN |
| | NO | FP | TN |

- **True Positive (TP):** Correctly predicted values for each class
- **True Negative (TN):** Correctly rejected prediction for certain class
- **False Positive (FP):** Incorrectly identified predictions for certain class
- **False Negative (FN):** Incorrectly rejected predictions for certain class

1. Accuracy: It is the ratio of correctly predicted class to the total instances. This way of finding accuracy is best known when there is an equal number of instances in each class in the datasets. The equation to find the simple accuracy is shown in Equation 30:

$$Accuracy = \frac{TP}{TP + TN + FN + FP} \quad (30)$$

- 2. Precision:** Precision is the ratio between the correctly predicted positive observations to the total predicted positive observations. The precision is calculated from the confusion matrix using Equation 31:

$$Precision = \frac{TP}{TP + FP} \quad (31)$$

- 3. Recall:** A recall is a ratio between the correctly predicted positive observation to the observations in the actual class. The recall from the confusion matrix is calculated using Equation 32:

$$Recall = \frac{TP}{TP + FN} \quad (32)$$

- 4. F1-Score/Measure:** Since we are using an unbalanced number of instances in each class, finding simple accuracy may not be reliable to check the performance of the model. This is because classes having a high frequency of instances might generate high true positives and making the accuracy high. Therefore, metric F1-Score was used to compute the overall accuracy of the classifier since it takes care of an unbalanced number of instances in the class. F1-score is the weighted average of precision and recall. Unlike simple accuracy method, F1 uses false positives and false negatives rate to find the F1-score. Equation 33 is used to compute the value of F1-score.

$$F1\ Score = 2 * \frac{Recall * Precision}{Precision + Recall} \quad (33)$$

The tabulation of the precision, recall, f1 score and the total number of test data used in the classifier is given in Table 18.

Table 18 Classification report showing precision, recall and f1-score

| Character | Hu's Moments | | | Proposed Methods | | | Frequency |
|-----------|--------------|--------|----------|------------------|--------|----------|-----------|
| | precision | recall | f1-score | precision | recall | f1-score | |
| 0 | 0.92 | 0.9 | 0.91 | 0.98 | 0.92 | 0.95 | 51 |
| 1 | 0.99 | 0.96 | 0.97 | 0.94 | 1 | 0.97 | 75 |
| 2 | 0.86 | 0.77 | 0.81 | 0.97 | 0.98 | 0.98 | 62 |
| 3 | 0.96 | 0.94 | 0.95 | 1 | 0.98 | 0.99 | 53 |

Table 18 (cont.)

| Character | Hu's Moments | | | Proposed Methods | | | Frequency |
|-----------|--------------|-------------|-------------|------------------|-------------|-------------|-----------|
| | precision | recall | f1-score | precision | recall | f1-score | |
| 4 | 0.85 | 0.8 | 0.83 | 0.98 | 0.96 | 0.97 | 51 |
| 5 | 0.85 | 0.87 | 0.86 | 1 | 0.96 | 0.98 | 47 |
| 6 | 0.62 | 0.54 | 0.57 | 1 | 1 | 1 | 54 |
| 7 | 0.89 | 0.96 | 0.92 | 0.98 | 0.93 | 0.95 | 57 |
| 8 | 0.82 | 0.69 | 0.75 | 0.88 | 0.98 | 0.93 | 59 |
| 9 | 0.53 | 0.66 | 0.59 | 1 | 1 | 1 | 59 |
| A | 0.84 | 0.94 | 0.89 | 0.96 | 0.96 | 0.96 | 52 |
| B | 0.77 | 0.75 | 0.76 | 0.91 | 0.94 | 0.92 | 77 |
| C | 0.91 | 0.95 | 0.93 | 0.96 | 0.98 | 0.97 | 56 |
| D | 0.85 | 0.9 | 0.88 | 0.98 | 0.88 | 0.93 | 59 |
| G | 0.89 | 1 | 0.94 | 0.95 | 0.95 | 0.95 | 58 |
| P | 0.94 | 0.91 | 0.92 | 0.98 | 1 | 0.99 | 64 |
| T | 0.96 | 0.86 | 0.91 | 0.93 | 0.91 | 0.92 | 56 |

In Table 18, since most of the classes have recall higher than 90%, it shows that there are low false-negative rates. Similarly, the precision of each class is high, which relates to the low false-positive rates. In the table, precision and recall rate generated by Hu's moments for license plate characters (6, 9), (B, 8) and "4" is very low resulting in high false-positive and false-negative rates. However, with the introduction of proposed methods along with the Hu's moments, the precision and recall for each character were high, resulting in low false-positives and false-negatives. The respective confusion matrix generated by the test sets for Hu's Moments and the Proposed Methods (Hu's Moments + Centroid Difference Method + Flip Method) is shown in Table 19 and Table 20.

Table 19 Confusion matrix of a test set using Hu's moment

| | | PREDICTED | | | | | | | | | | | | | | | | |
|--------|---|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | G | P | T |
| ACTUAL | 0 | 46 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |
| | 1 | 0 | 72 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 2 | 0 | 0 | 48 | 0 | 0 | 1 | 2 | 0 | 0 | 9 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| | 3 | 0 | 0 | 0 | 50 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 |
| | 4 | 0 | 1 | 0 | 0 | 41 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 2 | 1 |
| | 5 | 0 | 0 | 2 | 0 | 0 | 41 | 0 | 0 | 0 | 1 | 0 | 3 | 0 | 0 | 0 | 0 | 0 |
| | 6 | 0 | 0 | 1 | 0 | 0 | 0 | 29 | 0 | 0 | 23 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 55 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| | 8 | 3 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 41 | 0 | 0 | 9 | 0 | 3 | 0 | 0 | 0 |
| | 9 | 1 | 0 | 3 | 0 | 0 | 0 | 15 | 0 | 0 | 39 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | A | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 49 | 0 | 0 | 0 | 0 | 0 | 0 |
| | B | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 8 | 1 | 0 | 58 | 0 | 6 | 2 | 0 | 0 |
| | C | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 53 | 0 | 0 | 0 | 0 |
| | D | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 53 | 2 | 0 | 0 |
| | G | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 58 | 0 | 0 |
| | P | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 58 | 0 |
| | T | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 48 |

Table 20 Confusion matrix of a test set using proposed methods

| | | PREDICTED | | | | | | | | | | | | | | | | |
|--------|---|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | G | P | T |
| ACTUAL | 0 | 47 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| | 1 | 0 | 75 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 2 | 0 | 1 | 61 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 | 52 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| | 4 | 0 | 0 | 0 | 0 | 49 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 1 | 0 | 1 | 0 | 0 | 45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 58 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | A | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 50 | 0 | 0 | 0 | 0 | 1 | 0 |
| | B | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 72 | 0 | 0 | 0 | 0 | 0 |
| | C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 55 | 0 | 1 | 0 | 0 |
| | D | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 5 | 0 | 52 | 1 | 0 | 0 |
| | G | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 55 | 0 | 0 |
| | P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 64 | 0 |
| | T | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 51 |

The proposed methods for feature extractions have a more significant impact in distinguishing license plate character (6, 9), (B, 8), (A, 4) and (T, 7). The log loss and the classification error rate generated by the XGBoost Classifier is shown in Figure 50 and Figure 51. In the two graphs, there is a significant distinction in the convergence of log loss between the test and train. The error rate of classification in Figure 50 is relatively high compared to Figure 51.

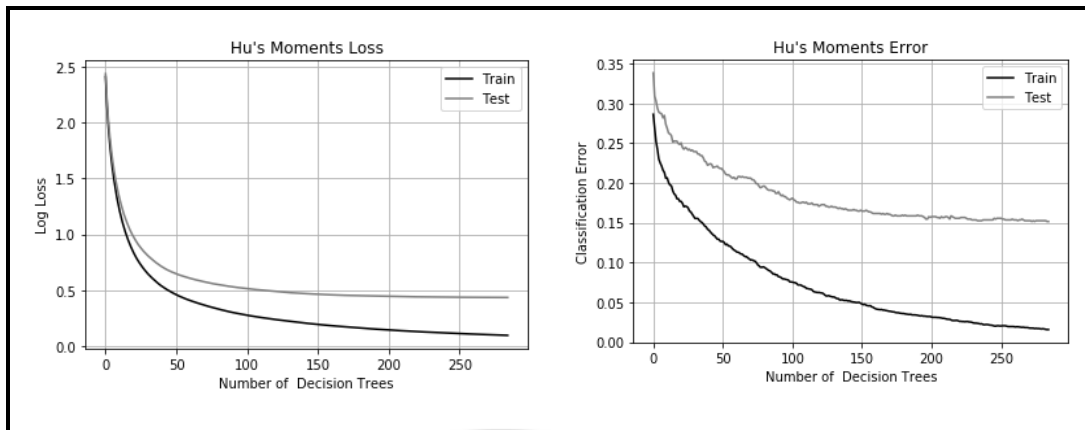


Figure 50 Log loss and classification error for Hu's moments

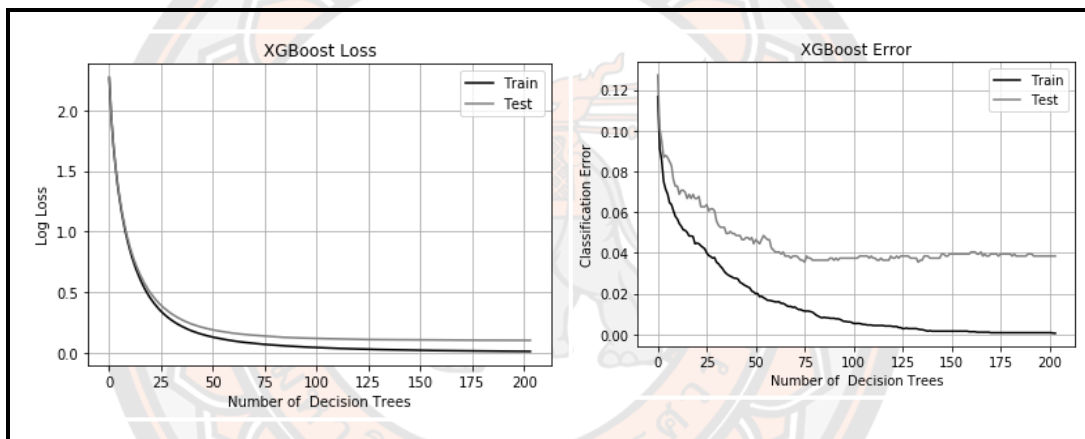


Figure 51 Log loss and classification error for proposed methods

Performance evaluation of character recognition

Since all the three types of license plates have letter “B” as the first character, the model does not classify the first character. The model classifies the license plate characters starting from 2nd place. In the segmentation phase, 1012 localized license plates were feed into the system, of which only 973 license plates passed through the segmentation phase. Now in this step, all the license plates which passed the segmentation phase were given as the input to recognition module to recognize the individual segmented characters. Before classifying the individual characters, a noise removal, as discussed in Chapter III, was applied to remove the

unwanted characters. The overall recognition accuracy generated by the classifier for individual license plate was 92.9%, as shown in Table 21.

Table 21 Results from character recognition

| | |
|------------------------------|-------|
| Total license plates | 973 |
| Correct recognition | 904 |
| Incorrect recognition | 69 |
| Recognition Accuracy | 92.9% |

The incorrect recognition of license plate characters is due to presence of characters written in difference fonts. In our approach, we have not deskew the license plate and this is a another reason for having incorrect recognition. To have a robust classifier, different character data need to be introduced in the classifier and trained using deep learning algorithm. Figure 52 shows the correctly recognized characters for three types of license plates.



Figure 52 Experimental results for character recognition for three types of license plates

The classifier was tested to recognize the license plate characters under the shadow and light regions. As, we can notice from the results that the proposed classifier robustly recognized the license plate characters, as shown in Figure 53.



Figure 53 Experimental results for character recognition under shadow and light region

Also, the performance of the system was tested to evaluate the recognition of license plates in the night lighting condition. The results depicted in Figure 54 proves that the system can recognize the license plate character with 100% accuracy.



Figure 54 Experimental results for character recognition in night view

Apart from night and shadow region, the system was checked by recognizing the license plate number on the skewed car image. In Figure 55, the front view of the license plate is a bit skewed. No skew correction is performed in this project. It is expected that the performance of the system can be increased if the skew correction is done before recognizing the characters. The proposed system can recognize all the characters if the plate is rotated up to 10 degrees.



Figure 55 Experimental results for character recognition in skewed image

The system was also tested with a license plate which is partially blocked by the vehicle spare tyres or crash guard and it was found that the system can recognize all the license plate characters with high accuracy as shown in Figure 56.



Figure 56 Experimental results for character recognition in partially blocked license plate

However, the proposed character classifier fails to recognize those license plate written in a different font. In most of the Bhutanese license plates, “ARIAL” font has been used, but while testing the accuracy, there are some images which contains license plates written in other fonts as shown in Figure 57.



Figure 57 Experimental results for wrong recognition of license plate characters

Overall performance comparison

Table 22 compares the performance of the proposed methods in our system with methods reviewed in Chapter II. The accuracy rate of the system is highly dependent on the number of test set used, and higher accuracy does not necessarily mean the method is better. In comparison, the number of test images and the methods used are included for each technique. The accuracy of each method is included, as described in the referenced papers.

The comparison shows that our proposed method has better accuracy among most of the methods. In vehicle detection, our YOLO approach vehicle detection achieves 98.5%, which is better than the background subtraction method discussed by (Vasu, 2010). Regarding license plate localization, the YOLO based proposed method achieved better accuracy of 97.9 than the other methods. However, in character segmentation, our proposed method outperforms over some methods proposed by Patel et al. (2013) and Sarfraz et al. (2003). In character recognition, the proposed method is better than the other two methods discussed by Patel et al. (2013) and C. H. Moreno, N. Trejo, M. Soto, and B. M. Montiel (2018). The overall accuracy given by each proposed methods is much higher than the methods discussed in the literature.

- VD: Vehicle Detection
- LPL: License Plate Localization
- CS: Character Segmentation
- CR: Character Recognition

Table 22 Performance comparison with other methods

| Methods (Authors) | VD | LPL | CS | CR | Total |
|---|------------|------------|------------|------------|---------------|
| | (%) | (%) | (%) | (%) | Images |
| Background subtraction (Vasu) | 90-93 | - | - | - | - |
| Deep Neural Networks (Soin & Chahande) | 100 | - | - | - | CIFAR-10 |

Table 22 (cont.)

| Methods (Authors) | VD | LPL | CS | CR | Total |
|---|------------|--------------|--------------|--------------|--------------------|
| | (%) | (%) | (%) | (%) | Images |
| Sobel Operator, vertical projection, template matching (Sarfraz et al.) | - | 96.22 | 94.04 | 95.24 | 610 |
| Colour features (Ashtari et al.) | - | 96.6 | - | - | - |
| Texture based (Hong-ke et al.) | - | 96 | - | - | - |
| Connected component analysis (Nukano et al.) | - | - | 98.4 | - | 128 |
| Projection profile (Duan et al.) | - | 97.61 | - | - | 795 |
| Template matching based on edge Hausdorff distance (Shuang-tong & Wen-ju) | - | - | - | 98 | 3000 characters |
| Sobel operator, vertical projection, ANN (Patel et al.) | - | 90 | 85 | 84 | 150 |
| Correlation factor (C. H. Moreno et al.) | - | - | - | 91.42 | |
| Proposed method | 98.5 | 97.9 | 96.1 | 92.9 | 1050 |

Note: *The accuracy less than the proposed methods are in bold face.*

CHAPTER V

CONCLUSION

In the first chapter, a detailed background about the study was discussed along with the definition of problem statements. The main aim of the study was to develop an ANPR system in Bhutan context and design a feature extraction method. The typical ANPR system consists of 3 stages: license plate, character segmentation and character recognition. In our approach, a vehicle detection was done to reduce the false positives while localizing the license plates. In the second chapter, detailed literature was conducted. This research investigates the use of YOLO state-of-the-art object detector for an automatic license plate localization after the detection of a vehicle. Unlike region proposal methods, a single convolutional network has been proposed to accomplish the given task. For evaluating the performance of the method, 1059 images were fed into the system and the proposed method achieved vehicle detection accuracy of 98.5% and license plate localization accuracy of 97.9%. After the localization of license plate, around 40% of the image was cropped from the top to remove the Bhutanese scripts and unwanted screw. This is mainly done since Bhutanese scripts do not play any significant role in identifying the vehicle. After the cropping, the license plates, a series of image preprocessing such as grayscale conversion, noise removal, contrast enhancement and thresholding were applied. For Bhutanese license plates, the thresholding output was not in the same format. Therefore “White pixel counting with inversion method” was proposed to have the same format of the binary image. After getting the required binary image, a connected component analysis was applied to segment the characters. All the localized license plates were evaluated to check the performance of the segmentation method and achieved an accuracy of 96.2%. For the creation of character classifier to recognition to characters, the features from the individual characters were extracted using Hu’s moments, centroid difference and flip method. The latter two methods were proposed to handle the characters having the same Hu’s moments. For training and testing the extract features, an XGBoost classifier was selected. It is an ensemble machine

learning algorithm which aims to reduce the error caused by the previous tree. The overall character recognition accuracy given by the system was 92.9%. Overall, the proposed methods in each stage have better accuracy than some of the techniques discussed in the literature review. The main contribution of this research are the development of feature extraction methods to differentiate characters having similar features and develop an algorithm for automatic localization of license plates after detecting a vehicle using single convolutional network. This research is first of its kind to use the Bhutanese license plate for the development of ANPR technology.

Limitation and future work

The system shows a satisfactory result in all the stage of the ANPR; however, there are certain limitations since it was developed with certain assumptions. The limitations are as follows:

1. The smartphone was used in data collection since there are no records of license plates.
2. Fewer datasets were used for training the YOLO algorithm.
3. Only 17 characters were used in the classifier.
4. In character recognition, we have hand-engineered the features to create a classifier.
5. The system is having difficulty in recognizing the characters written in other fonts.
6. No character skew correction is implemented.

Therefore, we want to give a suggestion for future work as follows:

1. In future, the datasets need to be collected using a high-resolution camera to have better accuracy.
2. The training dataset for the YOLO should be increased to have more robustness in detecting vehicle and license plates.
3. For the designing of character classifier, all the 26 alphabets and 10 digits need to be included so that a vehicle registered with new characters will be easily identified.

4. A character classifier should be designed using a convolutional neural network since it can learn the features without human intervention.
5. More characters written in different fonts need to be collected to create a classifier.
6. To have high recognition accuracy, the license plate skew correction needs to be implemented.



REFERENCES

- Alexey. (2016). Windows and Linux version of Darknet Yolo v3 & v2 Neural Networks for object detection (Tensor Cores are used): AlexeyAB/darknet. from <https://github.com/AlexeyAB/darknet>
- Ashtari, A. H., Nordin, M. J., & Fathy, M. (2014). An Iranian license plate recognition system based on color features. *IEEE transactions on intelligent transportation systems*, 15(4), 1690-1705.
- Bhutan-Transportation and Communications. (n.d.). Retrieved 26 December, 2019, from <http://www.country-data.com/cgi-bin/query/r-1497.html>
- Chen, T., & Guestrin, C. (2016). *Xgboost: A scalable tree boosting system*. Paper presented at the Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.
- Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2013). Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(2), 311-325. doi: 10.1109/TCSVT.2012.2203741
- Duan, T. D., Du, T. H., Phuoc, T. V., & Hoang, N. V. (2005). *Building an automatic vehicle license plate recognition system*. Paper presented at the Proc. Int. Conf. Comput. Sci. RIVF.
- Duan, T. D., Duc, D. A., & Du, T. L. H. (2004). *Combining Hough transform and contour algorithm for detecting vehicles' license-plates*. Paper presented at the Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.
- History of ANPR. (n.d.). Retrieved 26 December, 2019, from <http://www.anpr-international.com/history-of-anpr/>
- Hong-ke, X., Fu-hua, Y., Jia-hua, J., & Huan-sheng, S. (2005, 5-8 Dec. 2005). *A New Approach of the Vehicle License Plate Location*. Paper presented at the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05).
- Hsieh, C.-T., Juan, Y.-S., & Hung, K.-M. (2005). *Multiple license plate detection for complex background*. Paper presented at the 19th International Conference on

Advanced Information Networking and Applications (AINA'05) Volume 1
(AINA papers).

- Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2), 179-187.
- Hu, X., Xu, X., Xiao, Y., Chen, H., He, S., Qin, J., & Heng, P.-A. (2018). SINet: A scale-insensitive convolutional neural network for fast vehicle detection. *IEEE transactions on intelligent transportation systems*, 20(3), 1010-1019.
- Iglewicz, B., & Hoaglin, D. (1993). *Volume 16: How to Detect and Handle Outliers: Milwaukee, Wis. : ASQC Quality Press, [1993] ©1993.*
- Jamtsho, Y., Riyamongkol, P., & Waranusast, R. (2019a). *Bhutanese License Plate Recognition Using Hu's Moments and Centroid Difference*. Paper presented at the 2019 4th International Conference on Information Technology (InCIT).
- Jamtsho, Y., Riyamongkol, P., & Waranusast, R. (2019b). Real-time Bhutanese license plate localization using YOLO. *ICT Express*. doi: doi.org/10.1016/j.icte.2019.11.001
- Jia, W., Zhang, H., & He, X. (2007). Region-based license plate detection. *Journal of Network and computer Applications*, 30(4), 1324-1333.
- Jianbo, S., & Tomasi. (1994, 21-23 June 1994). *Good features to track*. Paper presented at the 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.
- Jin, L., Xian, H., Bie, J., Sun, Y., Hou, H., & Niu, Q. (2012). License plate recognition algorithm for passenger cars in Chinese residential areas. *Sensors*, 12(6), 8355-8370.
- Kanaparthy, P. K. (2012). *Detection and recognition of US speed signs from grayscale images for intelligent vehicles*. University of Toledo.
- Kaur, S., & Kaur, S. (2014). An Efficient Approach for Automatic Number Plate Recognition System under Image Processing. *International Journal of Advanced Research in Computer Science*, 5(6).
- Kawamura, R. (n.d.). RectLabel. from <https://rectlabel.com/>
- Khalid, Z., Mazoul, A., & Ansari, M. E. (2011). A new vehicle detection method. *International Journal of Advanced Computer Science and Applications*, 1(3).

- Koval, V., Turchenko, V., Kochan, V., Sachenko, A., & Markowsky, G. (2003). *Smart license plate recognition system based on image processing using neural network*. Paper presented at the Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2003. Proceedings.
- Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.
- Kumar, A. (2018). XGBoost Algorithm. from <https://acadgild.com/blog/xgboost-python>
- Lee, E. R., Kim, P. K., & Kim, H. J. (1994). *Automatic recognition of a car license plate using color image processing*. Paper presented at the Proceedings of 1st International Conference on Image Processing.
- Miyamoto, K., Nagano, K., Tamagawa, M., Fujita, I., & Yamamoto, M. (1991). *Vehicle license-plate recognition by image analysis*. Paper presented at the Proceedings IECON '91: 1991 International Conference on Industrial Electronics, Control and Instrumentation.
- Moreno, C. H., Trejo, N., Soto, M., & Montiel, B. M. (2018). *License Plates Recognition of Mexican Private Vehicles*. Paper presented at the COMPUTATION TOOLS 2018 : The Ninth International Conference on Computational Logics, Algebras, Programming, Tools, and Benchmarking.
- Moreno, C. H., Trejo, N., Soto, M., & Montiel, B. M. (2018). *License Plates Recognition of Mexican Private Vehicles*. Paper presented at the COMPUTATION TOOLS 2018 : The Ninth International Conference on Computational Logics, Algebras, Programming, Tools, and Benchmarking.
- National Statistics Bureau of Bhutan. (2017). 2017 Population and Housing Census of Bhutan. from http://www.nsb.gov.bt/publication/files/PHCB2017_national.pdf
- Neubeck, A., & Gool, L. V. (2006). *Efficient Non-Maximum Suppression*. Paper presented at the 18th International Conference on Pattern Recognition (ICPR'06).
- Nukano, T., Fukumi, M., & Khalid, M. (2004). *Vehicle license plate character recognition by neural networks*. Paper presented at the Proceedings of 2004 International Symposium on Intelligent Signal Processing and Communication

- Systems, 2004. ISPACS 2004.
- Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1), 62-66.
- Patel, R. P., Patel, N. M., & Brahmabhatt, K. (2013). Automatic license plate recognition. *International Journal of Computer Science and Mobile Computing (IJCSMC)*, 2(4), 285-294.
- Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., . . . Zuiderveld, K. (1987). Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3), 355-368.
- Prates, R., Cámara-Chávez, G., Schwartz, W. R., & Menotti, D. (2014). Brazilian license plate detection using histogram of oriented gradients and sliding windows. *arXiv preprint arXiv:1401.1990*.
- Qureshi, M., Asif, K., & Bashir, K. (2011). ANPRSRE: automatic number plate recognition system for real-time environments. *International Journal of Computer Science and Telecommunications*, 2(2), 14-19.
- Redmon, J. (2013). Darknet: Open Source Neural Networks in C. from <https://pjreddie.com/darknet/>
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Redmon, J., & Farhadi, A. (2017). *YOLO9000: better, faster, stronger*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Road Safety and Transport Authority. (2017). Annual Report For Financial Year (2017-2018). Retrieved May 26, 2019, from https://drive.google.com/file/d/1PBhs5CB1JMS97OA4Pn_UZjdcwTVSubvt/view?usp=sharing&usp=embed_facebook
- Road Safety and Transport Authority. (2018). Road Safety and Transport Regulations (1999). Retrieved June 3, 2019, from https://drive.google.com/file/d/1c1zgh7Sm4G4jDfomxEfAad1kEPA13Zv7/view?usp=sharing&usp=embed_facebook

- Rosenfeld, A., & Pfaltz, J. L. (1966). Sequential operations in digital picture processing. *J. ACM*, 13(4), 471-494.
- Sarfraz, M., Ahmed, M. J., & Ghazi, S. A. (2003). *Saudi Arabian license plate recognition system*. Paper presented at the 2003 International Conference on Geometric Modeling and Graphics, 2003. Proceedings.
- Shi, X., Zhao, W., & Shen, Y. (2005). *Automatic license plate recognition system based on color image processing*. Paper presented at the International Conference on Computational Science and Its Applications.
- Shuang-tong, T., & Wen-ju, L. (2005). *Number and letter character recognition of vehicle license plate based on edge Hausdorff distance*. Paper presented at the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05).
- Soin, A., & Chahande, M. (2017). *Moving vehicle detection using deep neural network*. Paper presented at the 2017 International Conference on Emerging Trends in Computing and Communication Technologies (ICETCCT).
- Sonawane, A. (2018). YOLOv3: A Huge Improvement. from https://medium.com/@anand_sonawane/yolo3-a-huge-improvement-2bc4e6fc44c5
- Thimphu Thromde. (n.d.). Multi Level car parking project. Retrieved January, 14, 2019, from <http://www.thimphucity.bt/projects/multi-level-car-parking-project>
- Tshering, P. S. (2017). Online bus ticket booking system launched. Retrieved April 6, 2019, from <http://www.bbs.bt/news/?p=71476>
- Vasu, L. (2010). *Effective step to real-time implementation of accident detection system using image processing*. Oklahoma State University.
- Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann.
- Yadav, V. (2017). Generating Anchor boxes for Yolo-like network for vehicle detection using KITTI dataset. from <https://medium.com/@vivek.yadav/part-1-generating-anchor-boxes-for-yolo-like-network-for-vehicle-detection-using-kittidataset-b2fe033e5807>
- Zekovich, S., & Tuba, M. (2013). Hu moments based handwritten digits recognition

algorithm. *Recent Advances in Knowledge Engineering and Systems Science*.

Zheng, D., Zhao, Y., & Wang, J. (2005). An efficient method of license plate location.

Pattern recognition letters, 26(15), 2431-2438.

