DEVELOPMENT OF A DEEP LEARNING BASED VEHICLE LICENSE PLATE DETECTION SCHEME

BY

BLESSING OLUCHI ILOKA

DEPARTMENT OF COMPUTER ENGINEERING

FACULTY OF ENGINEERING

AHMADU BELLO UNIVERSITY ZARIA, NIGERIA

NOVEMBER, 2018

DEVELOPMENT OF A DEEP LEARNING BASED VEHICLE LICENSE PLATE DETECTION SCHEME

BY

BLESSING OLUCHI ILOKA PGD (ABU)2014

P16EGCP8043

Iloka.blessing@yahoo.com

A DISSERTATION SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES, AHMADU BELLO UNIVERSITY, ZARIA

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF A

MASTER OF SCIENCE (M.Sc.) DEGREE IN COMPUTER ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

FACULTY OF ENGINEERING

AHMADU BELLO UNIVERSITY, ZARIA

NIGERIA

NOVEMBER, 2018

DECLARATION

I declare that this dissertation titled **"**Development of a Deep Learning Based Vehicle License Plate Detection Scheme**"** has been carried out by me in the Department of Computer Engineering, Ahmadu Bello University, Zaria. The information derived from literature has been duly acknowledged in the text and a list of references provided. No part of this dissertation was previously presented for another degree or diploma at this or any other institution.

Blessing Oluchi ILOKA _____ _____
(Student) Signature Date

CERTIFICATION

This Dissertation entitled "DEVELOPMENT OF A DEEP LEARNING BASED VEHICLE LICENSE PLATE DETECTION SCHEME" by Blessing Oluchi ILOKA meets the regulations governing the award of degree of Master of Science (M.Sc.) in Computer Engineering of the Ahmadu Bello University, and is approved for its contribution to knowledge and literary presentation.

Prof. M. B. Mu'azu
(Chairman, Supervisory Committee) _____ _____
Signature Date

Dr. E. A. Adedokun
(Member, Supervisory Committee) _____ _____
Signature Date

Prof. M. B. Mu'azu
(Head of Department) _____ _____
Signature Date

Prof. S. Z. Abubakar
(Dean, School of Postgraduate Studies) _____ _____
Signature Date

DEDICATION

This research work is dedicated to my father, the Late Mr. Emmanuel ILOKA.

ACKNOWLEDGEMENT

I am most grateful to God, whom I owe my life and all of my achievements including the realization of this dissertation.

I am deeply grateful to my supervisors, Prof. M. B. Mu'azu (Chairman) and Dr. E. A. Adedokun (Member), for their effort, guidance, and constant supervision they accorded me towards the successful completion of this work. This would not have been possible without their valuable participation and assistance. I pray Almighty God to continue to bless them.

I acknowledge with thanks all the lecturers of Department of Computer Engineering, Ahmadu Bello University namely: Dr Y. A. Sha'aban, Dr M. B. Abdulrasaq, Dr. A. T. Salawudeen, Dr. B. O. Sadiq, Dr. (Mrs) Basira Yahaya, Dr. I. J Umoh, Engr. Ibrahim Yusuf, I say a big thanks to you all.

I am also thankful to my friends and classmates, Nafisa Usman, Marshall Francis, Umar Abubakar, Alice Lawrence, Obari Johnson, Kabir Usman, Suleiman Usman, and all those whose names were not mentioned, I appreciate the contributions you all made to the success of this research work. Thank you all for guiding me through the different stages of this research work. I am equally grateful to all M.Sc. students in the Department.

I am sincerely grateful to my parents, Mr and Mrs Emmanuel Iloka. Your love, support, trust, and prayers brought me to this successful completion. I am eternally grateful to you both. To my cousin Gozie Nwakwesili and John Paul Anene, I say a big thank you for your support. And to all my siblings, I am really grateful for your love and belief. Thank you all for believing in me and for supporting me from the very beginning.

Blessing ILOKA

March 2019

ABSTRACT

This research developed a license plate and classification scheme using deep learning architecture which utilized transfer learning using pre-trained Convolutional Neural Network (CNN). The developed scheme used images obtained from Caltech dataset, Peking University VehicleID (PKU VehicleID) dataset and a developed dataset of vehicle licence plates from Ahmadu Bello University, Zaria called the ABU dataset. De-noising, downscaling operation and grayscale conversion was applied on the acquired image to reduce the cost inqured in using the original image. Sobel operation was performed to detect the edge of the pre-processed image. Edge density filtering and connected component analysis were used to extract and verify the region which constituted the licence plate number. AlexNet model pre-trained on ImageNet was used to extract features and classify the detected license plate candidate's regions. The performance of the developed scheme was evaluated on 150 images of each dataset of PKU VehicleID, Caltech, and ABU test images taken under different conditions. With the Caltech dataset the scheme achieved a precision rate of 85.10%, recall rate of 98.50%, and recognition accuracy of 98.08% while the PKU VehicleID dataset gave precision rate of 97.91%, recall rate of 97.91%, and accuracy of 100%. For the ABU dataset, the method obtained a precision rate of 95.8%, recall rate of 100%, recognition accuracy of 99.82%. The results for the developed deep learning-based scheme showed some performance improvements of 3.96% and 14.75% in the precision and recall rate, respectively, and 8.15% improvement in recognition rate, when compared with the existing scheme which utilized the edged based approach with SVM and achieved detection rate of 98% on the PKU VehicleID dataset, 90% on the Caltech dataset, and 96% on the ABU dataset in the presence of complex backgrounds and highly variable license plate patterns.

TABLE OF CONTENT

DECLARATION **III**

CERTIFICATION **IV**

DEDICATION **V**

ACKNOWLEDGEMENT **VI**

ABSTRACT **VII**

TABLE OF CONTENT **VIII**

LIST OF FIGURES **XI**

LIST OF TABLES **XIII**

LIST OF APPENDICES **XIV**

LIST OF ABBREVIATIONS **XV**

CHAPTER ONE

INTRODUCTION **1**

1.1 Background of Research 1

1.2 Significance of the Research 2

1.3 Statement of Problem 3

1.4 Aim and Objectives 3

CHAPTER TWO

LITERATURE REVIEW **5**

2.1 Introduction 5

2.2 Review of Fundamental Concepts 5

2.2.1 License plate detection and recognition 5

2.2.1.1 Image acquisition 6

2.2.1.2 Pre-processing 7
2.2.1.3 Image downscaling 7
2.2.1.4 Grayscale conversion 8
2.2.1.5 Image de-noising 9
2.2.2 Candidate extraction 11
2.2.2.1 Edge detection 11
2.2.2.2 Image binarization 14
2.2.2.3 Edge density filter 16
2.2.2.4 Candidate extraction using geometric properties of the license plate. 19
2.2.3 Datasets 19
2.2.4 Deep learning algorithm 20
2.2.4.1 Deep convolutional neural network (dCNN) 20
2.2.4.2 Convolutional neural network architecture(ConvNet) 21
2.2.4.3 Alexnet deep convolutional neural network 22
2.2.5 Character segmentation 24
2.2.6 Character recognition 24
2.2.7 Performance Evaluation 25
2.3 Review of Similar Work 26

CHAPTER THREE

MATERIALS AND METHODS 34
3.1 Introduction 34
3.2 Materials 34
3.3 Methodology 34
3.4 Development of ABU Vehicle Image Dataset 35
3.4.1 Image acquisition 36
3.4.2 Pre-processing the acquired image 37
3.4.2.1 Down scaling of the image 38
3.4.2.2 Converting RGB to grayscale Images 38
3.4.2.3 Noise elimination 38
3.4.3 Collation of publicly available license plate dataset 39
3.4.3.1 Caltech vehicle dataset 39
3.4.3.2 Pekins university dataset (PKU) 40
3.5 Development of the Deep Learning Based License Plate Detection Scheme. 41
3.5.1 Edge detection 44
3.5.2 Candidate's region localization 44
3.5.3 Verify the candidate region using connected component analysis (CCA). 45

3.5.4 Determination of the final License plate candidate region 46
3.5.5 Feature extraction and fine-tuning 47
3.5.6 Modification of the classification and output layer 47
3.5.7 Transfer learning and retraining of the model 47
3.5.8 Classification of the output 50

CHAPTER FOUR

RESULTS AND DISCUSSIONS 54
4.1 Introduction 54
4.2Grayscale Conversion of Images for Caltech, PKU VehicleID and ABU Images 54
4.3 Sobel Edge Detection on Caltech, PKU VehicleID and ABU Image 54
4.4 Candidate Result Obtained by the Edge Density Filter 55
4.5 Detection Result Obtained by the Developed LPD Approach on Caltech, PKU VehicleID and ABU Images 56
4.5.1 Detection result on the Caltech dataset 56
4.5.2 Detection result on the PKU VehicleID dataset 57
4.5.3 Detection result on the PKU dataset. 58
4.6 Extracted License Plate Number of Caltech, PKU VehicleID and ABU Images 59
4.7 Analysis of the Developed License Plate Detection 60
4.7.1 Detection performance on the dataset 60
4.7.2 Failed detection 61
4.7.4 Evaluating the precision, recall and classification accuracy of the scheme 64
4.8 Result of Detection on ABU Images Considering Various Environmental Conditions 65

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS 67
5.1 Summary of Findings 67
5.2 Conclusion 67
5.4 Significant Contributions 68
5.5 Recommendations for Further Work 68

REFERENCES 70

LIST OF FIGURES

Figure 2.1: Block Diagram of a Typical License Plate Recognition System 6

Figure 2.2: Flow Chart of the Edge Density Filter 18

Figure 2.3: General Overview of CNN Model 21

Figure 2.4: ALEXNET Archetecture **Error! Bookmark not defined.**

Figure 3.1: Sample of the ABU Dataset 37

Figure 3.2: Samples of the Caltech Dataset 40

Figure 3.3: Sample of the PKU VehicleID Dataset 41

Figure 3.4: Flow Chart of the Developed Scheme 43

Figure 3.5: Snippet of the Downscale Operation 38

Figure 3.6: Snippet of Median Filter 39

Figure 3.7: Snippet of code of a Sobel Edge Detector 44

Figure 3.8: Snippet code for the Connected Component Analysis (CCA) 46

Figure 3.9: Positive and Negative Training Samples 48

Figure 3.1: Sample of the ABU Dataset 37

Figure 3.2: Samples of the Caltech Dataset 40

Figure 3.3: Sample of the PKU VehicleID Dataset 41

Figure 3.4: Flow Chart of the Developed Scheme 43

Figure 3.5: Snippet of the Downscale Operation 38

Figure 3.6: Snippet of Median Filter 39

Figure 3.7: Snippet of code of a Sobel Edge Detector 44

Figure 3.8: Snippet of code for the Connected Component Analysis (CCA) 46

Figure 3.9: Positive and Negative Training Samples 48

Figure 3.10: Test images and their predicted labels 52

Figure 3.11: Confusion Matrix Plot 52

Figure 4.1: Result of the Converted Grayscale Images from the Dataset 54

Figure 4.2: Sobel Edge Detected Images 55

Figure 4.3: Edge Density Filtered Images. 55

Figure 4.4: Detection Result on Caltech Dataset 56

Figure 4.5: Detection Result on PKU VehicleID Dataset 57

Figure 4.6: Detection Result on ABU Dataset 58

Figure 4.7: Extracted Caltech License Plate Number 59

Figure 4.8: Extracted PKU VehicleID License Plate Number 59

Figure 4.9: Extracted ABU License Plate Number 60

Figure 4.10: Samples of Images that Resulted in Failed Detection 62

Figure 4.11: Confusion matrix of the test images 60

Figure 4.12: Images of Failed Detection 66

LIST OF TABLES

Table 3.1: Description of the ABU Vehicle Datasets 36

Table 4.1: Result of Detection Rate 61

Table 4.2: Precision and Recall Rate using the Existing Scheme on the Dataset 64

Table 4.3: Precision and Recall Rate using the Developed Scheme on the Dataset 65

Table 4.4: Result of the Detection Rate on ABU Dataset 65

LIST OF APPENDICES

APPENDIX A: ABU Dataset 73

APPENDIX B1: M-File for Pre-Processing Operation 76

APPENDIX B2: M-File for Connected Component Labeling 77

APPENDIX B3: M-File for Edge Density Filter 78

APPENDIX B4: M-File for Candidates Region Extraction 80

APPENDIX B5: M-File for Transfer Learning and Training using ALEXNET 75

LIST OF ABBREVIATIONS

Acronyms	Definition
ANN	Artificial Neural Network
ALEXNET	Alex Network
BP	Back Propagation
CalTech	California institute of technology
CNN	Convolutional Neural Network
dCNN	Deep Convolutional Neural Network
FN	False Negative
FP	False Positive
GPU	Graphic Processing Unit
ITS	Intelligent Transport System
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
LP	License plate
LPD	License plate Detection
LPDS	License Plate Detection Scheme
OCR	Optical Character Recognition
PKU	Pekins University
RELU	Rectified Linear Unit
RGB	Red-Green-Blue
ROI	Region of Interest
SVM	Support Vector Machine
TP	True Positive

TN	True Negative
VGGNET	Visual Geometric Group Network

CHAPTER ONE
INTRODUCTION

1.1 Background of Research

Advancement in intelligent transportation systems, has attracted considerable research interests in computer vision (Yuan *et al.,* 2016). This is because of its application in areas like vehicle management, electronic payment system (toll collection in express ways and parking fees payment), access control for monitoring area with limited accessibility like embassies, factories, military barracks, etc. and for identifying lost or stolen vehicles, border control and road traffic monitoring etc. (Du *et al.*, 2013).

The first licensed plate detection system was developed in 1976 at the police scientific development branch in the United Kingdom (UK) (Nguwi & Lim, 2015). At that time, the functions of license plate detection system were very limited. The essence of number plate detection is to apprehend unlicensed and auto thefts (Jenkins, 2007). In 2007, the automatic license plate recognition (ALPR) system was incorporated into the red-light camera in the United States of America (USA) to apprehend drivers whose vehicles drove past the red traffic lights. The offender's car plate information is captured by the camera and processed by the automatic license plate recognition (ALPR) (Jenkins, 2007).

License plate recognition system is divided into two component parts, detection and recognition (Zhao e*t al*., 2011). Detection is the ability to localize the license plate and generate a suitable bounding boxes that encompasses the detected license plate, while plate recognition aims to identify the characters depicted within the bounding boxes and classify it as a license plate. License plate detection and recognition are two separate processes, research on these two schemes are always been performed separately. Different algorithms

have been developed and applied to the two processes (Nguyen *et al.*, 2015). License plate detection is the most important aspect of the license plate recognition system, because the accuracy of the recognition depends on the detection stage (Zhao e*t al*., 2011). However, license plate recognition system is required for real time applications thus high detection rate is paramount in order to meet the requirements for such real time applications.

Although many algorithms have been proposed for license plate detection in the past two decades, some of which requires sophisticated camera to produce high quality images, demand vehicles to slowly pass a fixed access gate or even at a complete stop (Nguwi & Lim, 2015). All this conditions is to achieve a clearer view of the object. Despite all these, it is still a challenging task to detect license plates in an open and noisy environment. The problem becomes more complex, especially when the license plate number are not standardized, such as which may be faded, partially occluded by dirt and taken under different environmental conditions (Fomani & Shahbahrami 2017). Detecting these plates with the traditional methods may result in many false positives (Li & Shen, 2016). To tackle this problem, state of the art deep learning technique was explored.

1.2 Significance of the Research

License plate detection system is required for real time applications such as access control, traffic management and electronic toll system. Hence such real time applications require a license plate detection scheme that will accurately detect the license plate at a faster rate irrespective of the environmental condition and produce a fast, cost effective and highly accurate recognition system. This can be achieved by using current best image processing and deep learning technology.

1.3 Statement of Problem

Detecting plate number from an image is a very difficult task as the image contains a lot of noise which reduces its quality. These noises are as a result of the image taken under different weather conditions and lighting conditions such as uneven illuminations and presence of background clutters. Conventional approaches which mainly rely on certain morphological operations have been largely used for detecting the license plate position but have limitations in real time applications due to their time-consuming nature. Some of the techniques such as texture and colour based techniques are also not robust to detect noisy or corrupted images.

For the purpose of effectively and accurately detecting the license plate from any image irrespective of the complexity of the background, there is a need to achieve at least a human level accuracy. This can be achieved by using a deep CNN framework that utilizes a deep feature extraction technique via transfer learning approach. This is necessary in order to improve the detection rate while reducing the rate of achieving false positive results.

1.4 Aim and Objectives

The aim of this research work is to develop a deep learning based license plate detection scheme. The objectives of the research are:

1. To develop a dataset of vehicles images in Ahmadu Bello University (ABU) called the ABU dataset
2. To develop a dCNN-based licence plate detection system
3. To test the functionality of the developed scheme on the ABU, Caltech and PKU Vehicle ID.

4. To evaluate the performance of the developed scheme by comparing with the work of Yuan *et al*., 2016

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

This chapter reviews the fundamental concepts and similar works, which are relevant to the scope of this research work.

2.2 Review of Fundamental Concepts

Concepts that are fundamental to license plate recognition systems such as license plate detection, edge detection, edge density filter, license plate candidate extraction, deep learning, convolutional neural network, character segmentation and recognition etc. are discussed.

2.2.1 License Plate Detection and Recognition

License plate detection is the most important part of the license plate recognition. The performance of the system depends on the accuracy of the detection (Agarwal & Goswami, 2016). The input to this stage is a car image and the output is the portion of the image containing the potential license plate. The license plate is assumed to be found in a small region of the entire image and as such instead of processing the entire pixels of the image, the plate region can be segmented and localized by combining various algorithms based on the features of the license plate. These features include color features, texture features, geometric features, etc. (Zhao *et al*., 2011). Therefore, the system will process only those pixels that contain these features rather than processing every pixel. The target is to mark an area with maximum probability of containing a number plate and validate for the true license plate. A license plate detection system should be able to detect license plate numbers from

different countries and accurately locate the exact location of the license plate irrespective of the country's plate number format. Despite the different license plate detection algorithms proposed in literature, it is still a challenging task to detect license plate from natural environments which contain grass clutters, non-uniform illuminations and reflection glares (Yuan *et al.*, 2016).

Figure 2.1 shows a block diagram of a typical license plate recognition system. The system consists of the license plate detection module, where the license plate region is identified and the license plate recognition module where the number plate characters are segmented and recognize.

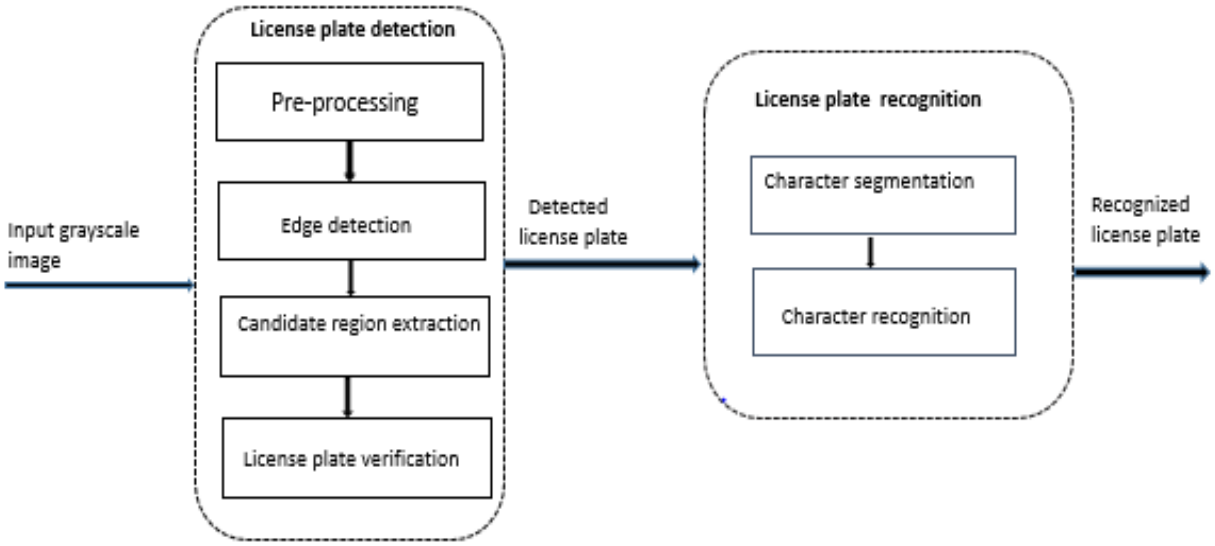


Figure 2.1: Block Diagram of a Typical License Plate Recognition System (Kim et al.,2018).

2.2.1.1 Image Acquisition

The first stage of license plate detection system is image acquisition. There are publicly available databases such as the Caltech dataset developed by the Computational Vision group at California Institute of Technology Caltech and the Pekins University VehicleID (PKU VehicleID) dataset developed by the National Engineering Laboratory for Video Technology (NELVT) (Yuan *et al.*, 2016).

The quality of the image is also very important as high quality images increase the accuracy of detection rate and recognition accuracy (Du, *et al*., 2013). However, the proposed detection scheme is expected to perform creditably well irrespective of the image quality.

2.2.1.2 Pre-Processing

Pre-processing is an image enhancement procedure that is applied to an image in order to improve its quality so as to enhance the detection and recognition rates of a detection algorithm. Before the license plate detection (LPD) stage, several pre-processing techniques are applied to remove shadows and noise in the image in order to improve the quality of images. The quality of the image is a major factor that guarantees the efficiency and accuracy of the license plate recognition system (Du, *et al*., 2013). There are different pre-processing steps in computer vision and image processing such as image resizing, gray scale conversion, histogram equalization and noise filtering etc. (Du, *et al*., 2013).

2.2.1.3 Image Downscaling

Images are usually captured at very high resolutions in order to ensure that small sizes of license plates with smaller characters are recognized using computer vision algorithms. The high resolution at which an image is captured tends to increase the computational complexity and consequently the processing time of the algorithm. To balance this, the size of the original image is usually resized in way that the detection rate is not affected (Yuan *et al.,* 2016).

The assumptions that the width of the licence plate is usually greater than its height and the characters on the license plates are printed horizontally influence the scaling factor, which is determined by the equation (Yuan *et al.,* 2016):

$$w_c = {wi}/{dw} \tag{2.1}$$

$$h_c = {hi}/{dh} \tag{2.2}$$

where w_i and h_i represent the width and height of the original image, w_c and h_c represent the downscaled dimensions, and dw and dh are the scale factor, subject to the condition $dh < dw$.

2.2.1.4 Grayscale Conversion

Grayscale images are images which the colour information from each channel is removed leaving only the luminance (brightness) information. This is why the maximum luminance is white and zero luminance is black, white everything in-between is a shade of gray. Grayscale conversion is used to convert the coloured images to the gray scale image by calculating the value of the gray level (Fisher *et al.*, 2003). This technique is very important in license plate detection system because using the original coloured image increases the processing time of the algorithm and consumes more space on the system memory. On the other hand, in grayscale image the red-green-blue (RGB) components have equal intensity values and can be represented by a single intensity value specified for each pixel as opposed to three intensity values needed in a full coloured image. Recent display and image capturing hardware can only support 8-bit image, thus, grayscale images are entirely sufficient for many tasks (Fisher *et al.*, 2003).

The conversion of coloured images to gray level images can be done using either the average method or the weighted method. The average method is considered the easiest of the two as it simply calculates the average of the three colours as given by (Saravanan, 2010):

$$Grayimage = \frac{R + G + B}{3} \quad (2.3)$$

where R, G and B represent Red, Green and Blue channel of the RGB image respectively. However, the problem with this method is that, the output might produce a dark image instead of a grayscale one. This is because, the three different colours have different wavelengths and each of these colours contribute to the formation of the image. When calculating the grayscale value, the average will be taken with respect to each individual colour contribution to the image (Saravanan, 2010).

The weighted method was developed to solve the problem of different wavelengths associated with the average method. Since the Red (R) colour has the highest wavelength of the RGB colours and the Green (G) has the least (with a soothing effect to the eyes), the contribution of the red colour is reduced while that of green is increased. Thus, given a coloured input image, the grayscale image can be obtained by (Saravanan, 2010):

$$Grayscale = (R \times 0.299 + G \times 0.587 + B \times 0.114) \quad (2.4)$$

where R, G and B represent Red, Green and Blue components respectively.

2.2.1.5 Image De-noising

Noises in digital images are unwanted signals that cause distraction in the image. Image distortion is the most significant problem in image processing and can be caused by various types of noise such as Gaussian noise, Poisson noise, salt and pepper noise etc. (Boyat & Joshi, 2015).

Noise is always present during image acquisition, coding process and transmission processes. Noise is very difficult to remove without prior knowledge of the noise model. Image de-noising is a very important process in digital image processing because it cleans

up the image in order to clearly highlight the edges, important details and reduce distraction in the image (Han, *et al.*,2003).

There are two major types of filters used to remove noise in images, namely linear and non-linear filters (Arias-castro & Donoho, 2009).

Linear filters are fast but they do not preserve the details of the image. A typical linear filter is the mean filter. A mean filter is also called an averaging filter because it computes the average value of the corrupt image and replaces the centre pixel intensity value with the average value. It is also a sliding window filter that replaces the centre value in the window with the average mean of all the pixel value in the window, which is usually a square window (Arias-castro & Donoho, 2009). These filters are very effective in eliminating Gaussian noise but not as effective in removing impulse noise and also have the problem of blurring edges (Arias-castro & Donoho, 2009).

Non-linear filters preserve the details of the images. Some examples of these filters are median, minimum and maximum. Each of these filters perform better on a particular type of noise as presented:

1. Median Filters: Median filter is a non-linear filter; it replaces the grayscale value of a pixel by the median of the values in its neighborhood. It is very effective in eliminating impulse noise, salt and pepper noise and still preserve the useful details in the image. However, license plate are mostly corrupted by salt and pepper noise therefore a median filter is the best filter for eliminating noise present in license plate (Arias-castro & Donoho, 2009).
2. Maximum filter*:* Maximum filter is a nonlinear filter that selects the maximum value of all pixel value within a local region of the image. Maximum filters brighten the

image and are useful in eliminating pepper noise and negative outliers noise (Jihui *et al.,* 2015).

3. Minimum filter*:* Minimum filter is a nonlinear filter that replaces the pixel with the lowest pixel value from the local set. Its size is relative to window size, large filter sizes result to a stronger effect and as the window size gets bigger more information loss occurs. A minimum filter darkens the image and is useful in eliminating salt noise (Jihui *et al*., 2015).

2.2.2 Candidate Extraction

After the image has been pre-processed, the license plate region is extracted using edge detection, adaptive thresholding (image binarization) and edge density filter.

2.2.2.1 Edge Detection

In digital image processing, edges are the collection of pixels whose gray value has a step change or the part of an image where the brightness of the local area changes significantly (*Gao et al.,* 2010). The edge of an object is reflected in the discontinuity of the gray level, thus edge s are detected by studying the changes of a single image pixel in a gray area and uses the variation of the neighboring pixel to detect the edge (Gao *et al.,* 2010). The edge detector operators highlight the local edges and then define the pixel edge strength and set the pixel threshold to extract the edge point (Gao *et al.*, 2010).

The fact that license plates are characterized by abundant edge information, many methods based on this have been proposed (Zhao *et al.,* 2011). Edges and boundaries are used to detect the position of the plate in a given image. The edge information can be extracted by applying local operators such as Sobel, Canny, Laplacians, and Prewitt etc. to highlight the vertical and horizontal gradients of the image. Edge based methods are simple and require

less computational effort but require the continuity of the edges. When it is combined with other techniques like morphological operators they are able to find the rectangle that is regarded as the candidate region but consumes more time due to the morphological iterative process. The basic edge detector operators are as follows (Zhao *et al.*, 2011):

1. **Sobel operator**: This is a classical edge detector that is used in detecting the boundaries of an image by calculating the gradient of the image for each pixel position in two- dimension (2D). Sobel operator performs 2D spatial gradient measurement of an image and so emphasizes regions of high spatial frequency that corresponds to edges. Sobel operator uses a pair of 3×3 or 5×5 convolution kernel to estimate the vertical and horizontal gradient of the image pixel.

   The 3×3 Sobel mask for the vertical and horizontal gradient is given as (Aggarwal & Maini 2009):

$$GX = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad (2.6)$$

$$GY = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \quad (2.7)$$

   where Gx and Gy represents the vertical and horizontal gradient, respectively.

   The image kernel is a small matrix used to apply effects such as blurring, outlining etc. on the image. This process is generally refer to as convolution. There are different kernel sizes ranging from 1×3 , 3×3 and 5×5, however using small kernel size always gives a better result than larger kernels and reduces the convolution time. This is because as the kernel size increases more pixel tends to

join part of the convolution process, thus blurring the detected edges to a point. It is noted that the larger the kernel the more time it takes to perform pixel convolution. The magnitude of the gradient is given as (Aggarwal & Maini, 2009):

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{2.8}$$

However, the large convolution kernel of the Sobel operators smoothens the input image to some extent and so makes the operator less sensitive to noise and output fine details. Sobel operator is based on convoluting the image with a small, separate, integer value filter in both vertical and horizontal directions. It is therefore computationally less expensive than its counterpart (Gao *et al.*, 2010).

2. **Canny operator**: This is a multi-stage algorithm that is capable of detecting various kinds of edges in an image (Zhigang *et al*., 2012). The operator finds edges by searching for local maxima in the gradient of the image, calculated using the derivative of a Gaussian filter. The image is smoothed (in order to reduce its sensitivity to noise) using a Gaussian filter with a specific standard deviation thereby reducing the details on the image. The local gradient and edge point (point whose strength is a local maximum) are computed at each point (Biswas & Sil, 2012). The Gaussian kernel used in smoothing the image can result in loss of fine details but this can be controlled by adjusting the width of the kernel. However, the localization error in the detected image increases slightly as the width of the kernel increases (Biswas & Sil, 2012). In view of this, the intensity discontinuities in an image can be found using a canny edge detector. However, it is not guaranteed that these discontinuities correspond to the actual edge giving room for false positive edges (Aggarwal &Maini 2009).

2.2.2.2 Image Binarization

Binary images are images whose pixels have only two possible intensity values. They are normally displayed as 0 for black and 1 or 255 for white (Bradley & Roth, 2007). Binary image is obtained from a grayscale image or coloured image using thresholding operation. Thresholding separates an object in the image from its background. The colour of the object is usually white and is referred to as the foreground colour while the rest is usually black and referred to as the background colour (Bradley & Roth, 2007).

Thresholding is an efficient technique in binarization, which is widely used for image segmentation. The goal is to create a binary representation of the image, classifying each pixel into one of two categories, such as "black" or "white" by selecting an optimal grayscale level threshold value for separating object of interest from its background based on their grayscale level distribution (Bradley & Roth, 2007).

During thresholding, pixels are classified into any of the two categories (Bradley & Roth, 2007):

1. Pixel values of the image that falls below the threshold.
2. The pixel values of the image that are equal to or greater than the threshold

Given an input image f (x, y), the threshold image g (x, y) can be defined as (Bradley & Roth, 2007):

$$g(x,y)=\begin{pmatrix}1if(x,y)\geq T\\0if(x,y)>T\end{pmatrix} \tag{2.9}$$

Thresholding operation T is defined as (Vala & Baxi2013):

$$T=M[x,y,p(x,y),f(x,y)] \tag{2.10}$$

where T is the threshold value, f (x, y) is gray value pixel at point (x, y) and p (x, y) denote some local properties of the point.

Based on this, thresholding can be grouped into two major classes (Vala & Baxi, 2013):

1. **Global thresholding:** When T depends on f (x, y) i.e. on gray-level value only, it is said to be a global thresholding techniques. T represents the global threshold value of the pixels. The illumination of all images are not uniform, some images are brighter and some are darker and as such choosing a global threshold will not yield a good result in all the cases. However, finding a single threshold value that will be compatible to the entire image will be difficult.
2. **Local thresholding:** If threshold T depends on f (x, y) and p (x, y), the thresholding technique is called local thresholding. This method divides an original image into several sub-regions and reasonably chooses the various thresholds T for each sub region**.**

   In local adaptive thresholding, the image is grouped into sub images and an individual threshold value is chosen for each sub image. Since the threshold for each pixel depends on its location within an image this technique is said to be adaptive. Adaptive thresholding generates a binary edge image by thresholding each pixel using a threshold that is adaptively computed from a local window (Bradley & Roth, 2007). The threshold value is calculated for each pixel and if the pixel value is below the threshold, it is set as the background value otherwise it assumes the foreground value. The local threshold is statically determined by examining the intensity value of the local neighbourhood of each pixel. This method cannot completely compensate for the loss of information, but it preserves the information that may be lost when a global threshold is chosen (Bradley & Roth, 2007).

Given an image E_g, the binary image E_b is generated by thresholding each pixel p(x, y) using a threshold that is adaptively computed from a local window and E_i by summing all pixel values from the upper left corner for each pixel in image E_g.

$E_{b(x,y)}$is computed as follows (Yuan *et al*., 2016):

$$E_{b(x,y)} = \begin{pmatrix} 255 if Ei(x,y) \geq \beta\varpi(x,y) \\ 0 else if Ei(x,y) < \beta\varpi(x,y) \end{pmatrix} \quad (2.11)$$

where β is the coefficient used to control the threshold and ϖ (x, y) is the average of all pixel values within the local hw ×hw window surrounding pixel p (x, y).

However, the size of the neighbourhood has to be large enough to cover sufficient foreground and background pixels to avoid choosing a poor threshold value (Yuan *et al.*, 2016). In order to solve the problem of unbalanced illumination, the common solution is using local adaptive thresholding. The main difference here is that a different threshold value is computed for each pixel in the image. This technique provides more robustness to changes in illumination (Bradley & Roth, 2007).

2.2.2.3 Edge Density Filter

The algorithm uses edge density information to detect region of high density, by connecting regions of high edge density and removing sparse regions in each row and column in the binary edge image. The regions that does not exhibit the characteristic of the license are referred to as low density region while those that exhibit the license plate characteristics are the high density region. However, the conventional morphology filter such as the erosion and dilation common in literatures are very time-consuming due to its several iterations, and may not be appropriate for license plate detection that requires real-time processing. The approach is based on the following assumptions of the license plate (Yuan *et al.*, 2016).

1. License plates generally exhibit a high edge density.
2. The characters on a license plate are printed horizontally, and the height of each character is almost identical.
3. A significant distance will exist between each of the plates if an image contains multiple license plates.

It is assumed that each row of pixels in an image consists of an edge pixel and a non-edge pixel. The edge pixels are represented in white while the non-edge pixels are represented in black. If a long black pixel exists on the both ends and a sparely distributed white pixels found at the middle, ideally it is assumed that the non-plate areas exist on both sides of the row and the plate area can likely be found on the middle where there are both white and black pixels sparely distributed (Yuan *et al.*,2016). Therefore, the edge density Ed, is defined as the proportion of edge pixels in the line segment. The target line segment which contains both edge pixels and non-edge pixels, is what is either connected or removed with respect of the white pixel. The decision is based on the edge information which is defined as the proportion of edge pixels in the line segment and is given as (Yuan *et al*., 2016):

$$Ed = \frac{wl}{wl + bl} \tag{2.12}$$

where wl and bl is the white and black pixels respectively in the line segment.

The flow chart of the edge density filter is presented in Figure 2.2.

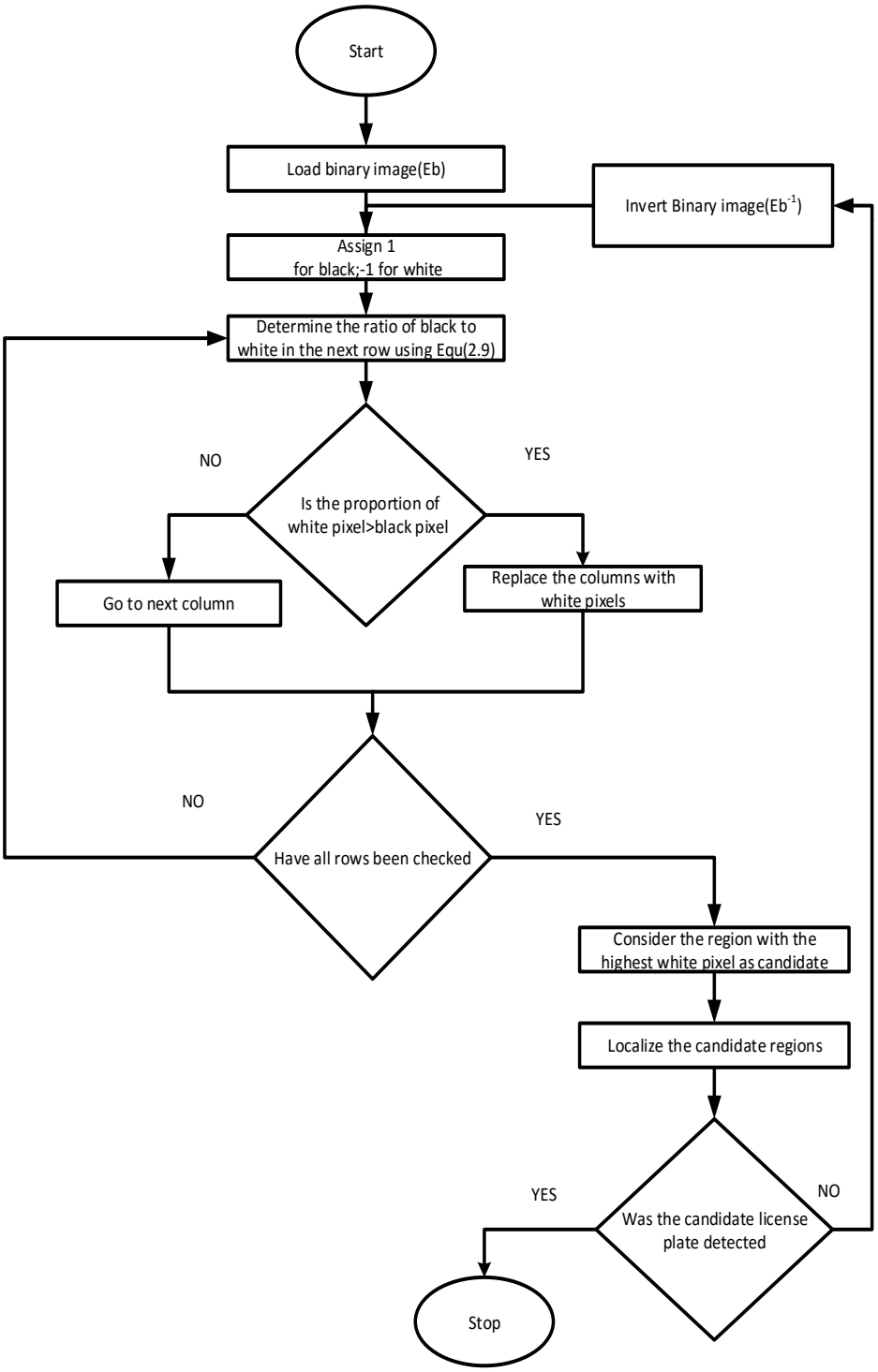


Figure 2.2: Flow Chart of the Edge Density Filter (Yuan *et al*., 2016).

2.2.2.4 Candidate Extraction using Geometric Properties of the License Plate.

License plate region can be detected by using efficient geometric attributes to locate lines that forms the desired rectangles. The geometric features include, rectangle height, width, area, aspect ratio etc. these properties are used to filter out unlikely candidates, if the region does not exhibit these properties it is likely not a license plate. The output of this process contains a few candidate regions where the possible license plates might be located. The predefined values for these features are calculated based on the standard width and high value of a license plate which are mostly given as:

$$24 < w < 256 \tag{2.13}$$

$$6 < h < 32 \tag{2.14}$$

$$256 < w \times h < 15000 \tag{2.15}$$

$$w/h < 6 \tag{2.16}$$

where $w, h, w \times h$, and w/h represents width, height area and aspect ratio respectively.

2.2.3 Datasets

There are publicly made available dataset which are used in computer vision community to evaluate the performance of various algorithms. An example of these dataset is the Caltech dataset. This database consists of thousands of various categories of images used for training and validation processes. They are made publicly available for research processes. The database consists of dataset of various objects e.g. the category of car 2001 (rear) consist of 526 images of rear view of vehicle while that of car 1999 (rear)2 contains 126 images of the vehicle. These images are taken in an ideal condition and are captured with a standard camera so cannot accurately be used to evaluate the performance of newly developed

algorithms. Therefore, researchers tend to develop their own dataset which are more complex than the formal. However, Yuan *et al.,* (2016). collated the PKU VehicleID dataset which comprises of several hundreds of vehicle images to validate the robustness of their algorithm.

2.2.4 Deep Learning Algorithm

Deep learning is a broad family of machine learning based on feature learning or representation learning as opposed to task specific algorithm. This implies that the system automatically discovers the representations needed for feature detection and classification for raw data. Thus replaces manual feature engineering and allow the machine to both learn the features and use them to perform specific tasks (Singh *et al.,* 2017).

2.2.4.1 Deep Convolutional Neural Network (dCNN)

Deep Convolutional neural network (dCNN) is a special (deep hidden-layer) neural network used as a machine learning paradigm whose goal is to learn complex functions. They utilize a supervised learning technique called back propagation for training. CNN requires relatively little pre-processing compared to other classification algorithms, which means that the network learns the filter that in traditional algorithms is hand engineered (Goodfellow *et al.,* 2016).

However, CNN usually requires large amounts of training data in order to avoid over-fitting. A common technique applied when large dataset is not available is to train the network on a large data set from a related domain and this process is known as transfer learning. Once the network parameters converge, an additional training setup is performed using the pre-trained dataset to fine-tune the network weights. This allows the CNN to be successfully applied to problems with small training datasets (Goodfellow *et al*., 2014).

2.2.4.2 Convolutional Neural Network Architecture (ConvNet)

ConvNet is a deep learning model that consist of several convolutional layers, in each layer a nonlinear activation function is applied on its input and provides a representation in its output. The model is not only used for feature extraction purpose as some widely used feature extractions such as GIST, HOG, SIFT and LBP etc. but combines the extraction and classification process into one integrated system. The model automatically learns complex representation of the unsupervised image data and then classifies them based on supervised data. It extracts the feature of the object feed into it, by convolving the image pixel with the convolutional filters (kernels) which is usually a 3*3 filter mask. The kernel slides through the entire image window. These features are represented as a feature map and are pass to the pooling layer where they are down sampled and further sent to other convolutional layer for further feature learning. Figure 2.3 shows a typical CNN architecture.

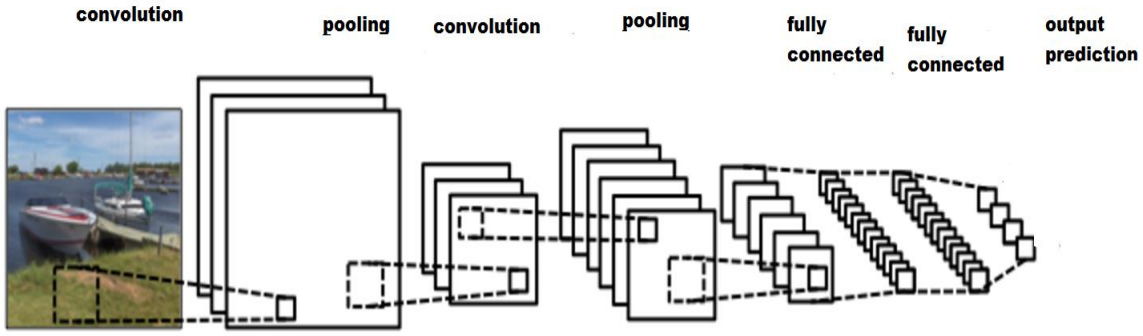


Figure 2.3: General Overview of CNN Model (Goodfellow *et al*., 2014).

There are several pre-trained convNet model such as the ResNET, Inception, GoogleNET, AlexNET and VGGNET etc. The ALEXNET model will be explored due to its high accuracy and rich features.

2.2.4.3 Alexnet Deep Convolutional Neural Network

ALEXNET is a deep convolutional neural network (dCNN) developed by supervision group, consisting of Alex Krizhevsky, Geoffrey Hinton and IIya Sutskever (source). The model was name after the first author Alex Krizhevsky. It is made of 5 convolutional layers and 3 fully connected layers. Each convolutional layers contains 96 kernels of filters of the same size $(11\times11\times3)$ which represent the width height and the depth of the image (number of channels). The model is inspired by the 2012 ImageNet competition where it achieved a top-5 error (rate of not finding the true label of a given image among its top five predictions) reducing the error rate from 26.1% to 15.3% more than 10.8% points ahead of the second runner up during 2012. ImageNet large scale visual recognition challenge (ILSVRC). In Practise only very few people train convolutional network from scratch because it is rare to have a dataset of sufficient size. Instead it is common to adopt a pre-trained convNet model and then use the convNet as feature extractor for the specific task. This is made possible by transfer learning which is simply the application of skills, knowledge that were learnt in one situation to another learning situation (Tajbakhsh *et al.*, 2016).

The Alexnet dCNN consist of the following layers as showed in Figure 2.4.

Figure 2.4: The ALEXNET Architecture (Tajbakhsh *et al.*, 2016)

1 'data'	Image Input	227x227x3 images with 'zerocenter' normalization
2 'conv1'	Convolution	96 11x11x3 convolutions with stride [4 4] and padding [0 0 0 0]
3 'relu1'	ReLU	ReLU
4 'norm1'	Cross Channel Normalization	cross channel normalization with 5 channels per element
5 'pool1'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0 0 0]
6 'conv2'	Convolution	256 5x5x48 convolutions with stride [1 1] and padding [2 2 2 2]
7 'relu2'	ReLU	ReLU
8 'norm2'	Cross Channel Normalization	cross channel normalization with 5 channels per element
9 'pool2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0 0 0]
10 'conv3'	Convolution	384 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]
11 'relu3'	ReLU	ReLU
12 'conv4'	Convolution	384 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1]
13 'relu4'	ReLU	ReLU
14 'conv5'	Convolution	256 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1]
15 'relu5'	ReLU	ReLU
16 'pool5'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0 0 0]
17 'fc6'	Fully Connected	4096 fully connected layer
18 'relu6'	ReLU	ReLU
19 'drop6'	Dropout	50% dropout
20 'fc7'	Fully Connected	4096 fully connected layer
21 'relu7'	ReLU	ReLU
22 'drop7'	Dropout	50% dropout
23 'fc8'	Fully Connected	1000 fully connected layer
24 'prob'	Softmax	Softmax
25 'output'	Classification Output	Cross entropyex with 'tench' and 999 other classes

The ALEXNET structure is presented in Figure 2.4: The first layer represents the data layer which contains the input image. The size of the input image 227 x 227. The convolutional kernels or filter extract interesting features in the image. Then the RELU which is the rectified linear units is an activation function that triggers the neurons of the network. RELU helps the network to converge faster. A max pool layers which reduces the dimension and sizes of image without altering the feature representations of the object. The fully connected layer fc8 is a fully connected multiple layer neural network which comprises of input neuron, hidden layers and output layer. Finally, the softmax layer which is as a classifier used to classify the predicted label at the output layer.

2.2.5 Character Segmentation

If an image of the vehicle is given as an input to the system, the first step will be to detect the location of the license plate and determine its position, then extract the plate. After the extraction of the plate the resulted image is segmented so that each image segment contains one character or one-digit number. This operation decomposes words in an image into individual characters. It is the most important stage in the recognition process because the accuracy of the recognition depends on the segmentation (Kaur *et al.*, 2015). However, in order to recognize the characters, the plate is segmented into patches, where each patch contains only one character. This is reported to be a challenging task as a result of plate frame, plate screws and fused characters etc. (Li & Shen, 2016). Several approaches have been proposed for segmentation of characters such as; the classical approach where segments are identified based on character like features, implicit segmentation (recognition based segmentation) where the system searches the image for components that matches the character classes in the alphabets (Bansal & Sinha, 2012). The advantages of this method is that they bypass the segmentation problem by operating directly on the image pixel. Holistic method tends to recognize the entire word as a unit, the disadvantage of this method is that, they only restricted it to a predefined word, this is because they deal directly with words and not characters (Casey & Lecolinet, 2013). This method is usually used for on-line recognition. However, this work's approach to segment the character on the license plate will be a hybrid of the classical and the recognition based segmentation.

2.2.6 Character Recognition

The final stage in the license plate recognition system is the character recognition. This process entails recognizing each segmented individual alphabets and numbers of the license

plate by assigning a label to the segmented character and number and classifying them to their best matching characters (Li & Shen, 2016). Many researchers has proposed a few methods for the recognition of characters like template matching, feature extraction, fuzzy logic, neural network, support vector machine, hidden Markov and Bayes net (Sharma, *et al.*, 2017).

Generally, character recognition is performed by using the learning based techniques to train the extracted features of the characters (numbers and alphabets), which are then used to recognize characters in test samples. On the other hand, template matching uses character stored as template in the database to classify images to correct labels. Fuzzy logic method is based on clustering where a membership function is built using the available data by using a clustering techniques which is used to partition the data and produce a membership function from the resultant clusters hence characters with similar features are contain in one clusters. Hence the cluster is identified first and then the actual character.

2.2.7 Performance Evaluation

The performance of the developed license plate recognition scheme was evaluated using the following matrices:

1. **Detection rate:** This is the total number of license plates that are completely extracted without including any extra pixels of the number plate.
2. **True positive (TP**): This denotes the total number of license plates that are correctly recognized.
3. **False positive (FP):** This denotes the total number of license plates that are incorrectly recognized as a license plate.

4. **False negative (FN):** This represent the total number of license plates that was actually a license plate but recognized as a non-license plate.
5. **Precision:** Precision is defined as the number of license plates that are detected successfully divided by the total number of detected regions. It provides information on the amount of false alarms. Mathematically it is defined as (Yuan et al., 2016):

$$\frac{TP}{TP + FP}$$

6. **Recall:** Recall is defined as the total number of license plates detected successfully divided by the total number of ground truth image. It measures how many ground truth objects have been detected. Mathematically it is defined as:

$$\frac{TP}{TP + FN}$$

7. **Recognition accuracy**: This is defined as the fraction of the labels that the model predicted correctly. Mathematically it is defined as:

$$\frac{TP + TN}{TP + TN + FN + FP}$$

2.3 Review of Similar Work

The following are the review of similar works that have been carried out in the area of license plate detection. This review is carried out to understand the extent of research in this area, tools and approaches used. The knowledge gained facilitate this research work by using different tool and approach to get better result.

Zhou *et al.*, (2012) proposed a license plate detection algorithm that employed the Principal Visual Word (PVW) and local feature matching which was motivated by the Bag of Word (BoW) model. The approach consisted of three key components: PVW generation, visual word matching and license plate localization. In PVW generation, sufficient feature samples from labeled training images were collected and the PVW for each character in license plate was generated via clustering. During visual word matching the extracted shift invariance feature transform (SIFT). features of the test image were compared with each character in PVW. and the license plate was located based on the matched results. A feature is considered as a candidate match if the minimum descriptor distance to a certain PVW of a certain character was less than a certain threshold. However, their approach was not suitable for applications that require real time processing as a result of the high computation cost inquired during the feature extractions processes.

Al-ghaili *et al.*, (2013) proposed a method for license plate detection using low-resolution images based on vertical edge detection to reduce the computational complexity of the detection. The coloured image was enhanced and converted to grayscale, then adaptive thresholding is applied to binarize the image, and the vertical edge detection algorithm was used to extract the vertical edges. The plate region was detected by applying Unwanted Line Elimination Algorithm (ULEA). Finally, the statistical analysis of the license plate (proportion of the black and white pixel in the width and height of the license plate) was used to identify the true license plate region. However, this feature used for the verification process is not robust enough to detect the license plate correctly. The (ULEA) exhibited certain morphological characteristic and not suitable for real time applications like the license plate detection.

Lalimi *et al.*, (2013) presented a license plate detection system that used a region based filter to extract the license plate region. Histogram equalization was used to improve the contrast locations containing a license plate and a region-based filter was used to smoothen the background areas of the image. The Sobel operator was used to detect the edges of the image and morphological operator bused to extract the candidate regions. Finally, the plate region was segmented using some geometrical features of the license plate such as height, width area and aspect ratio. However, this approach was time consuming due the morphological technique used and also the experimental results showed that the proposed method is only very effective on predefined license plates.

Dewan *et al.*, (2015) presented a license plate detection system using ant colony optimization (ACO) to improve segmentation of the characters for number plate recognition. The algorithm is inspired by the natural behaviour of ant species that deposit pheromone on the ground for foraging and the ACO was used to detect the edges of the image by establishing a pheromone matrix that represents the edge information presented at each pixel position of the image. The edge information is inspired by the movement of the ant that was driven by the local variation of the image intensity thereby distinguishing the license plate region. The edge detected image was dilated to fill in the holes after which morphological operation of closing and erosion was applied. Finally, the license plate bounding box was detected using connected component analysis (CCA). The ant colony optimization (ACO)approach apparently reduces the computation time that might arise using a conventional edge detection. However, the erosion and dilation operation tends to consume more time.

Nguwi & Lim (2015) proposed a method for number plate recognition in noisy images. The input image was converted to grayscale and a median filter is applied to remove noise from the input image. Morphological operation of opening and closing was used to detect the number plate region by eliminating the components with improper aspect ratios. The character recognition aspect is carried out using back propagation neural network. The results showed that the training process converged easily. However, the algorithm did not consider images of license plates from more challenging backgrounds such as occluded images, faded plate numbers and images taken under hazardous conditions.

Agarwal & Goswami (2016) presented a method for plate detection using boundary information. The proposed method used a Canny operator to detect the edge, the boundaries of the license plates where represented using the colour transition between the license plate and the background of the plate. The edge information is based on the intensity variation between the background and the foreground. Segmentation was carried out using morphological operation to separate the characters and remove other unwanted areas on the license plate. Finally, the individual characters on the plate were recognized using template matching. However, the morphological operation used was time consuming and may not be suitable for real time applications and the canny edge detector resulted in loss of details in the image.

Attah *et al.*, (2016) presented an automatic vehicle license plate detection and classification system using the colour information of the license plate. Vehicles images were classified into government, commercial or private vehicles based on the colour of the license plates.

Image acquisition was done using a low cost digital camera, then, various pre-processing operation such as grayscale conversion, noise filtering and edge detection were carried out on the acquired images. Image segmentation was done using watershed (morphological) segmentation technique, while template matching was applied to classify the matched digit. However, the detection process only considered the region where the license plate is located rather than the complete vehicle image as the input to the system.

Li & Shen (2016) presented a method of license plate detection that explores the convolutional neural network using character features. They developed a convolutional neural network using stochastic gradient decent with back propagation. They trained a 36 class CNN classifier using character features. 4-layer CNN classifier was used to classify whether the image patch contains characters or not. The detected license plate bounding boxes was filtered based on some geometric of the bounding box. The license plate region is refined based on the edge information of the license plates; vertical edge detection was done on the cropped license plate image using Sobel operator. A convolution neural network binary classifier was used to classify the image into plate and non-plate region. The framework was able to effectively detect license plate with both high recall and precision. However, an improved result can be achieved when the convolution neural network was trained on a large dataset containing images of similar domain to improve the classification rate. A further increase in the convolutional neural network layers will yield an improved result.

Yuan *et al.*, (2016) proposed a license plate detection system using feature descriptor based on colour saliency features of the license plate to detect the region of the license plate on the

vehicle image. In their work the input colour image is resized and converted to grayscale image. Sobel operator was used to detect the edges of the image and adaptive thresholding is applied to binarize the edge image. Finally, a line density filter which uses the edge information of the license plate was applied to extract the candidates' region. In order to distinguish the candidate region from other regions on the image geometric attributes of the plate was used. To classify the final license plate region from the other detected regions a Cascaded License Plate Classifier (CLPC) based on Support Vector Machine (SVM) was used to classify the true license plate from the other candidates. However, their implementation remains shallow due to the use of a single feature extractor which is a hand engineered, time consuming and cannot completely represent the complex image structures which was necessary for effective and accurate classification of the labels.

Fomani *&* Shahbahrami (2017) proposed a license plate detection algorithm using morphological opening and closing techniques. The method consisted of the following steps, pre-processing, adaptive morphological closing (AMC), local adaptive thresholding (LAT) and adaptive morphological opening (AMO). Histogram equalization was used to enhance the image contrast and AMO operation such as dilation and erosion was applied to disconnect region of weak junctions. The Local Adaptive Thresholding(LAT) was used to segment the image. To disconnect the extra pixels that are connected to the license plate, Adaptive morphological operation of closing was applied to fill in the holes and flatten the license plate region using a group of structural element. The algorithm was developed using OpenCV library. The results when tested on some real dataset shows an improved result in detection accuracy as a result of using a group of structural element as opposed to a fixed one which was computational expensive. However, the proposed approach was not suitable

for real time applications because of the time-consuming nature of the morphological operations.

Selmi *et al.*, (2017) proposed a license plate detection system using convolutional neural network. In their work the coloured image was converted to HSV colour space, image de-noising was carried out using gaussian blur filter, adaptive thresholding and contour algorithm was applied to obtain the region of interest, geometric filtering was applied to obtain the bounding box. Finally, deep learning architecture (CNN) model was used to distinguish between LPs and non LPs. The CNN model used was implemented with tensor flow framework and the result of their experiment when tested only on caltech dataset and obtained a precision of 83.80% and recall of 91.30% and accuracy of 92.01%. However, exploring transfer learning techniques rather than training the model from scratch will increase the rate of recognition due to the limited available Caltech dataset.

Hossen *et al.*, (2018) presented a license plate detection system that was capable of detecting a Bangla number plate using morphological operation, their algorithm was based on the colour information of the license plate. The input image was converted to binary image and then closing operation was used to fill the connected using, contour algorithm and aspect ratio was used to locate the license plate region while opening operation was used to dilate the image based on the structural element. The performance of their work was evaluated on 180 images of Bangla license plate and achieved a detection rate of 93.89%. However, due to the series of iterative processes involved in the dilating the image more computational time was needed for the operation and thus not suitable for real time applications.

From the literature reviewed, it is evident that most of the algorithms used for license plate detection systems usually comprise of morphological operations, and these are known to be time-consuming due to their iteration processes. This problem limits their application for real-time systems. Therefore, combining the edge density filter with a deep learning (dCNN-based) scheme for recognizing the license plate region would improve the detection and recognition of the license plate in this developed work.

CHAPTER THREE
MATERIALS AND METHODS

3.1 Introduction

In this chapter, the materials and methods employed for the successful completion of this research are discussed. The methodology adopted for the development of deep learning based license detection is also presented.

3.2 Materials

The materials that were used for this research work include the following:

1. A computer system with 4G RAM and 2.2GHZ processor was used as the processing system
2. Nikon D40 camera for capturing images of vehicle.
3. MATLAB/Simulink R2018a

3.3 Methodology

The methodology adopted in carrying out the research work are itemized as follows:

1. Development of the ABU vehicle image dataset using the following steps:
    a) Image acquisition
    b) Image pre-processing
    c) Create and save the image in the database

2. Collation of publicly available license plate datasets
    a) Collation of PKU VehicleID dataset
    b) Collation of the Caltech dataset
    c) Create and save the images in a database

3. Development of the dCNN-based license plate detection system using the following steps:
    a) Load the pre-processed input image
    b) Detect the edges of the image
    c) Candidate's region identification
    d) Candidate region extraction using CCA
    e) Determine the final candidate region using geometric attributes of the plate.
    f) Load the pre-trained ALEXNET model
    g) Perform feature extraction and fine-tuning
    h) Modify the classification and output layer.
    i) Perform transfer learning and retrain the ALEXNET model
    j) Classify the predicted labels
4. Testing of system developed in 3 with datasets of 1 and 2 using the following steps:
    a) Load the pre-processed image
    b) Perform the operation in methodology 3b-3j
    c) Display the confusion matrix

5. Comparison of results obtained in 4 with results obtained from the scheme developed by Yuan *et al* (2016) using the performance metrics of detection rate, precision,recall and recognition accuracy**.**

3.4 Development of ABU Vehicle Image Dataset

In this research work, images of vehicle were captured using a digital camera in Ahmadu Bello University, Zaria. These image are captured at a resolution of 300 x 1508 pixels with a Nikon D40 camera specification, under different environment and light conditions. The dataset can be accessed through this link:

https://drive.google.com/drive/folders/1D4KuB9MpTxM7ZqPKsoJf5mZWXBM0kdsu.

3.4.1 Image Acquisition

The acquired images obtained from Ahmadu Bello University Zaria as listed in Table 3.1: are grouped into 5 categories namely B1, B2, B3, B4 and B5 based on the following conditions: day, night, images with single and multiple license plate, and images affected by reflection. A sample of the dataset is shown in Figure 3.1. The number of images obtained is based on available vehicle images.

Table 3.1: Description of the ABU Vehicle Datasets

Dataset	Image group	Scenes Description	No of images
ABU	BI	Images taken at night	23
ABU	B2	Images taken in a sunny weather	50
ABU	B3	Reflected images	8
ABU	B4	Multiple vehicle images	20
ABU	B5	Image captured under the rain	25



Figure 3.1: Sample of the ABU Dataset

Figure 3.1 shows some samples of the acquired images. The images consist of single, multiple vehicle images, images with distractions such human beings, windows, tress and light posts road lanes, and buildings. They are either the front or rear view of the vehicle images captured at a stead still or in motion.

3.4.2 Pre-processing the Acquired Image

The license plate detection scheme starts with the image acquisition, after which the acquired images are pre-processed. The developed datasets contain images captured at a very high resolution (3000 x 1508) which increases the computational complexity of the

scheme, some pre-processing techniques are then applied to help reduce the computational cost associated in using the original images and increases the performance of the system. In this work, the pre-processing techniques applied are discussed in the following sub-section:

3.4.2.1 Down Scaling of the Image

The original coloured image was downscaled to reduce the computational complexity. The scaling factor used for the operation is determined by the width and height of the characters on the license plate region as given in equations (2.1) and (2.2):

The width and height of the license plate region were measured to be 60cm by 20cm to match the actual size of a standard number plate. The scale used was 3 and 2 respectively for the height and width of the plate. This is because the width is always greater than the height. Figure 3.5: shows the snippet code of the downscaling operation.

```
[hi, wi, ~] = size(I);
dw = 3;
dh = 2;
ws = wi/dw;
hs = hi/dh;
Ir = imresize(I,[hs, ws],'bilinear');
figure; imshow(Ir)
title('resized image')
```

Figure 3.5: Snippet Code of the Downscale Operation

3.4.2.2 Converting RGB to Grayscale Images

The input image is converted to grayscale for easy processing. In this research work the conversion technique used is an average weighted method as described by equation (2.4) to avoid outputting dark images.

3.4.2.3 Noise Elimination

In this research work, the noise in the input image was remove using a 3×3 median filter, this is because median filter preserves the edges of the image and produced edged images

with fine details. However, license plates are known to be corrupted by salt and paper noise hence using a median filter completely eliminates the noise without blurring the rest of the objects in the image. The program code for the operation is depicted in Figure 3.6

```
[fname path]=uigetfile('*.*','Enter an image');
fname=strcat (path,fname);
% subplot(2,2,1)
a=imread(fname);
imshow(f)
g=rgb2gray(f);
c=medfilt2(g,[3 3]);
subplot(2,2,1);imshow(g);title('gray image')
subplot(2,2,2);imshow(c);title('filtered image')
```

Figure 3.6: Snippet Code of Median Filter

3.4.3 Collation of Publicly Available License Plate Dataset

The publicly available datasets were obtained as follows.

1. Vehicle images developed by California Institute of Technology obtained from (Caltech, 2005)
2. Vehicle image from Peking University obtain from. www.wzou.eu.

3.4.3.1 Caltech Vehicle Dataset

The Caltech dataset (developed by Computational Vision group at Caltech (California Institute of Technology)) consists of 126 rear view of vehicles images captured from outdoor scenes, each image contains only a single vehicle. The sizes of the images are 750 x 520pixels with 256 (8-bit) grey levels per pixel. Figure 3.2: shows some images from Caltech database.

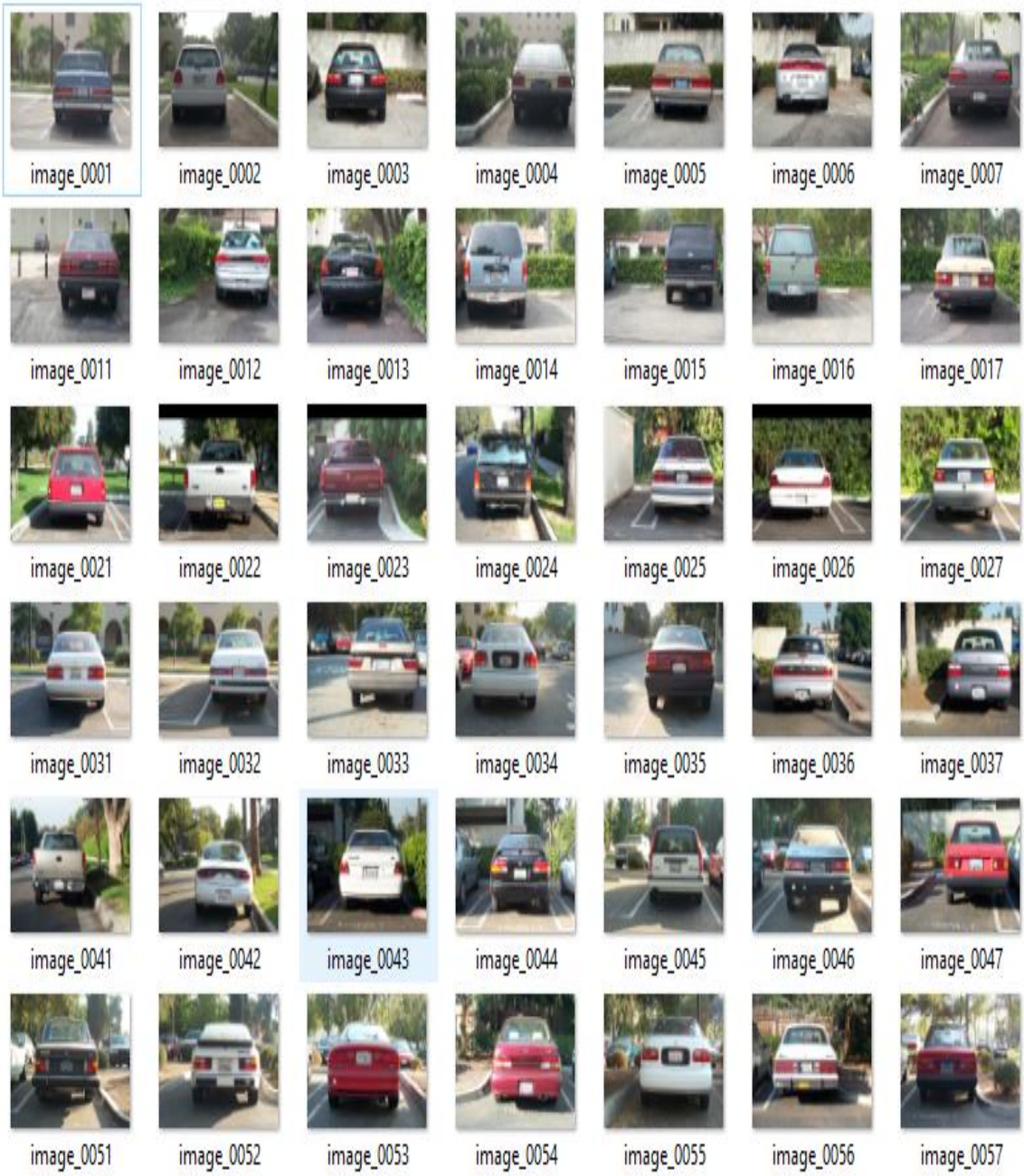


Figure 3.2: Samples of the Caltech Dataset

3.4.3.2 Pekins University Dataset

Pekins University VehicleID (PKU VehicleID) dataset was developed by the National Engineering Laboratory for Video Technology (NELVT)). The dataset consists of five groups of vehicle images labelled as G1-G5. These images captured variations of vehicles

from different weather conditions (day, night, sunny single and multiple plate numbers.) The size of the images are 1750×508 pixels with 256 (8-bit) grey levels per pixel. Figure 3.3 shows some images of PKU VehicleID database obtained from G1 category containing singles images captured in the day time.



Figure 3.3: Sample of the PKU VehicleID Dataset

3.5 Development of the Deep Learning Based License Plate Detection Scheme.

The processes involved in the development of the license plate detection scheme are discussed in details in the following sub-sections. The license plate detection scheme was developed using current image processing and computer vision techniques, which are Sobel

edge detector, edge density filtering, connected component analysis and the geometric characteristics of the license plate and lastly a pre- trained deep convolutional neural network (ALEXNET)) to extract features and classify the extracted license plate candidates. The flow chart of the processes carried out is described in Figure 3.4 and a detail description of the steps involves for development of the scheme are also discussed.

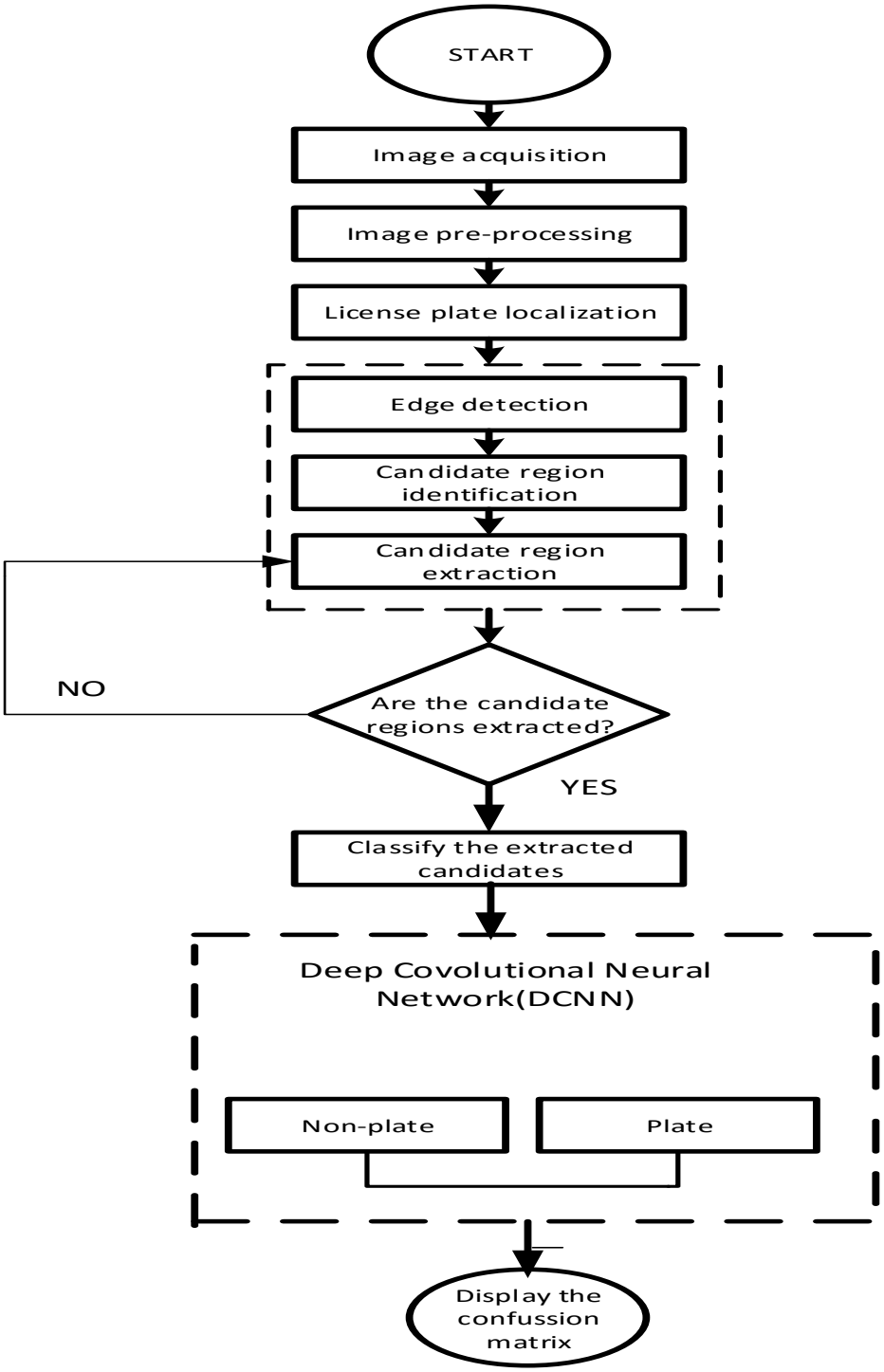


Figure 3.4: Flow Chart of the Developed Scheme

3.5.1 Edge Detection

Edges represent the discontinuity of an object in an image. It is used to describe the boundary of the object. Sobel operator was applied to obtain regions in the image which shows discontinuity in intensity. This operation was applied in both vertical and horizontal direction. The kernel size used was a 3×3 window and as depicted in equation 2.6 & 2.7 respectively. The choice of using a 3×3 kernel is to prevent blurring the detected edges which in turns output a better result with fine details. The program code for the sobel operator is depicted in Figure 3.7.

```
Ie = edge(Ig, 'Sobel', [ ], 'vertical');
figure; imshow(Ie)
title('sobel edge detection image')
```

Figure 3.7: Snippet Code of a Sobel Edge Detector

3.5.2 Candidate's Region Identification

When Locating the candidate region on the edged image, the following processes was carried out.

1. Determination of the exact location and position of the license plate.
2. Appending a bounding box rectangle over the license plate.

Considering the edged information, the license plate is believed to have a high contrast on its number plate region, the alpha-numeric characters and background in an image have a sharp variation in intensity, these features were used in detecting the license number plate region. The binary image obtained during the binarization process consist of both black and white pixels. However, license plate is mostly found at the highest edged dense region i.e. the region with the highest probability of white pixel. These regions are regarded as candidate regions. This was achieved using the edged density (ED) as defined in equation

(2.12). Considering the equation, if the proportion of the white pixel is more than the black pixel, there is a probability that the region contains a license plate.

In order to achieve this, a line map for each pixel in the binary image is generated and the pixel on the line segments are analysed by representing, a black pixel as 1 and a white pixel as -1, if the row contains a continuous black pixel, they are grouped together. For continuous white pixel the cumulative length of each white line segment represented by a negative number is calculated. Considering the license which contains mostly white background on the license plate regions, such as the Caltech and the ABU dataset, the binary image is inverted, that is all white pixels are replaced with black pixels and vice versa. The program code for the edged density filter is presented in appendix C:

The following threshold values were set: **Tmin** which is the minimum length threshold for a black line, **Tgap** is the gap-length threshold and the edge density threshold **Tdl.** The default values for **Tmin**, **Tgap** and **Tdl** are as 10, 6 and 0.03, respectively. The flowchart of the algorithm is depicted in Figure 2.2.

3.5.3 Verify the Candidate Region using Connected Component Analysis (CCA).

To verify the candidate region, Connected Component Analysis was applied to identify regions that do not exhibit the geometric property of a license plate. These properties include the width, height, area and aspect ratio of the plate region. Since the license plate candidate areas are discriminated by the area and its sizes, regions that do not exhibit these properties are not considered as the candidate region. Based on these assumptions the following process was carried out:

1. The bounding box coordinate was specified as 60cm x 20cm which is the size of the extracted candidate's region.
2. The candidate region was also specified by considering various sizes of the number plates using equation (2.13) - (2.16). To ensure that the region identified satisfies the condition of a typical license plate. Figure3.8 depict the snippet code for the operation.

```
candidate = regionprops (If, 'BoundingBox');
 ind = [];
for i= 1:numel(candidate)
   w = candidate(i). BoundingBox(3);
   h = candidate(i). BoundingBox (4);
   C1 = (w>24) && (w<256);
   C2 = (h>6) && (h<52);%(h<32);
   C3 = (w*h>256) && (w*h<15000); %(w*h<4096);
   C4 = w/h < 6;
%     [C1 C2 C3 C4]
   if ~ (C1 && C2 && C3 && C4)
      ind = [ind, i];
candidate(ind) = [];
```

Figure 3.8: Snippet code of the Connected Component Analysis (CCA)

3.5.4 Determination of the Final License Plate Candidate Region

If the license plate area is found using the size, area and the aspect ratio only, it will not be detected accurately, because objects with similar sizes and aspect ratio are detected along. It is possible that other regions such as (headlamps, stickers, sign post etc.) that looks exactly like a license plate are also being marked as one. To eliminate those other false regions, a classification techniques was use to verify the true license plate region. ALEXNET pre-trained CNN was used feature extraction and classification of the true license plate from other detected regions.

3.5.5 Feature Extraction and Fine-Tuning

ALEXNET pre-trained model has been trained on over a million images IMAGENET and has learnt rich features from the wide range of images. In this work, instead of training the entire network from scratch, features from the already trained network are exploited to extract an initial set of feature representation which was used in the classification of the new dataset. The derived representation was transferred into a supervised neural network classifier.

3.5.6 Modification of the Classification and Output Layer

The last three layers of the pre-trained network are configured for 1000 classes, these layers depicted as layer 23-25 in Figure2.4 consist of the FC8, the softmax layer and the classification output layers respectively. The earlier layers of the ALEXNET are fine-tuned to coincide with the new dataset instead of training the entire network. During the fine-tuning process the fully connected layer depicted as (fc8) as shown in Figure 2.4 is replaced with a new layer (A single layer neural network) having the same number of output as the number of classes of the new dataset. While the softmax layers which outputs the probability of the predicted label is removed since the expected output is not a probability distribution but class labels. The final classification output layer is responsible for classifying the new dataset and then predicts the resultant output of the new dataset.

3.5.7 Transfer Learning and Retraining of the Model

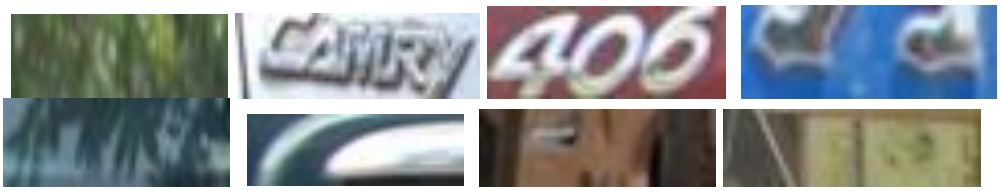
The following steps were carried out during transfer training process

1. Specify the dataset for the training and testing images: The training and testing images are the extracted candidate regions of both negative and positive samples as depicted in Figure 3.9. The size of the training data required by the ALEXNET

model is 227 by 227 RGB image. Therefore, since the input is grayscale, it is converted to RGB image by replicating the single channel to obtain a 3-channel RGB image.



a) Positive Training Samples of the Developed ABU Dataset.



b) Negative Training Samples of the Developed ABU Dataset.

Figure 3.9: Positive and Negative Training Samples

2 Perform data augmentation: Data augmentation was carried out on the dataset in order to increase the number of dataset thereby generating additional data from the existing data. During the data augmentation process the training images were randomly flipped about the vertical direction and rotates with an angle of 90 degrees to obtained an image different from the original image, the earlier flipping operations creates a mirror image which is also a valid image and double the sizes of the original dataset. The code for the data augmentation carried out on the image is depicted in the program code in the appendix.

This operation is necessary to avoid overfitting as a result of the limited number of available dataset. The operation generated additional 450 images of each dataset (PKU vehicle ID, caltech and ABU dataset) resulting to a total of 600 images of each of the datasets.

3 Split the augmented dataset into training and testing sets: The training and testing sets were splitted into ratio of 70:30 which is depicted by the line of code: [trainingSet, testSet] = splitEachLabel (candidatesSet,0.7,'randomized').

4 Load the pre-trained ALEXNET convolutional neural network and checks the layers and classes. The network weights are available on caffe toolbox through the link: http: 16-layer model: information page in the caffe zoo, weights (500MB), layer configuration.

5 View the network architecture using the Layers property on the matlab command window, (layers = net. Layers). And identify the layers to freeze via the Alexnet architecture as presented in Figure 2.4.

6 Set training option: During the training process, the training algorithm options were set as follows:

   a Initial Learning rate for transfer learning was set as 0.001, a smaller learning rate used slows down the learning process and prevents the algorithm from memorizing the datasets.

   b Stochastic gradient descent (SGD) was used to train the network. The SGD provides an iterative technique to achieved the optimal result.

   c Specify the number of mini batch size as 52, maximum training epoch 6 and number of iterations 30.

7 Run the model consisting of the transferred layers only on the training and test data

During the process the weights of the earlier convolutional layer [2-22] which is the transferred layer are frozen by setting its learning rates to zero, however, the gradient of the frozen layers is not updated during the training, therefore learning only the high level features of this layer and fine-tuning layer 23-25 to coincide with the new dataset. The program code depict the layer transfer process.

```
layers Transfer = net. Layers (1: end-3);
numClasses = numel (categories (augimdsTrain. Labels));
    layers = [ layers Transfer
 fullyConnectedLayer(numClasses,2)
     softmaxLayer[none];
     classificationLayer];
```

8 Load the output weight obtain in 7 and train a single layer neural network on the augmented dataset using the function code:

```
[LPDnet, info] = trainNetwork (augimdsTrain, layers, options);
```

augimds stores the augmented input image data, layers define the network architecture and options defines the training options.

3.5.8 Classification of the Output

In this work the main goal of the classifier is to compare the representation of license plate image with those of the training set templates, and determine the categories to which each extracted candidate image belongs, the fully connected layer on the ALEXNET model classifies the output of the 1000 object classes of the ImageNet data repository, however, seems there are only two labels to be classified in this work, a the single layer neural network is a binary classifier used to classify the predicted class of either a license plate and non–license plate.

3.6 Testing of the Developed Scheme on the ABU, Caltech, and PKU vehicle ID

During testing of the developed scheme the following processes were carried out.

1 Load the Pre-Processed Image

The image which have been pre- treated using image processing tools and resized to 227 by 227 which is the desired input size of the ALEXNET model serves as the input to the developed scheme

2 Perform the Operation in Methodology 3b -3j

The basic operations which is detailed in methodology 3 is applied to the pre-processed image, these operations are said to be the main functionality of the developed system.

3 Display Eight Samples of Test Image with Their Predicted Labels

The 8 randomly selected test images with their predicted label are displayed as depicted in Figure3.10.



Figure 3.10: Test images with Their Predicted Labels

4 Compute the Confusion Matrix

Confusion matrix is a table that shows the true and false positive and negative predictions of the actual and predicted labels. Figure 3.10 depict a confusion matrix plot that used to evaluate the performance of the classifier on the sets of the test data.

		Predicted class	
		P	N
Actual class	P	True Positive(TP)	False Negative(FN)
	N	False Positive(FP)	True Negative(TN)

Figure 3.11: Confusion Matrix Plot

The confusion matrix plot in Figure 3.11: shows the actual class and the predicted class where P represent the positive prediction and N represent the negative prediction of the classes respectively. The plot is made of true positive, true negative, false positive and false negative prediction which are used to determine the precision, recall and the classification accuracy of the developed scheme using the equations 2.17, 2.18 and 2.19 respectively.

3.7 Comparing the Performance of the Developed Scheme with the existing Scheme

To evaluate the performance of the developed system certain matrices are used. These matrixes include, Detection rate, precision, recall and classification accuracy they are explicitly explained in 2.2.7.

The detection performance is based on the fact that the license plate must be completely encompassed within the bounding box before it is considered successful otherwise it is not

successfully detected. Randomly selected 50 images of each dataset was used to evaluate the performance and compared with the existing scheme

Comparing the performance of the scheme using precision recall and accuracy of the models the false and true positive and negative values are required, and was obtained using a confusion matrix plot depicted in Figure 3.11.

CHAPTER FOUR
RESULTS AND DISCUSSIONS

4.1 Introduction

The results obtained from testing the developed scheme are presented and discussed in this chapter. In this research, the developed scheme was tested and evaluated using two sets of publicly available dataset and the developed ABU dataset.

4.2 Grayscale Conversion of Images for Caltech, PKU VehicleID and ABU Images

The acquired images of Caltech, PKU VehicleID and ABU images were converted to grayscale images using equation 2.3 in order to reduce the processing time of the algorithm. Sample results of the gray conversion of test images from the datasets are presented in Figure 4.1.



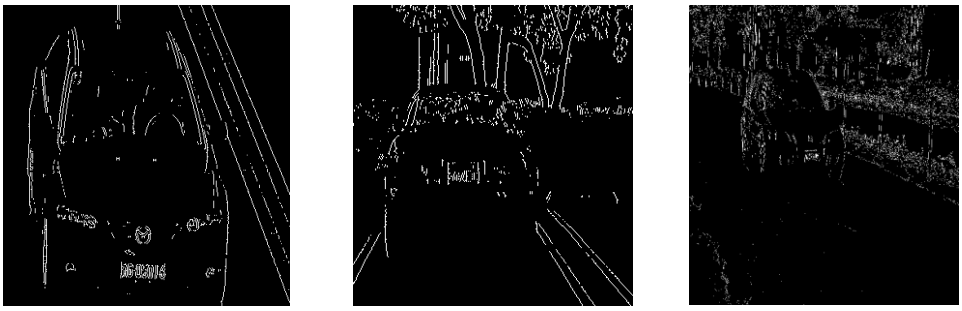
(a) Caltech image (b) PKU image (c) ABU image

Figure 4.1: Result of the Converted Grayscale Images from the Dataset

The operation eliminated all the colour information of the image (chrominance) leaving only the luminance. The significance of this operation is that lesser time is required to process the 8-bit gray channel image than the 24-bit RGB channel.

4.3 Sobel Edge Detection on Caltech, PKU VehicleID and ABU Image

The results of the Sobel operation on the images are presented in Figure 4.2



(a) Caltech image (b) PKU image (c) ABU image

Figure 4.2: Sobel Edge Detected Images

This operation detects the contours and boundaries of the object in the image. The detected edges show discontinuity in intensity values of the image such that areas or regions with high intensities are highlighted as shown. Edges of the images containing background clutters are also detected.

4.4 Candidate Result Obtained by the Edge Density Filter

The result of the edge density filter operation on the images from the various datasets are presented in Figure 4.3.



(a) Caltech image (b) PKU image (c) ABU image

Figure 4.3: Edge Density Filtered Images.

The edge density filter operation highlights the high-density regions i.e. regions that contain only white pixels. License plates are likely to be found in these regions as from Figure 4.3, the region of high density is clearly shown and represented by the white pixels while the

dark regions in the image are the low-density regions. Using this approach, the edges of all the test images were correctly found and detected.

4.5 Detection Result Obtained by the Developed LPD Approach on Caltech, PKU VehicleID and ABU Images

The detected results on the datasets are presented in Figure 4.4, 4.5 and 4.6 respectively. A randomly selected 50 vehicle images from each dataset were used to test the detection performance of the developed scheme.

4.5.1 Detection Result on the Caltech Dataset

The results of the detection scheme on six (6) random images selected from the Caltech dataset are as shown in Figure 4.4.



Figure 4.4: Detection Result on Caltech Dataset

Figure 4.4: shows the result of the developed LPD approach on the Caltech dataset. The image dataset is made up of rear view of single image vehicle. A rectangular and square bounding box are drawn in all the regions that exhibit the license plate properties based on the defined criteria. The green boxes denote the detected candidate regions; the red box represent the final detected license plates. It is evident that the developed scheme detected these regions correctly and accurately with few candidate regions.

4.5.2 Detection Result on the PKU VehicleID Dataset

The results of the detection scheme on six (6) random images selected from the PKU VehicleID dataset are as shown in Figure 4.5.



Figure 4.5: Detection Result on PKU VehicleID Dataset

Figure 4.5: shows the result of the developed LPD approach on the PKU VehicleID dataset. The image dataset is made up of rear view of single image vehicles. A rectangular and square

bounding box are drawn in all the regions that exhibit the license plate properties based on the defined criteria. The green boxes denote the detected candidate regions; the red box represent the final detected license plates. It is observed that a smaller number of candidate regions were detected on the PKU VehicleID dataset than on the Caltech and the ABU dataset. This is because the images seem to contain only the object (vehicle) and as such are free from road and lane distractions (background noise).

4.5.3 Detection Result on the PKU Dataset.

The results of the detection scheme on six (6) random images selected from the ABU dataset are as shown in Figure 4.6.



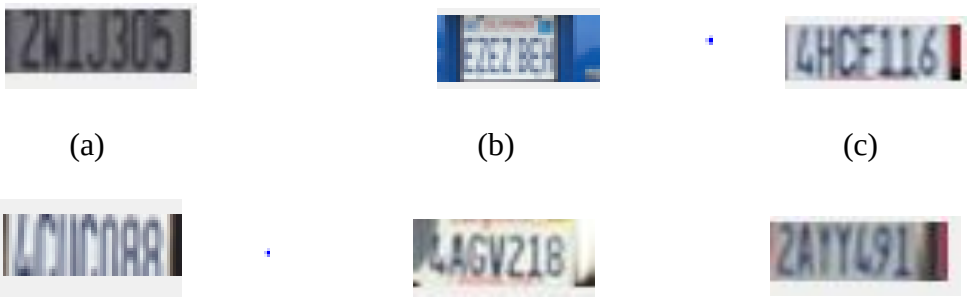
Figure 4.6: Detection Result on ABU Dataset

Figure 4.6: shows the result of the developed LPD approach on the ABU dataset. The image dataset is made up of single and multiple vehicle images. A rectangular and square bounding

box are drawn in all the regions that exhibit the license plate properties based on the defined criteria. The green bounding boxes denote the detected candidate regions; the red box represent the final detected license plates. It is observed that a greater number of candidate regions were detected from the ABU dataset than the Caltech and PKU VehicleID image datasets. This is because the images contain so many objects around their surroundings.

4.6 Extracted License Plate Number of Caltech, PKU VehicleID, and ABU Images

The extracted license plates from the detection process are given in Figures 4.7, 4.8 and 4.9 respectively for Caltech, PKU VehicleID and ABU Images. Figure 4.7 shows the extracted license plates from the Caltech dataset sample images. Some of the extracted license plates (Figure 4.7 (a) and (f)) contain some extra pixels from the vehicle image thus limiting its recognition.



(a) (b) (c)

(d) (f) (h)

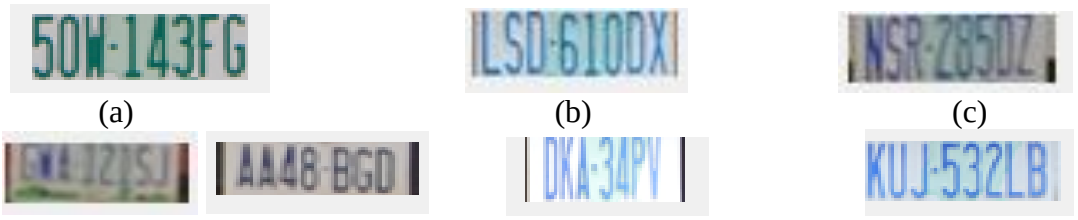
Figure 4.7: Extracted Caltech License Plate Number

However, from Figure 4.8: it is observed that all the extracted license plates are correctly extracted and contain no extra pixels as the license plate are uniformly spread out within the bounding box.



Figure 4.8: Extracted PKU VehicleID License Plate Number

Figure 4.9: shows that only clearly taken and pre-processed images are extracted correctly without any extra pixels due to the surroundings. The vehicle image with the extracted license plate of Figure 4.9 (c) was taken at a distance of about 10m away from the vehicle and contain no extra pixel as the number plate is clearly visible. Also, when multiple vehicles were captured as in Figure 4.9 (d), the developed scheme was able to detect the multiple license plates irrespective of their distances away.



(a) (b) (c)

(d) (e) (f)

Figure 4.9: Extracted ABU License Plate Number

4.7 Analysis of the Developed License Plate Detection

The testing of the developed algorithm was carried out on 50 randomly selected images of single vehicles obtained from the Caltech, PKU VehicleID and ABU datasets captured under clear conditions and comparison carried out against the existing scheme of Yuan *et al.*, (2016).

4.7.1 Detection Performance on the Dataset

To determine the detection performance, it is considered that the license plate is correctly detected if the plate region is completely encompassed within the red bounding box as shown in Figures 4.4 to 4.6. The result obtained from each dataset is as presented in Table 4.1.

Table 4.1: Result of Detection Rate

Datasets	No of Correctly Detected Image	Developed Scheme (%)	Existing Scheme (%)
Caltech	45	90	82
PKU	49	98	92
ABU	48	96	86

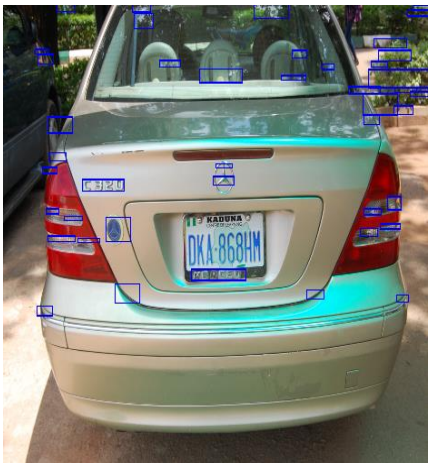
The detection result presented in Table 4.1 shows that the developed scheme achieved a higher detection rate on the test images than the existing scheme. It is evident that the PKU vehicle ID dataset achieved the highest detection rate as a result of its standardized dataset containing less noise. This implies that the detection rate is dependent on the quality of the images in the dataset.

4.7.2 Failed Detection

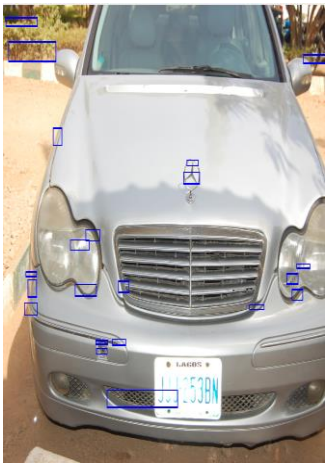
The experimental results presented in Table 4.1 shows that the license plates were not correctly detected in some of the images. The failed detections, as depicted in Figure 4.10, were mainly as a result of the salt and paper noise present in the images, such as reflections, deteriorating state of the plates, etc. which were not completely dealt with during the pre-processing stage or when the license plates were not clearly captured or when the distance between the camera and the vehicle is very close.



(a) ABU Image



(b) ABU Image



(c) ABU Image



(d) Caltech Image



(e) Caltech Image



(f) Caltech Image

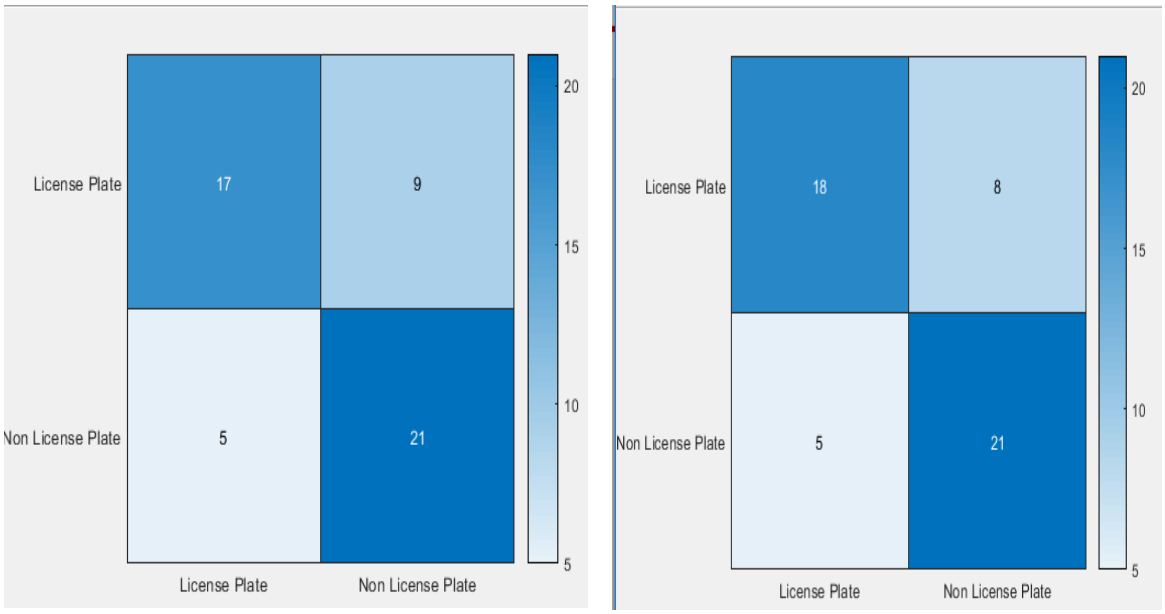


(g) PKU Image

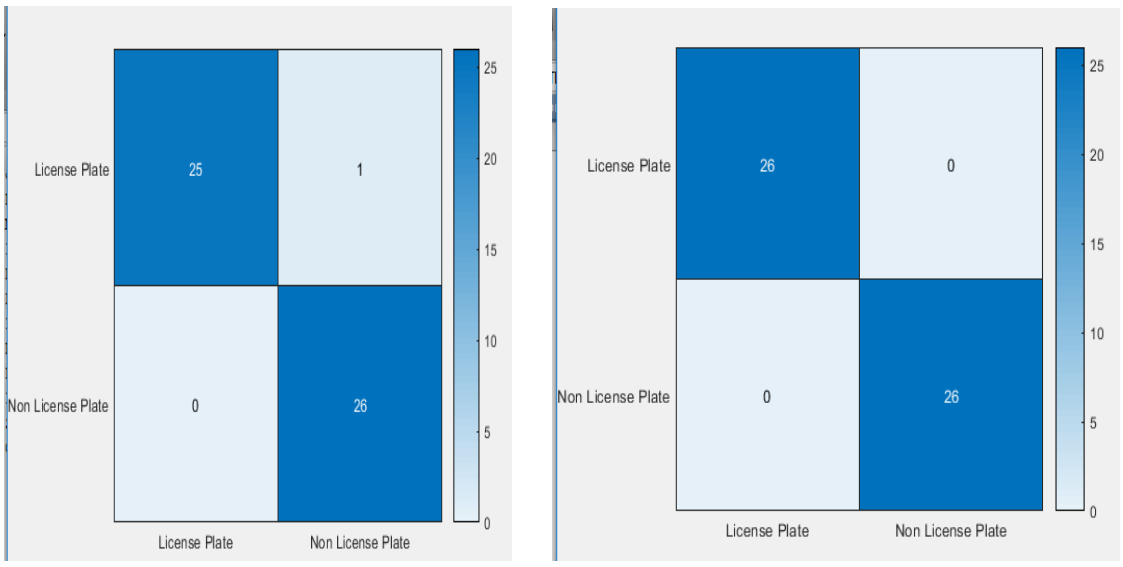
Figure 4.10: Samples of Images that Resulted in Failed Detection

4.7.3 Confusion Matrix of the Classifier over the Test Set

The result of confusion matrix over the test image are presented in Figure 4.11. the confusion matrix is generated for both the developed model and the existing scheme.



a) Confusion Matrix Result of the Existing Scheme



b) Confusion Matrix Result of the Developed Scheme.

Figure 4.11. Confusion matrix of the classifier over test dataset

Considering Figure 4.11 it can be observed that the confusion matrix derived from the developed scheme achieved a better classification task than that of the existing scheme which uses feature descriptor techniques (colour silence features) for its feature representation.

4.7.4 Evaluating the Precision, Recall and Classification Accuracy of the Scheme

To evaluate the precision, recall and classification accuracy the confusion matrix result obtained from the test images depicted in Figure 4.11 was used. The matrix consists of the true and false positive and negative prediction for the actual and predicted labels. The confusion matrix derived from the Figure 4.11 are summarized in Tables 4.2 and 4.3. Each of the process were carried out on the 3 datasets (Caltech, PKU Vehicle ID and ABU images), using the same number of test images (126). The true and false values obtained for each dataset are also presented.

Table 4.2: Precision and Recall Rate using the Existing Scheme on the Dataset

Dataset	**TP**	**TN**	**FP**	**FN**	**Precision (%)**	**Recall (%)**	**Accuracy (%)**
Caltech	17	23	3	9	85	65.38	76.92
PKU VehicleID	23	22	4	3	97.91	99.2 9	98.61
ABU	21	20	6	5	84.0	87.50	97

From the confusion matrix presented in Table 4.2, it is observed that using the existing scheme of Yuan *et al*., (2016) on the PKU VehicleID dataset obtained a higher precision, recall and accuracy values than the Caltech and ABU dataset. This is because the algorithm

generalises well on PKU dataset as a result of the standardized dataset, which results in the minimized training error.

Table 4.3: Precision and Recall Rate using the Developed Scheme on the Dataset

Dataset	TP	TN	FP	FN	Precision (%)	Recall (%)	Accuracy (%)
Caltech	25	26	0	1	85.1	98.50	98.08
PKU VehicleID	23	23	3	3	97.91	97.91	100
ABU	26	25	1	0	95.8	100	99.82

The results presented in Table 4.3 shows that the developed scheme obtained higher precision, recall and accuracy rates when compared with the existing scheme. This can be attributed to the utilization of deep learning approach of transfer learning in the developed scheme.

4.8 Result of Detection on ABU Images Considering Various Environmental Conditions

To further evaluate the robustness of the developed scheme, the scheme was tested on images of ABU dataset captured under defined environmental conditions. Table 4.4 shows the various conditions of the images under which the developed scheme was evaluated.

Table 4.4: Result of the Detection Rate on ABU Dataset

Conditions	No of Test Images	No of detections	Detection Rate (%)
Rainy	8	8	100
Night	12	10	83.33
Multiple vehicles	6	6	100

Considering the conditions presented in Table 4.4, the developed scheme correctly detected the license plates in all the scenarios except for night. The two instances of the failed detection are as shown in Figure 4.12.



(a) (b)

Figure 4.12: Images of Failed Detection

The failed detection in (a) is as a result of the close distance at which the image was taken while (b) is a result of the license plate region been affected by reflected glare.

CHAPTER FIVE
CONCLUSION AND RECOMMENDATION

5.1 Summary of Findings

This research work developed a deep learning-based license plate detection scheme, which was aimed at improving the rate of detection and classification accuracy of license plate in real time and reducing the rate of achieving a false positive classification result. This was successfully achieved and the following findings were recorded:

1. Improved classification accuracy
2. Fast computational time as a result of the pre-trained model used.
3. The algorithm generalizes well as a result of the transfer learning process.

5.2 Conclusions

This research work developed a license plate detection scheme that is capable of detecting and classifying license plate from a given image vehicle.

An Ahmadu Bello University (ABU) dataset containing 126 images of vehicle captured under different environmental conditions was developed to validate the performance of the scheme.

The developed scheme was based on current image processing and computer vision techniques, to accurately detect and classify the license plate number from a vehicle with less false positive prediction.

The algorithm was tested on two standard datasets (Caltech and PKU VehicleID dataset) and on the developed ABU dataset, the experimental result shows that the developed scheme outperformed the existing one with a large margin.

5.3 Limitation

The ABU dataset could not be as populated as expected due to security challenges encountered in obtaining the images.

5.4 Significant Contributions

The significant contributions of this research work are as follows:

1. Development of an ABU dataset of vehicle images captured under different environment conditions.
2. Development of an efficient transfer learning based method of license plate extraction for an efficient detection performance.
3. The developed deep learning based license plate detection scheme showed
    i. An improvement of 3.96% and 14.75% in the precision and recall rate respectively, and 8.15% improvement in classification accuracy when compared with the existing scheme.
    ii. It also achieved a detection rate of 98 % on the PKU VehicleID dataset, 90% on the Caltech dataset and 96% on the ABU dataset in sunny conditions and achieved a detection rate of 98.5% in detecting license plate from different environmental conditions.

5.5 Recommendations for Further Work

The following possible areas of further work are recommended for consideration for future research:

1. Researcher should explore other pre-trained model like Residual neural network (RESNET), GoogleNET and Visual geometric group network (VGGNet).

2. The algorithm can be implemented on hardware using raspberry pi.
3. Character segmentation and recognition should be considered on Nigerian license plates.
4. Researchers can explore on detecting the license plate number in video streams rather than still images.
5. More dataset containing Nigeria vehicle images should be developed.

REFERENCES

Agarwal, A., & Goswami, S. (2016). An Efficient Algorithm for Automatic Car Plate. *2016 Second International Conference on Computational Intelligence & Communication Technology*. https://doi.org/10.1109/CICT.2016.133

Aggarwal & Maini. (2009). Study and Comparison of Various Image Edge Detection Techniques, *147002*(3), 1–12.

Al-ghaili. (2013). Vertical-Edge-Based Car-License-Plate Detection Method, *62*(1), 26–38.

Arias-castro, E., & Donoho, D. L. (2009). arXiv : math / 0612422v2 [math . ST] 20 Apr 2009, *37*(3), 1172–1206. https://doi.org/10.1214/08-AOS604

Attah, A. B. A. A. . & A. A. M. (2016). Development of Nigeria Vehicle License Plate Recognition AND Classification System, *12*(1), 125–133.

Bansal, V., & Sinha, R. M. K. (2012). Segmentation of Touching and Fussed Devanagari Characters.

Biswas, R., & Sil, J. (2012). An Improved Canny Edge Detection Algorithm Based on Type-2 Fuzzy Sets. *Procedia Technology*, *4*, 820–824. https://doi.org/10.1016/j.protcy.2012.05.134

Boyat, A. K., & Joshi, B. K. (2015). A Review Paper : Noise Models in Digital Image Processing, *6*(2), 63–75.

Bradley, D., & Roth, G. (2007). Adaptive Thresholding using the Integral Image. *Journal of Graphics Tools*, *12*(2), 13–21. https://doi.org/10.1080/2151237X.2007.10129236

Caltech. (2005). Computational Vision: Archive. Retrieved from http://www.vision.caltech.edu/Image Datasets/cars markus/cars markus.tar

Dewan, S., Bajaj, S., & Prakash, S. (2015). Using Ant's Colony Algorithm for improved segmentation for number plate recognition. *2015 IEEE/ACIS 14th International Conference on Computer and Information Science, ICIS 2015 - Proceedings*, (June 2015), 313–318. https://doi.org/10.1109/ICIS.2015.7166612

Du, S., Ibrahim, M., Shehata, M., & Member, S. (2013). Automatic License Plate Recognition (ALPR): A State-of-the-Art Review, *23*(2).

Fisher, R., Perkins, S., Walker, A. and Wolfart, E. (2003). Grayscale Images , Available at: https://homepages.inf.ed.ac.uk/rbf/HIPR2/gryimage.htm.

Fomani, &shahbahrami. (2017). License Plate Detection using Adaptive Morphological Closing and Local Adaptive Thresholding, (Ipria), 146–150.

Gao, W., Yang, L., Zhang, X., & Liu, H. (2010). An improved Sobel edge detection.

Proceedings - 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2010, *5*, 67–71. https://doi.org/10.1109/ICCSIT.2010.5563693

Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S., & Shet, V. (2016). Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks arXiv : 1312 . 6082v4 [cs . CV] 14 Apr 2014, 1–13.

Han, P. U., Han, W. E., Wang, D., & Zhai, Y. (2003). Car license plate feature extraction and recognition based on multi-stage classifier. *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, (November), 128–132. https://doi.org/10.1109/ICMLC.2003.1264456

Hermawati, F. A., Koesdijarto, R., Fourier, T., & Model, H. M. (2010). A Real-Time License Plate Detection System, 97–106.

Hossen, M. K., Roy, A. C., Chowdhury, S. A., & Deb, K. (2018). License Plate Detection and Recognition System based on Morphological Approach and Feed-Forward Neural Network, *18*(5), 36–45.

Jenkins, H. . (2007). Wall Street Journal (Eastern edition), January 31, 2007, A12. *Business World: Future of Driving*.

Jihui, T. U., Jiang, Z., & Wu, X. (2015). A Sobel-TV based Hybrid Model for De-noising Remote Sensing Image with Gaussian and Salt-pepper Noise, *8*(3), 325–336.

Kaur, A., Baghla, S., & Kumar, S. (2015). Study of Various OF Various Character Segmentation Techniques using Handwritten Off-Line Cursive Word : A Review, (3), 154–158.

Kim, J.-H., Kim, S.-K., Lee, S.-H., Lee, T.-M., & Lim, J. (2018). License Plate Detection and Recognition Algorithm for Vehicle Black Box. Retrieved from https://doi.org/10.1109/CACS.2017.8284273

Lalimi, M. A., Ghofrani, S., & McLernon, D. (2013). A vehicle license plate detection method using region and edge based methods. *Computers and Electrical Engineering*, *39*(3), 834–845. https://doi.org/10.1016/j.compeleceng.2012.09.015

Li, H., & Shen, C. (2016). Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs.

Nguwi, Y. Y., & Lim, W. J. (2015). Number Plate Recognition in Noisy Image, (Cisp), 476–480.

Nguyen, C. T., & Nguyen, T. B. (2015). Reliable Detection and Skew Correction Method of License Plate for PTZ camera-based License Plate Recognition System, 1013–1018.

Saravanan, C. (2010). Color image to grayscale image conversion. *2010 2nd International Conference on Computer Engineering and Applications, ICCEA 2010*, *2*(June), 196–

199. https://doi.org/10.1109/ICCEA.2010.192

Selmi, Z., Ben Halima, M., & Alimi, A. M. (2017). Deep Learning System for Automatic License Plate Detection and Recognition. *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 1132–1138. https://doi.org/10.1109/ICDAR.2017.187

Sharma,S. Cheeran, A. Anumol, S. (2017). *A SVM based Character Recognition System*. https://doi.org/https://doi.org/10.1109/RTEICT.2017.8256890

Singh, M. K., Baluja, P. G. S., & Sahu, D. P. (2017). Analysing Deep Learning as A Machine Learning, *5*(Vi), 790–796.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., … Member, S. (2016). Convolutional Neural Networks for Medical Image Analysis : Full Training or Fine Tuning ?, *35*(5), 1299–1312.

Vala, M. H. J., & Baxi, A. (2013). A review on Otsu image segmentation algorithm. *International Journal of Advanced Research in Computer Engineering and Technology*, *2*(2), 387–389.

Yuan, Y., Zou, W., Zhao, Y., Wang, X., Hu, X., & Komodakis, N. (2016). A Robust and Efficient Approach to License Plate Detection, *7149*(c), 1–13. https://doi.org/10.1109/TIP.2016.2631901

Zhao, Y., Yuan, Y., Bai, S., Liu, K., & Fang, W. (2011). Voting-based license plate location. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 314–317. https://doi.org/10.1109/ITSC.2011.6082854

Zhigang, C., Yueli, C., & Aihua, C. (2012). An Multi-scale Edge Detection Approach. *Physics Procedia*, *25*, 1616–1620. https://doi.org/10.1016/j.phpro.2012.03.284

Zhou, W., Li, H., Lu, Y., & Tian, Q. (2012). Principal visual word discovery for automatic license plate detection. *IEEE Transactions on Image Processing*, *21*(9), 4269–4279. https://doi.org/10.1109/TIP.2012.2199506

APPENDICES

APPENDIX A: ABU Dataset



Images of Vehicle Captured at Night



Images of Vehicle Captured During the Rain



Images Containing Multiple vehicle

APPENDIX B1: M-File for Pre-Processing Operation

```
Clear
clc
close all
% I = imread ('testImages/G1 (588).jpg');
% I = imread('testImages/N2.jpg');
    I = imread('ABU22/DSC_0937.jpg');
% I = imread ('pku dataset/G1 (576).jpg');
figure; imshow(I)
title ('original image')
% % Image preprocessing
[hi, wi, ~] = size(I);
dw = 3;
dh = 2;
ws = wi/dw;
hs = hi/dh;
Ir = imresize (I, [hs, ws], 'bilinear');
figure; imshow(Ir)
title ('resized image')
Ig = rgb2gray(Ir);
figure; imshow(Ig)
title ('grayscale image')
% % Candidate extraction
% edge detection
Ie = edge (Ig, 'Sobel', [], 'vertical');
figure; imshow(Ie)
title ('sobel edge detection image')
% Line density filter
If = LDF (Ie, 'horizontal', 10, 6, 0.03);
figure; imshow(If)
title ('Line Density Filter image')
% connected component labeling
candidate = CCL(If);
figure; imshow(Ir)
for i= 1: numel(candidate)
    hold on
    rectangle ('Position', candidate(i). BoundingBox, 'EdgeColor','b',
...
        'LineWidth',1')
    hold off
    CLP = [];
    for c=1:3
        CLP(:c) = imresize(imcrop(Ir(:c), candidate(i). BoundingBox),
[20,60]);
    end
    CLP = uint8(CLP);
    fileName = strcat ('CLP_', num2str(i),'.jpg');
    imwrite (CLP, fileName)
end
title ('candidate license plate image')
```

APPENDIX B2: M-File for Connected Component Labeling

```
function candidate = CCL(If)
% clc
% clear
% close all
% a = imread('filterImg.jpg');
If = If > 100;
% figure; imshow (If)
 % return
% a = imread('circlesBrightDark.png');
% If = a < 100;
candidate = regionprops (If, 'BoundingBox');
ind = [];
for i= 1: numel(candidate)
    w = candidate(i). BoundingBox (3);
    h = candidate(i). BoundingBox (4);
    C1 = (w>24) && (w<256);
    C2 = (h>6) && (h<52);%(h<32);
    C3 = (w*h>256) && (w*h<15000); %(w*h<4096);
    C4 = w/h < 6;
%    [C1 C2 C3 C4]
    if ~ (C1 && C2 && C3 && C4)
        ind = [ind, i];
%        [w h]
%         hold on
%         rectangle ('Position', candidate(i). BoundingBox,
'EdgeColor','b', ...
%             'LineWidth',1')
%         hold off
%         drawnow
%         pause
    end
    end
candidate(ind) = [];
end
```

APPENDIX B3: M-File for Edge Density Filter

```
function Ef = LDF (Eb, direction, Tmin, Tgap, Td)
% clear
% clc
% Eb = double(imread('edge_image.jpg'));
% figure; imshow(uint8(Eb))
% Eb = [zeros (1, 20) 1 1 0 0 0 0 1 1 1 0 0 1 1 zeros (1, 20)
%       zeros (1, 20) 1 0 0 0 0 0 1 0 1 1 1 1 1 zeros (1, 20)
%      ];
% Tmin = 20;
% Tgap = 8;
% Td = 0.15;
Eb = double(Eb);
if strcmp (direction, 'vertical')
    Eb = Eb';
end
Ef = zeros(size(Eb));
Mhl = horzLenMap(Eb);
for l = 1: numel(Mhl)
    ind = find(Mhl{l}. map<-Tmin-1);
      for i=1: numel(ind)-1
%         display ('==============================')
%         for j = i+1: numel(ind)
%             display ('--------------------------')
            j=i+1;
            lb1 = abs(Mhl{l}. map(ind(i)));
            lb2 = abs(Mhl{l}. map(ind(j)));
           C1 = (lb1 >= Tmin) && (lb2 >= Tmin);
            C2 = (i == 1) && (lb1 < Tmin) && (lb2 >= Tmin);
            C3 = (j == numel(ind)) && (lb1 >= Tmin) && (lb2 < Tmin);
           if C1 || C2 ||C3
%                 display ('--------------------------')
                m = ind(i)+1;
                n = ind(j)-1;
                l3 = Mhl{l}. map (m: n);
                wl = sum(l3(find(l3>0)));
                bl = abs(sum(l3(find(l3<0))));
                di = wl/(wl+bl);

%
%     if l>300
%         clc
%         display(['================Line', num2str(l),
'===============']);
%         [Mhl{l}. map
%             Mhl{l}. begin]
%         [C1 C2 C3]
%         [lb1 lb2 di]
%         [i, j]
%         pause
%     end
%
                x = Mhl{l}. begin(m); %sum(abs(Mhl{l}. map(1:m-1))) +1
```

```
                y = Mhl{l}. begin(n+1)-1; % x + sum(abs(Mhl{l}. map (m:
n)))-1

                if (di >= Td)
                    Ef (l, x: y) = 255;
                else
                    Ef (l, x: y) = 0;
                end
            end
            if C1
                if y-x< Tgap
                    Ef (l, x: y) = 0;
                end
            end
            %         end
    end
%     figure; plot (Eb (l, :))
%     title (['Line', num2str(l)])
%     %             xlim ([0 size(Eb,2)])
%     ylim([0 500])
%     pause
%     if ~isempty(dl)
%         i
%         dl
%         pause
%     end
end
% Eb = im2bw(Eb, 0.5);
% figure; imshow(uint8(Eb))
% size(Eb)
if strcmp (direction, 'vertical')
    Ef = Ef';
end
% Ef = Ef > 100;
End
```

APPENDIX B4: M-File for Candidates Region Extraction

```
function extractCandidates (imageDir, candidateDir)
% clear
% clc
% close all
imageDir = 'C:\Users\success\Documents\MATLAB\Blessing Oluchi\LPD\G1';
candidateDir = 'C:\Users\success\Documents\MATLAB\Blessing
Oluchi\LPD\C1';
count = 1;
images = dir (fullfile (imageDir, '/', '*.jpg'));
for i=1: numel(images);
  imFileName = strcat (imageDir, '/', images(i).name);
    I = imread(imFileName);
     figure; imshow(I)
     % % Image preprocessing
    [hi, wi, ~] = size(I);
    dw = 3;
    dh = 2;
    ws = wi/dw;
    hs = hi/dh;
    Ir = imresize (I, [hs, ws], 'bilinear');
    %     figure; imshow(Ir)
       Ig = rgb2gray(Ir);
      Candidate extraction
    edge detection
    Ie = edge (Ig, 'Sobel', [], 'vertical');
        figure; imshow(Ie)
     Line density filter
    If = LDF (Ie, 'horizontal', 10, 6, 0.03);
        figure; imshow(If); title (Edge Density Filter')
    connected component labeling
    candidate = CCL(If);
        figure; imshow(Ig)
    for c= 1: numel(candidate)
                 hold on
                rectangle ('Position', candidate(c). BoundingBox,
'EdgeColor','b', ...
                    'LineWidth',1')
                 hold off
        CLP = [];
        for ch=1:3
            CLP(:ch) = imresize(imcrop(Ir(:ch), candidate(c).
BoundingBox), [20,60]);
        end
        CLP = uint8(CLP);
        fileName = strcat (candidateDir, '/C_', num2str(count),'.jpg');
        imwrite (CLP, fileName)
        count = count+1;
         end
  end
end
```

APPENDIX B5: M-File for The Transfer Learning and Testing

```
% transfer learning and trainning
% Get training images
candidatesSet = imageDatastore('Candidates',...
    'Include Subfolders',true,....
    'LabelSource','foldernames');
% Split into training and testing sets
[trainingSet, testSet] = splitEachLabel(candidatesSet,0.7,'randomized');
% Determine the number of candidate classes
numClasses = numel(categories(candidatesSet.Labels));
% Create a network by modifying AlexNet
% Get the layers from AlexNet
net = alexnet;
layers = net.Layers;

% % % % % % % %
% Display only network architecture.
% The network has only five convolutional layers
% and three fully connected layers.
layers

% The first layer, which is the image input layer,
% requires images of size 227-by-227-by-3,
% where 3 represent the number of color channels.
inputSize = is layers (1). InputSize
%data augmentation
  ImageAugumenter=ImageDataAugmenter (…
  'RandXRotation',   [ -90, 90] ,…
  'RandFlip', [vertical  ], …
  'DataAugmentation', ImageAugumenter);
% Modify the classification and output layers
layers(end-2) = fullyConnectedLayer(numClasses,'Name','fc8');
layers(end) = classificationLayer('Name','output');
% Set training algorithm options
% Lower the learning rate for transfer learning
options = trainingOptions ('sgdm','InitialLearnRate', 0.001);
% Perform training
[LPDnet, info] = trainNetwork (augimdsTrain, layers, options);
figure; plot (info. TrainingLoss)
% Use the trained network to classify test images
predictedLabels = classify (LPDnet, augimdsTest);
% Display eight sample test images with their predicted labels.
idx = randperm (numel (testSet.Labels),8);
figure
for i = 1: numel(idx)
    subplot(4,2,i)
    I = readimage(testSet,idx(i));
    label = predictedLabels(idx(i));
    imshow(I)
    title(char(label))
end
% Evaluate the results
% Calculate the accuracy
```

```
accuracy = mean(predictedLabels == testSet.Labels)
% Visualize the confusion matrix
[candidateconf,candidatenames] =
confusionmat(testSet.Labels,predictedLabels);
figure;
heatmap(candidatenames,  candidatenames, candidateconf);
```