

VEHICLE LICENSE PLATE NUMBER RECOGNITION
(VLPNR) UNDER CHALLENGING CONDITION USING
YOLO-NAS AND OPTICAL CHARACTER
RECOGNITION

SIA PING HUI

MATHEMATICS WITH COMPUTER GRAPHICS
FACULTY OF SCIENCE AND NATURAL RESOURCES
UNIVERSITI MALAYSIA SABAH

2024

VEHICLE LICENSE PLATE NUMBER RECOGNITION (VLPNR) UNDER CHALLENGING CONDITION
USING YOLO-NAS AND OPTICAL CHARACTER RECOGNITION

SIA PING HUI

THIS DISSERTATION IS SUBMITTED FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF BACHELOR OF SCIENCE WITH HONOURS

MATHEMATICS WITH COMPUTER GRAPHICS PROGRAMME

FACULTY OF SCIENCE AND NATURAL RESOURCES

UNIVERSITI MALAYSIA SABAH

2024

DECLARATION

I hereby declare that all the work presented in this dissertation is my own original work and the result were of my own investigation except for certain citations that had been duty acknowledged. Formulation and ideas in this dissertation were taken from sources such as books, articles, research paper, and other electronic sources were cited as well.

Lce

SIA PING HUI

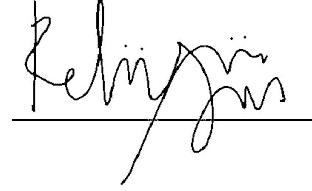
(BS20110144)

19 Jan 2024

CERTIFICATION

1. Supervisor
(RECHARD LEE)

Signature

A handwritten signature in black ink, appearing to read "Richard Lee", is written over a horizontal line.

ACKNOWLEDGEMENTS

I extend my sincere appreciation to several individuals who played a pivotal role in the successful completion of my Final Year Project, titled "Vehicle License Plates Number Recognition under Challenging Conditions."

First and foremost, I express my gratitude to my supervisor, Mr. Rechar Lee, for his invaluable guidance and unwavering support throughout the project. His patience, feedback, and mentorship were instrumental in my growth as a researcher, making the completion of the thesis and timely development of the license plate recognition model possible.

I am also deeply thankful to my examiner, Associate Professor Dr. Abdullah Bade, whose insightful suggestions during the system stage and critical corrections to my thesis significantly elevated the overall quality and organization of the project.

My heartfelt appreciation extends to my parents for their constant encouragement and support, serving as a consistent source of strength throughout the development process.

Additionally, I want to acknowledge the contributions of my Mathematics with Computer Graphics lecturers, whose dedicated teaching provided the knowledge and skills that formed the foundation of my project, contributing to its success.

Lastly, I extend my sincere thanks to my friends for their invaluable contributions. Their willingness to share ideas and offer support during the thesis writing process had a profound impact. I am truly fortunate to have such a supportive network.

ABSTRACT

Vehicle License Plate Recognition (VLPR) is a critical technology with applications in traffic management, law enforcement, and security. This abstract highlights the essential components, applications, and challenges faced by VLPR systems. Adverse weather conditions and low lighting hinder recognition accuracy, necessitating innovative solutions. Previous research demonstrates the need for improved recognition rates, motivating the proposal of a combined object detection and optical character recognition method to enhance accuracy, particularly in hazy environments. The objective of this project is to design a vehicle license plate recognition model using YOLO-NAS object detection and EasyOCR as optical character recognition, and to recognize and identify the vehicle license plate number under challenging conditions using the model. The model was trained and tested on different experiments to identify the performance of the model. The proposed model is to use YOLO-NAS as an object detection model and optical character recognition model. The proposed model was tested on recognizing vehicle license plate under four different condition: (1) Visible license plate under normal condition; (2) Vehicle license plate under occluded/obscured condition; (3) vehicle license plate under foggy/hazy weather condition, (4) vehicle license plate under raining conditions. The test done on the model was to test the performance when running the model, the rate of success detection of vehicle license plate of the model, and the accuracy of the model in recognizing the license plate number. The model was trained using 1013+ images dataset from roboflow and was tested with valid dataset of 50 normal condition, 10 obscured condition, 10 hazy condition, and 10 raining condition. All the tests done on the model were in 3 iterations, and the results were taken and analyzed. Overall, the proposed model yields a results of 93.22% when recognizing license plate under normal conditions, 93.0% when recognizing under obscured condition, 93.21% under hazy condition, and 85.9% under raining condition. The result had shown that the proposed method is valid and able to perform on-par with existing method for recognition under challenging conditions.

PENGECAMAN NOMBOR PLAT KENDERAAN DALAM KEADAAN YANG MENCABAR

ABSTRAK

Pengecaman Nombor plat Kenderaan (VLPR) adalah teknologi penting dengan pelbagai aplikasi dalam pengurusan trafik, penguatkuasaan undang-undang, dan keselamatan. Abstrak ini menyorot komponen penting, aplikasi, dan cabaran yang dihadapi oleh sistem VLPR. Keadaan cuaca buruk dan pencahayaan rendah menghalang ketepatan pengecaman, memerlukan penyelesaian inovatif. Penyelidikan terdahulu menunjukkan keperluan untuk kadar pengecaman yang lebih baik, memotivasikan cadangan untuk menggunakan gabungan pengesanan objek dan pengecaman aksara optik untuk meningkatkan ketepatan, terutamanya dalam persekitaran berkabus. Objektif projek ini adalah untuk mereka bentuk dan membina model pengecaman nombor plat kenderaan menggunakan YOLO-NAS pengesanan objek dan pengecaman aksara optik, serta mengenali dan mengenal pasti nombor plat kenderaan dalam keadaan mencabar menggunakan model tersebut. Model telah dilatih dan diuji dalam pelbagai eksperimen untuk mengenal pasti prestasi model. Model yang dicadangkan adalah menggunakan YOLO-NAS sebagai model pengesanan plat kenderaan dan EasyOCR sebagai model pengecaman aksara optik. Model yang dicadangkan diuji untuk mengiktiraf nombor plat kenderaan dalam empat keadaan berbeza: (1) Nombor plat kenderaan yang di bawah keadaan normal; (2) Nombor plat kenderaan di bawah keadaan terhalang/samartutup; (3) Nombor plat kenderaan di bawah keadaan berkabus; (4) Nombor plat kenderaan di bawah keadaan hujan. Ujian telah dilakukan ke atas model untuk menguji prestasi semasa menjalankan model, kadar kejayaan pengesanan nombor plat kenderaan oleh model, dan ketepatan model dalam mengenali nombor plat. Model telah dilatih menggunakan dataset yang mengandungi 1013 imej yang didapati dari roboflow dan diuji dengan dataset yang mengandungi sebanyak 50 keadaan normal, 10 keadaan terhalang, 10 keadaan berkabus, dan 10 keadaan hujan. Semua ujian yang dilakukan ke atas model adalah dalam 3 iterasi, dan hasilnya diambil dan dianalisis. Secara keseluruhan, model yang dicadangkan menghasilkan sebanyak 93.22% apabila pengecaman nombor plat dalam keadaan normal, 93.0% dalam keadaan terhalang, 93.21% dalam keadaan berkabus, dan 85.9% dalam keadaan hujan. Hasilnya menunjukkan bahawa model yang dicadangkan adalah sah dan dapat berfungsi dengan baik dan dapat membandingkan dengan model yang wujud dalam pengecaman nombor plat kenderaan dalam keadaan yang mencabar.

Table of Content

	Pages
DECLARATION	ii
CERTIFICATION	iii
ACKNOWLEDGEMENTS	iv
ABSTRACT	v
ABSTRAK	vi
Table of Content	vii
LIST OF TABLES	xi
LIST OF FIGURES	xiii
LIST OF ABBREVIATIONS	xiv
LIST OF SYMBOL	xv
CHAPTER 1 Introduction	1
1.1 Overview	1
1.2 Problem Background	2
1.3 Problem Statement	3
1.4 Aim	3
1.5 Objective	3
1.6 Scope of Research	4
1.7 Justification	4
1.8 Thesis Organization	4
CHAPTER 2 Literature Review	6
2.1 Introduction	6
2.2 Object Detection	6
	vii

2.3	Vehicle License Plate Detection	7
2.4	Machine Learning-Based Technique	9
2.5	You Only Look Once (YOLO)	11
2.6	Image Processing	13
2.7	Image Pre-processing	14
2.7.1	Contrast Adjustment	14
2.8	Optical Character Recognition (OCR)	16
2.9	Vehicle License Plate Recognition	17
2.10	Super Resolution	19
2.11	Non-Neural Network VLPR	19
2.12	Evaluating Vehicle License Plate Recognition System	20
2.13	Summary	21
CHAPTER 3	Methodology	22
3.1	Introduction	22
3.2	Research Framework	23
3.2.1	Research Phase	23
3.2.2	Implementation Phase	24
3.3	System Framework	24
3.3.1	Input	25
3.3.2	Image Pre-processing	25
	The input image will undergo image pre-processing before the detection process. The pre-processing are super-resolution, contrast adjustment, and image sharpening.	25
3.3.2.1	Super Resolution	25
3.3.2.2	Contrast Adjustment	26
3.3.2.3	Image Sharpening	27

3.3.3	VLP Detection using YOLO-NAS	28
3.3.4	Extraction of VLP	28
3.3.5	Optical Character Recognition	29
3.3.6	Output	30
3.4	Experiment Setup	31
3.5	Benchmarking	32
3.6	Summary	33
CHAPTER 4 System Design And Implmentation		34
4.1	Overview	34
4.2	Use Case Diagram	34
4.3	Class Diagram	35
4.4	Sequence Diagram	37
4.5	Flowchart	38
4.6	Algorithm	40
4.7	Summary	45
Chapter 5 Result and Dicussion		46
5.1	Overview	46
5.2	Experimental Setup	46
5.2.1	Performance Test	46
5.2.2	Detection Test	47
5.2.3	Recognition Accuracy Test	47
5.3	Result and Analysis	48
5.3.1	Result from Performance Test	48
5.3.2	Result from Detection Test	56

5.3.3 Result from Recognition Accuracy Test	59
5.4 Comparison of proposed model with other methods	64
5.5 Summary	66
Chapter 6 Conclusion	67
6.1 Overview	67
6.2 Conclusion	67
6.3 Contribution	68
6.4 Limitations	68
6.5 Future work	69
6.6 Summary	70
References	71

LIST OF TABLES

Table No.	Pages
Table 3.1: Research Framework	23
Table 3.2: Algorithm of Super-resolution	25
Table 3.3: Algorithm of Contrast Adjustment	26
Table 3.4: Algorithm of Image Sharpening	27
Table 3.5: Algorithm of License Plate Detection	28
Table 3.6: Algorithm of Image Cropping	29
Table 3.7: Algorithm of Optical Character Recognition	29
Table 3.6: Experiment Setup	31
Table 3.7: Benchmarking	32
Table 4.1: Algorithm of the overall flow of the proposed model	41
Table 4.2: Algorithm of the training of the proposed model	41
Table 4.3: Algorithm of Image Pre-processing	42
Table 4.4: Algorithm of the license plate detection	43
Table 4.5: Algorithm of the image cropping	44
Table 4.6: Algorithm of the OCR	44
Table 5.1: Result of Performance Test on Normal Image Dataset	48
Table 5.2: Result of Performance Test on Obscured Image Dataset	51
Table 5.3: Result of Performance Test on Hazy Image Dataset	51
Table 5.4: Result of Performance Test on Rain Image Dataset	52
Table 5.5: Average Value of Performance Test	53
Table 5.6: Success Rate of Detection Test on Normal Dataset	56
Table 5.7: Success Rate of Detection Test on Obscured Dataset	56
Table 5.8: Success Rate of Detection Test on Hazy Dataset	56
Table 5.9: Success Rate of Detection Test on Rain Dataset	56
Table 5.10: Example of Image with worse conditions	57
Table 5.11: Result of Recognition Accuracy Test on Normal Dataset	59
Table 5.12: Result of Recognition Accuracy Test on Obscured Dataset	61
Table 5.13: Result of Recognition Accuracy Test on Hazy Dataset	62
Table 5.14: Result of Recognition Accuracy Test on Rain Dataset	62

Table 5.15:	The comparison of Success Detection Rate under normal circumstances	64
Table 5.17:	The comparison of recognition accuracy under normal circumstances	65
Table 5.18:	The comparison of recognition accuracy under challenging circumstances	65

LIST OF FIGURES

Figure No.	Pages
Figure 2.1: Example of Object Detection	6
Figure 2.1: CNN Steps of object detection	7
Figure 2.3: Canny edge detection on license plate	8
Figure 2.4: Example of Template Matching	8
Figure 2.5: Number plate classification based on multiple convolutional neural networks (CNNs)	9
Figure 2.6: One-Stage and Two-Stage Object Detection Model	10
Figure 2.7: Structure of YOLO	11
Figure 2.8: The evolution of YOLO	12
Figure 2.9: Image Before and After Contrast Adjustment	15
Figure 2.10: Example of OCR	16
Figure 2.11: Example of Vehicle License Plate Detection	18
Figure 2.12: License plate recognition with OCR	18
Figure 3.1: System framework	24
Figure 4.1: Use Case Diagram of the vehicle license plate recognition system.	34
Figure 4.2: Class Diagram	35
Figure 4.3: Sequence Diagram of license plate recognition system.	37
Figure 4.4: Flowchart of the model	39
Figure 4.5: Comparison of original image and pre-processed image	43
Figure 4.6: License plate detected in the pre-processed image by YOLO-NAS	43
Figure 4.7: License plate number recognized by EasyOCR	45
Figure 5.1: Image 46 in the normal dataset	54
Figure 5.2: Image 4 in the obscure dataset	54
Figure 5.3: Image 3 in the hazy dataset	55
Figure 5.4: Image 9 in rain dataset	55
Figure 5.5: Result of Accuracy test of normal dataset	63
Figure 5.6: (a) Image of non-character recognized as character.	
(b) Image of several non-related characters been recognized as character	64

LIST OF ABBREVIATIONS

VLP	Vehicle License Plate
AVLPR	Automated Vehicle Number/License Plate Recognition
OCR	Optical Character Recognition
RAM	Random Access Memory
YOLO	You Only Look Once
CPU	Computer Processing Unit
GPU	Graphic Processing Unit
CNN	Convolutional Neural Network
RCNN	Region-based Convolutional Neural Network

LIST OF SYMBOL

$+$	Addition
$=$	Equal
e	Exponential
$gaussian(x, y)$	Gaussian function
π	Pi
σ	Variance
a	alpha
b	beta

CHAPTER 1

Introduction

1.1 Overview

Vehicle License Plate Recognition (VLPR) stands as a pivotal technology, offering solutions across various domains such as traffic management, law enforcement, and security applications. This technology relies on sophisticated computer vision and image processing techniques, utilizing surveillance cameras to automatically extract license plate information from images or video streams. This comprehensive overview delves into the essential components, applications, and recent advancements in VLPR, drawing upon influential research studies to provide a nuanced understanding of this transformative technology.

In the domain of VLPR, the process begins with image capture and preprocessing. Surveillance cameras serve as the primary medium for capturing images or video streams of vehicles, with preprocessing steps including noise reduction and contrast enhancement to ensure optimal conditions for subsequent license plate recognition (Zhang *et al.*, 2016). Feature extraction follows, a critical step in isolating relevant license plate characteristics, such as alphanumeric characters and distinctive patterns. This step is crucial for the success of identification and matching processes (Chen *et al.*, 2017). Additionally, the integration of pattern recognition algorithms and machine learning models is integral to the VLPR system. These components enable the system to recognize and interpret license plate information, adapt to diverse scenarios, and continually enhance recognition accuracy over time (Zhang *et al.*, 2016; Chen *et al.*, 2017; Jain *et al.*, 2020).

The applications of VLPR extend across various sectors, significantly impacting traffic management, law enforcement, and security. In traffic management, VLPR systems play a vital role in monitoring and controlling vehicular movement, optimizing traffic flow, and enforcing traffic regulations (Zhang *et al.*, 2016). Law enforcement agencies leverage VLPR for automatic identification of vehicles involved in criminal activities, tracking stolen vehicles, and enforcing parking regulations (Chen *et al.*, 2017). Furthermore, VLPR seamlessly integrates into security

systems, enhancing access control, monitoring parking areas, and improving overall security in sensitive locations (Zhang *et al.*, 2016; Wang *et al.*, 2019).

As technology progresses, researchers have proposed innovative template-based approaches to enhance the adaptability of VLPR in unconstrained scenarios, including those with adverse weather conditions (Chen *et al.*, 2017). Challenges posed by adverse conditions, such as fog, rain, and low lighting, have spurred investigations into methods that enhance the robustness of VLPR, addressing issues related to image blurring and reduced visibility (Gong *et al.*, 2015).

1.2 Problem Background

Vehicle License Plate Recognition (VLPR) systems have become indispensable in modern traffic management, law enforcement, and security applications, leveraging advanced technologies such as image processing and machine learning. These systems are designed to automatically identify and extract license plate information from images or video streams captured by surveillance cameras (Zhang *et al.*, 2016). While VLPR has demonstrated significant success in numerous scenarios, it encounters formidable challenges when confronted with obscure and hazy conditions, presenting obstacles to its optimal functionality.

One of the primary challenges faced by VLPR systems is adverse weather conditions, including fog, rain, and snow, which can substantially reduce visibility and distort captured images. Haze and fog particles scatter light, leading to diminished contrast and complicating the recognition of license plate characters (Chen *et al.*, 2017). In urban environments, pollution, smog, and airborne particles contribute further to reduced visibility, while natural elements such as dust and debris can accumulate on camera lenses, progressively degrading image quality over time.

Low lighting conditions during dawn, dusk, or nighttime represent another significant challenge. In such scenarios, inadequate illumination can result in increased noise in images, impacting the accuracy of license plate recognition algorithms. Additionally, camera angles and positions play a crucial role, as surveillance cameras may be positioned in ways that capture glares or reflections, hindering the visibility of license plates. Shadows cast by overhead lighting or direct sunlight further complicate the recognition process (Gong *et al.*, 2015).

Hardware limitations also contribute to the challenges faced by VLPR systems. The quality of surveillance cameras and associated hardware can vary, influencing the system's ability to capture clear and detailed images. Outdated or poorly maintained cameras may struggle to

produce the high-resolution images necessary for accurate license plate recognition, exacerbating the impact of adverse weather and environmental conditions.

In the previous research done by Sultan *et al* (2023), their proposed method in license plate recognition yields an 96.02% in recognizing vehicle license plate under normal circumstances, and 88.80% in challenging circumstances.

Hence, it is required to develop a license plate recognition method that can achieve higher recognition rate in these challenges.

1.3 Problem Statement

Due to the detection and extraction of vehicle license plates having been an important steps for the Vehicle License Plate Recognition (VLPR), the issue that caused the license plate fail in detection and recognition must be overcome. The previous methods proposed in detecting vehicle license plates have shown less accuracy and are time consuming. Thus, the combination of object detection and optical character recognition is proposed that can detect the accurate area before performing image cropping for the vehicle license plate recognition during hazy environment.

1.4 Aim

The aim of this project is to propose a vehicle license plate recognition model that can function under both normal and challenging conditions by using object detection and optical character recognition.

1.5 Objective

1. To design a vehicle license plate recognition model using You Only Look Once-Network Architecture Search (YOLO-NAS) object detection.
2. To recognize the vehicle license plate number using the designed model with optical character recognition

1.6 Scope of Research

The scope of research in this study are:

- a) The images used must contain vehicles with readable license plate.
- b) The input images are in JPEG (Joint Photographic Experts Group) Format
- c) The dataset used contain only static image.
- d) The proposed method only applies to vehicles license plates.
- e) The language used in this project is coded Python using Google Collab.
- f) The sample images are dataset from Roboflow.

1.7 Justification

This project aims to improve the vehicle license plate recognition technique based on optical character recognition combined with the usage of YOLO-NAS as object detection model and image pre-processing. This project aspires to contribute to the field of vehicle license plate recognition by providing a technique to improve recognition accuracy.

1.8 Thesis Organization

There are six chapters within this project. Chapter 1 is the introduction to this project, it includes the background of vehicle license plate recognition, the problem background, the problem statement of this research. This chapter also included the aim, the objectives, the scope of research, the justification, and the project timeline of this project.

Chapter 2 discusses the literature review on related works related to vehicle license plate recognition, object detection, image segmentation, and optical character recognition. The information provided in this chapter is based on existing research, studies, surveys, and journals.

Chapter 3 is the methodology of this project, which includes the framework, the techniques, and the benchmarking of this project.

Chapter 4 will discuss the design of this project in detail. The analysis of the proposed method is also included in this chapter.

Chapter 5 will show the result and the evaluation of this project with a summary on the outcome of the testing.

Chapter 6 will summaries all the outcomes of this project. This chapter also includes the conclusion of this project. The contribution of this project and recommendations for future works are discussed at the end of this chapter.

CHAPTER 2

Literature Review

2.1 Introduction

This section reviews the literature found that are related to the objective of this study. It is important as this section provides valuable information on vehicle license plate recognition techniques and the method image processing. In the end of this chapter, a summary of the chapter is given.

2.2 Object Detection

Object detection is an essential computer vision technique that aims to detect and precisely locate objects in digital images or video frames. This fundamental task is of great significance in various domains such as autonomous driving, surveillance systems, robotics, and image/video analysis. By accurately identifying objects, object detection enables advanced applications and facilitates the development of intelligent systems.

Current detection systems utilize classifiers in a novel way to accomplish object detection. These systems employ a classifier specifically designed for the object of interest and evaluate it at different scales and positions across a test image. For example, deformable parts models (DPM) employ a sliding window technique, where the classifier is applied at evenly spaced positions throughout the entire image. This approach allows the detection system to efficiently scan and identify objects by systematically analyzing various regions and sizes within the image (Felzenszwalb, 2010). Figure 12 shows an example of object detection.

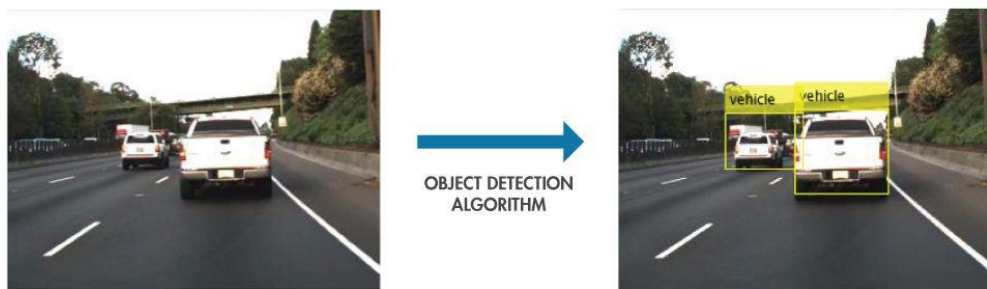


Figure 2.1: Example of Object Detection

(Source: Felzenswalb, 2010)

Recent approaches, such as R-CNN, have introduced regional proposal methods to improve object detection. These methods first generate a set of potential bounding boxes within an image. Subsequently, a classifier is applied to these proposed boxes to determine the presence of objects. Following classification, post-processing techniques are employed to enhance the accuracy of the bounding boxes. This involves refining the bounding boxes, removing duplicate detections, and adjusting the scores of the boxes based on the context and relationships with other objects in the scene. These advanced steps contribute to more precise and reliable object detection results (Girshick, 2014). Figure 13 shows the steps of object detection of CNN.

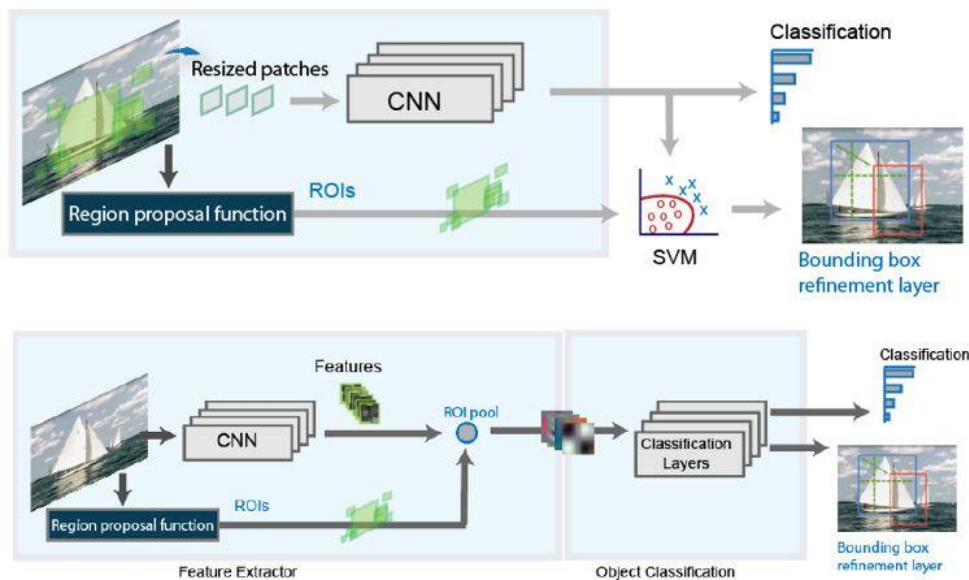


Figure 2.1: CNN Steps of object detection

(Source: Girshick, 2014)

2.3 Vehicle License Plate Detection

Vehicle license plate detection (VLPD) has transcended its initial role in toll collection and parking enforcement, becoming a key component in intelligent transportation systems (ITS), security

surveillance, and even autonomous driving. By accurately identifying and recognizing license plates, systems can automate tasks like traffic flow analysis, stolen vehicle tracking, and access control. The ability to accurately and swiftly recognize license plates is crucial for enhancing the efficiency and effectiveness of these systems (Liu *et al*,2021).

Early VLPD methods relied heavily on image processing techniques. Edge detection algorithms (Zhou *et al.*, 2014), like the Canny edge detection, proved effective for basic plate localization in controlled environments by identifying sharp intensity changes outlining the plate. However, they struggled with background clutter and complex lighting.

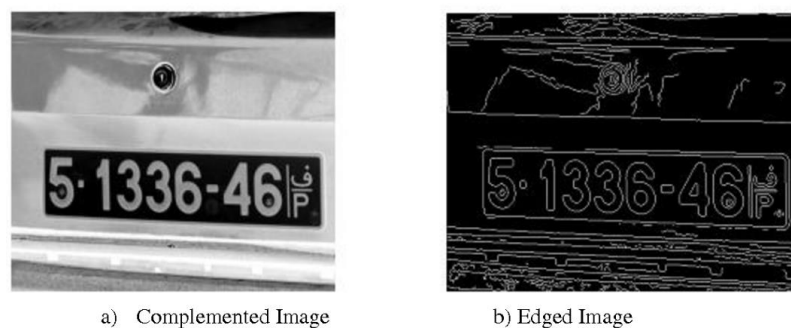


Figure 2.3: Canny edge detection on license plate

Another early technique, template matching, compares image regions to pre-defined templates of license plates, akin to matching puzzle pieces (Huang *et al.*, 2006). Imagine sliding a stencil of a specific license plate format over the image, calculating how well it fits at each location. However, its lack of adaptability to diverse plate formats limited its usefulness.



Figure 2.4: Example of Template Matching

VLPD also embraced the influences of color. Color segmentation techniques capitalize on the distinct hues of license plates, often white or yellow, to isolate them from the surrounding scene (Yin *et al.*, 2009). Imagine transforming the image into a color space where license plates stand out like vibrant sunflowers amidst a field of grass, then applying a threshold to separate them. However, this approach can be easily swayed by lighting variations and background objects with similar colors, like a white van parked behind a car.

2.4 Machine Learning-Based Technique

To overcome these limitations, VLPD embraced the revolution of machine learning. Convolutional Neural Networks (CNNs), the maestros of this movement, learned complex features from images, enabling them to detect license plates with remarkable accuracy and robustness (Ren *et al.*, 2015). Imagine a multi-layered orchestra of neurons, each layer extracting intricate details, culminating in the graceful identification of a license plate even amidst a bustling traffic scene.

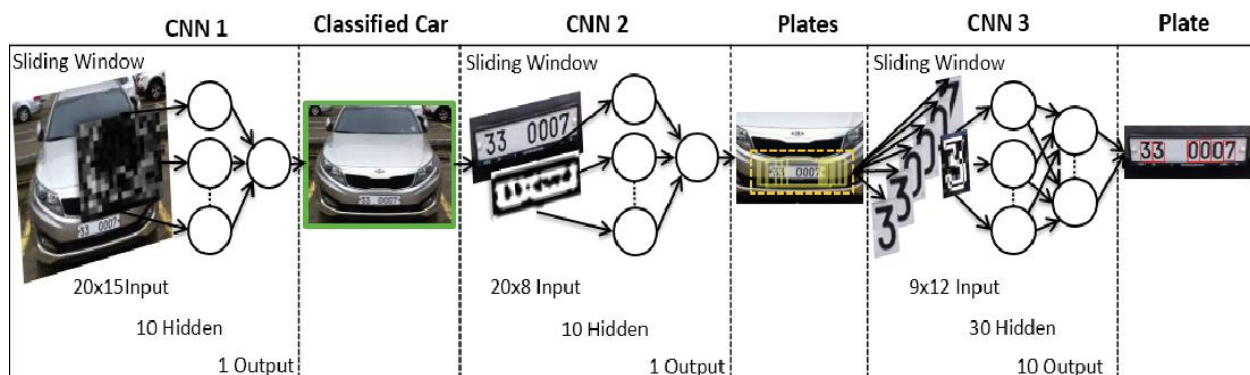


Figure 2.5: Number plate classification based on multiple convolutional neural networks (CNNs)

(source: Gerber & Chung, 2016)

Several CNN architectures have emerged as star performers on the VLPD stage. YOLO, a nimble acrobat, identifies objects in real-time with impressive precision introduced by Redmon and Farhadi (Redmon & Farhadi 2016). Faster R-CNN, a meticulous detective, excels at accurate object localization (Ren *et al.*, 2015). East Text Detection, a specialist in deciphering text, shines in locating license plates even under challenging conditions (Zhou *et al.*, 2017). And, like a skilled

composer weaving diverse melodies, attention mechanisms can be added to these models to focus on critical regions within the image, further enhancing plate detection and recognition.

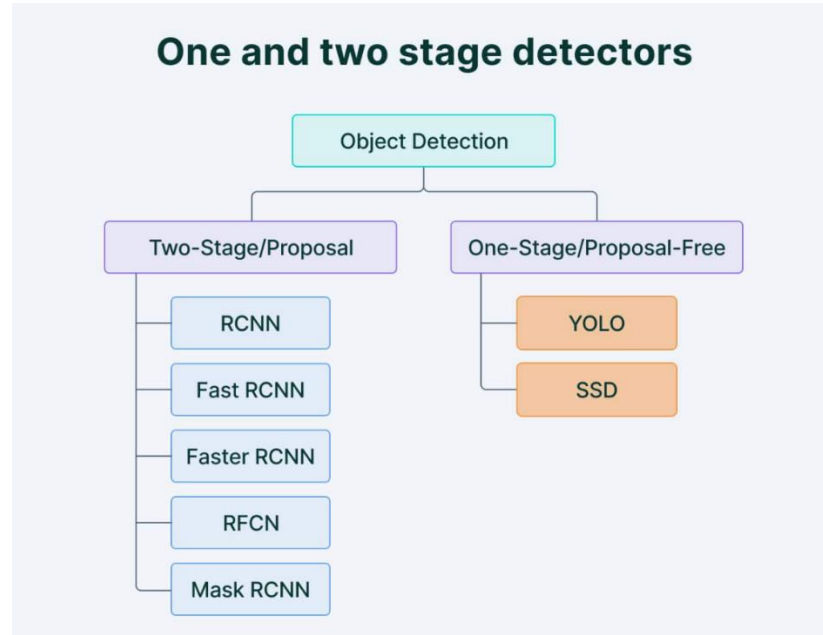


Figure 2.6: One-Stage and Two-Stage Object Detection Model

Recent CNN architectures incorporate attention mechanisms to focus on critical regions within the image, leading to enhanced plate localization and recognition. For instance, Wang *et al.* (2022) proposed an attention guided YOLOv7 model achieving exceptional accuracy in diverse datasets. Real-time applications, particularly in resource-constrained environments, demand lightweight CNN architectures. Liu *et al.* (2021) introduced a compact CNN model for VLPD with remarkable efficiency and accuracy, paving the way for deployment on embedded devices. To address the challenge of plate format variations, domain adaptation techniques are gaining traction. Zhang *et al.* (2023) proposed a novel domain-adaptive CNN able to accurately detect plates across different countries with minimal training data.

Despite its remarkable progress, VLPD faces persistent challenges. Environmental factors like lighting, weather, and camera angles can disrupt even the most sophisticated algorithms. The multitude of plate formats across countries and regions demands adaptable solutions, akin to playing a melody on instruments with different tunings. Moreover, real-time applications,

particularly in autonomous driving, require lightning-fast detection, adding another layer of complexity to the performance.

2.5 You Only Look Once (YOLO)

Real-time object detection has become a crucial element in various applications across fields such as autonomous vehicles, robotics, video surveillance, and augmented reality. Among the many object detection algorithms available, the YOLO (You Only Look Once) framework has gained significant attention due to its exceptional balance between speed and accuracy. It enables rapid and dependable object identification in images. Over time, the YOLO family has undergone multiple iterations, with each version building upon its predecessors to address limitations and enhance performance (refer to Figure 2.6). This paper aims to provide a comprehensive review of the YOLO framework's evolution, starting from the original YOLOv1 and progressing to the most recent YOLOv8. It will discuss key innovations, differences, and improvements introduced in each version.

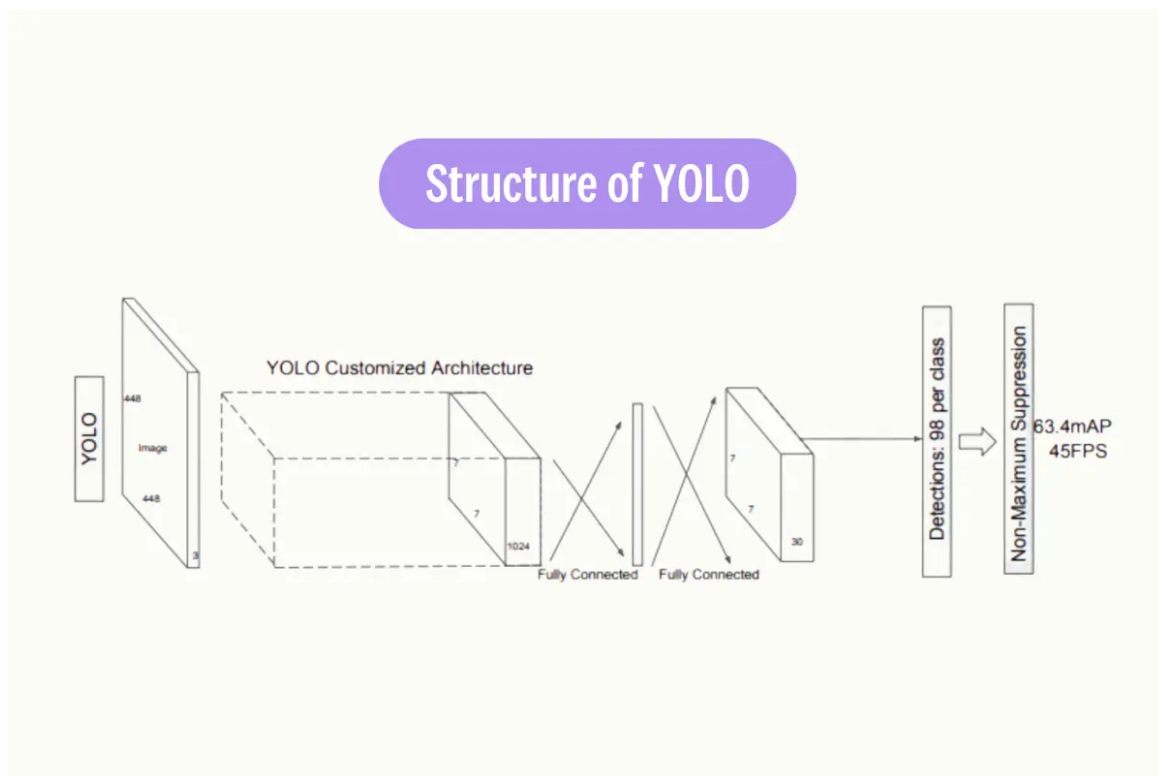


Figure 2.7: Structure of YOLO

(Source: Terven *et al*, 2023)

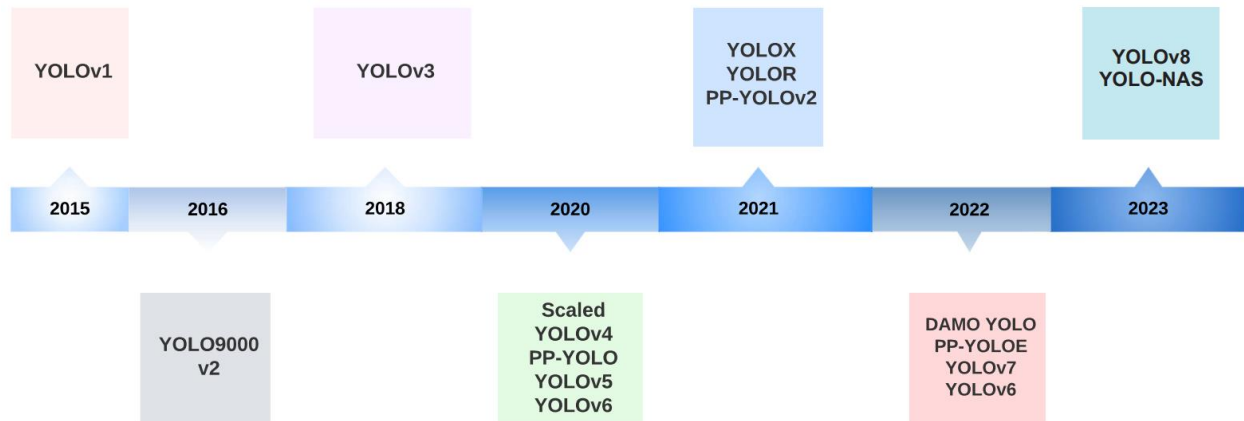


Figure 2.8: The evolution of YOLO

(Source: Terven *et al*, 2023)

The real-time object detection capabilities of YOLO have proven to be highly valuable in autonomous vehicle systems. This technology allows for efficient identification and tracking of diverse objects such as vehicles, pedestrians (Lan *et al*, 2018; Hsu *et al*, 2021), bicycles, and various obstacles (Benjumea *et al*, 2021; Dazlee *et al*, 2022; Liang *et al*, 2022; Li *et al*, 2021). These capabilities have found applications in several domains, including action recognition [7] in video sequences, surveillance (Ashraf *et al*, 2022), sports analysis (Zheng *et al*, 2022), and human-computer interaction (Ma *et al*, 2021). YOLO's ability to perform rapid and accurate object detection has made it a versatile and widely utilized tool across various fields.

YOLO models also applied in the agricultural sector, specifically in the detection and classification of crops (Tian *et al*, 2019; Wu *et al* 2020). These models have been utilized to identify pests and diseases (Lippi *et al*, 2021), aiding in precision agriculture techniques and automating farming processes. Moreover, YOLO models have been adapted for face detection tasks in biometrics, security systems, and facial recognition applications (Yang *et al*, 2018; Chen *et al*, 2021). The versatility of YOLO's object detection capabilities has allowed it to contribute to advancements in various fields, including agriculture and biometrics.

In the medical field, YOLO has been successfully applied for cancer detection (Al-Masni *et al*, 2018; Nie *et al*, 2019), skin segmentation, and pill identification. These applications have led

to enhanced diagnostic accuracy and more efficient treatment processes. Additionally, in remote sensing, YOLO has been utilized for object detection and classification in satellite and aerial imagery. This has proved beneficial in tasks such as land use mapping, urban planning, and environmental monitoring (Qing *et al*, 2021; Zakria *et al*, 2022). YOLO's versatility and accuracy make it a valuable tool in diverse areas of healthcare and remote sensing, facilitating advancements in diagnostics, treatment, and environmental analysis.

Security systems have successfully incorporated YOLO models to enable real-time monitoring and analysis of video feeds. This integration has facilitated swift detection of suspicious activities (Kumar *et al*, 2021), enforcement of social distancing measures, and identification of individuals not wearing face masks. Moreover, YOLO models have been employed in surface inspection applications to detect defects and anomalies, thereby enhancing quality control in manufacturing and production processes (Ukhwah *et al*, 2019; Du *et al*, 2021). The utilization of YOLO models in security systems and surface inspection has improved the efficiency and accuracy of monitoring and quality control procedures, contributing to enhanced safety and productivity.

In traffic applications, YOLO models have played a significant role in tasks such as license plate detection and traffic sign recognition. These applications have contributed to the advancement of intelligent transportation systems and traffic management solutions. YOLO models have also been employed in wildlife detection and monitoring, aiding in the identification of endangered species for biodiversity conservation and ecosystem management (Roy *et al*, 2023). Furthermore, YOLO has been extensively utilized in robotic applications (Kulik *et al*, 2020; Reis *et al*, 2019) and for object detection from drones (Sahin *et al*, 2021; Chen *et al*, 2023). The versatility and robustness of YOLO models have made them valuable tools in a wide range of applications, spanning traffic management, wildlife conservation, robotics, and aerial monitoring. (Tervern *et al* 2023)

2.6 Image Processing

Image processing is a field of signal processing that involves the manipulation and analysis of images. It focuses on the enhancement, interpretation, and extraction of information from visual data. Image processing techniques can be applied to digital or analog images, but in

contemporary contexts, digital image processing is more common due to the prevalence of digital imaging devices. The field of image processing has become fundamental, with applications spanning medical imaging, computer vision, and remote sensing (Gonzalez and Woods, 2008).

Image acquisition and preprocessing constitute the initial steps in image processing. Advances in sensor technology (Pratt, 2007) have enhanced the quality of acquired images. Techniques such as noise reduction and contrast enhancement are crucial preprocessing steps for subsequent analysis (Gonzalez and Woods, 2008).

2.7 Image Pre-processing

Image preprocessing involves a set of techniques used to enhance or prepare digital images for further analysis or interpretation. These techniques aim to improve the quality, accuracy, and effectiveness of subsequent image processing tasks (Gonzalez & Woods, 2008). Image preprocessing encompasses various operations, including normalization, noise reduction, and contrast enhancement. These techniques are applied to raw images to address issues such as uneven lighting, distortions, and artifacts, making the images suitable for subsequent analysis.

The most common image pre-processing are normalization, noise reduction, and contrast enhancement. Normalization involves adjusting pixel values to a standard scale, reducing the impact of variations in lighting conditions across images. Noise reduction techniques, like median filtering or Gaussian blurring, aim to remove unwanted disturbances in the image caused by sensor noise or environmental factors. Contrast enhancement methods, such as histogram equalization, improve the visibility of features in an image by adjusting the distribution of pixel intensities.

2.7.1 Contrast Adjustment

Contrast enhancement is a fundamental image preprocessing technique that focuses on improving the visibility of structures and details in an image by adjusting the distribution of pixel intensities. This process aims to increase the visual separation between different regions of the image, making it easier for subsequent analysis and interpretation. The common contrast enhancement is histogram equalization, contrast stretching, and adaptive histogram equalization.



Figure 2.9: Image Before and After Contrast Adjustment

Histogram equalization is a widely used contrast enhancement technique that redistributes the intensity values of pixels in an image, aiming for a more uniform histogram. It improves the overall brightness and contrast of an image, making it suitable for various computer vision applications (Gonzalez & Woods, 2008). The process of histogram equalization is by stretching the pixel intensity values across the entire dynamic range, it enhances the contrast and makes features more distinguishable.

Contrast stretching, also known as normalization or intensity scaling, involves linearly mapping the original pixel values to a new range to enhance the overall contrast of an image, particularly useful when the original dynamic range is not fully utilized. By expanding or compressing the original intensity range, contrast stretching aims to better utilize the available dynamic range (Pratt, 2007).

Adaptive histogram equalization is an extension of histogram equalization that operates on local regions within an image rather than the entire image. It adapts the enhancement process to different regions, preventing over-amplification of noise and preserving local details and is

especially effective in enhancing contrast in images with varying illumination conditions (Pizer *et al*, 1987).

2.8 Optical Character Recognition (OCR)

Optical Character Recognition (OCR) is a technology that enables the extraction of text from images or scanned documents, allowing the text to be edited, searched, or analyzed. OCR systems utilize machine learning algorithms and computer vision techniques to recognize and interpret characters and words within an image.

An OCR system analyzes the structure of an input document images and segments the image into different components such as text blocks, tables, and images (refer to Figure 15). Within these components, lines are further divided into words and then into individual characters. Once the characters are identified, they are compared against a predefined set of pattern images. The program analyzes and processes all potential matches, ultimately presenting the user with the recognized text. Through this sequence of steps, OCR enables the extraction of text from scanned documents and images, allowing for further manipulation and utilization of the recognized content.

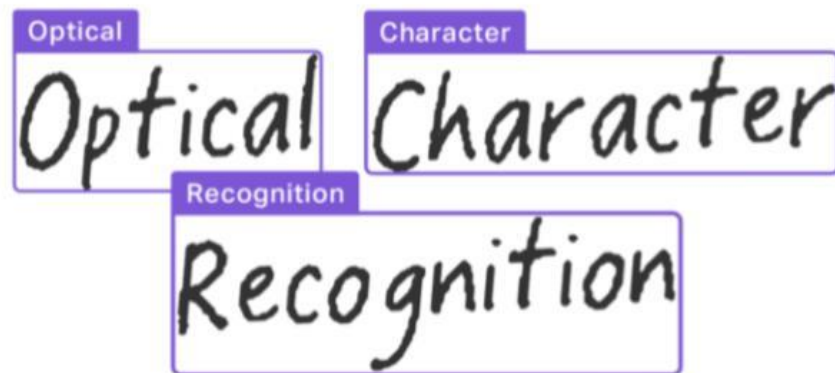


Figure 2.10: Example of OCR

2.9 Vehicle License Plate Recognition

The image segmentation technique is often used to detect vehicle license plates along with the object detection techniques. Automatic vehicle license plate detection is developed to provide a more effective and technical advantage compared to traditional methods as traditional traffic monitoring systems often fail to provide optimal solution when the density of traffic is high (Tian, 2008).

The process of vehicle license plate detection typically involves four fundamental steps: image acquisition, pre-processing, license plate localization, and extraction of the license plate (Chowdhury, 2018). In the first step, image acquisition, a camera or other imaging device captures an image of the vehicle. This image serves as the input for the subsequent steps. Next, in the pre-processing step, various techniques are applied to enhance the quality of the image and improve the visibility of the license plate. This may include operations such as noise reduction, contrast adjustment, and image normalization. The third step, license plate localization, focuses on identifying the region within the image where the license plate is located. This is typically achieved by employing algorithms that detect and analyze specific characteristics of license plates, such as their shape, color, or text patterns. Finally, in the extraction step, the identified license plate region is segmented and separated from the rest of the image. This involves techniques such as image cropping, boundary detection, or image masking to isolate the license plate for further processing or recognition purposes. By following these four steps, the vehicle license plate detection process aims to accurately locate and extract license plate information from images, facilitating applications such as automated vehicle identification, security systems, or traffic management. Figure 16 shows an example of vehicle license plate detection.



Figure 2.11: Example of Vehicle License Plate Detection

(Chowdhury, 2018)

OCR is often used combined with vehicle license plate detection techniques to identify each character and number on the license plate. This allows the character and number recognized to be extracted out for other usage. Figure 17 shows the example of OCR used in license plate recognition.



Figure 2.12: License plate recognition with OCR

(Source: Sultan *et al*, 2023)

2.10 Super Resolution

Super-resolution (SR) imaging, a transformative technology in digital image processing, seeks to enhance image resolution beyond the limitations of the original data. This comprehensive literature review explores the evolution of SR techniques, ranging from classical methods to state-of-the-art deep learning approaches. The discussion encompasses applications, challenges, and future directions in the field. Super-resolution refers to the process of enhancing the resolution and quality of digital images beyond their original constraints. The goal is to generate high-resolution images from low-resolution counterparts, unveiling finer details and improving overall visual fidelity.

SR techniques find applications in diverse fields, including medical imaging (Huang *et al.*, 2017), satellite imaging (Romano *et al.*, 2017), and surveillance systems (Yang *et al.*, 2019). Improved resolution contributes to enhanced diagnostics, detailed mapping, and advanced video surveillance. Single Image Super-Resolution (SISR) Technique deals with the challenge of enhancing a single low-resolution image, have witnessed advancements. Sparse coding-based approaches (Yang *et al.*, 2008) and non-parametric patch-based techniques (Glasner *et al.*, 2009) have contributed to the enhancement of image details.

2.11 Non-Neural Network VLPR

License plate detection is a fundamental component in various applications, from traffic management to law enforcement, where accurate identification of vehicles is paramount. Despite the recent dominance of convolutional neural networks (CNNs) in this field, non-neural network techniques persist as viable solutions in certain scenarios (Kim *et al.*, 2019). Traditional image processing methods such as edge detection, thresholding, and morphological operations are fundamental tools for license plate segmentation. These techniques rely on fundamental principles of image analysis, leveraging gradient information, intensity thresholds, and spatial operations to isolate candidate regions resembling license plates. Feature-based approaches, including Histogram of Oriented Gradients (HOG) and Haar cascades, extract salient features from image regions and employ classifiers to detect patterns resembling license plates. Template matching and contour analysis further contribute to the localization of plate-like structures within the image. However, these techniques encounter challenges, including sensitivity to variations in

lighting conditions, occlusions caused by objects or other vehicles, and diverse appearances of license plates due to different designs and reflective materials.

Additionally, these methods may struggle with complex backgrounds or non-standard plate layouts, necessitating careful parameter tuning and potentially resulting in suboptimal performance. Nonetheless, combinations of these techniques or their integration with machine learning methods like Support Vector Machines (SVM) can improve detection accuracy, especially in scenarios with constrained computational resources or limited training data availability (Zhu *et al.*, 2018).

Despite their enduring relevance, traditional techniques are increasingly being overshadowed by deep learning approaches, which automatically learn discriminative features from data and offer superior performance in various domains. Nonetheless, traditional techniques remain valuable as baseline methods for benchmarking and in situations where computational resources are limited or where neural network approaches may be impractical.

2.12 Evaluating Vehicle License Plate Recognition System

VLPR systems encompass several key components, including image acquisition, preprocessing, license plate localization, character segmentation, and recognition. These systems leverage techniques such as deep learning, image processing, and pattern recognition to achieve accurate and efficient license plate detection and recognition (Singh & Kumar, 2020).

Benchmarking VLPR models is essential for evaluating their performance, accuracy, identifying limitations, and guiding future research directions. Despite challenges, advancements in deep learning and innovative methodologies offer promising opportunities for enhancing the effectiveness and practicality of VLPR technology.

Benchmarking VLPR models involves meticulous selection of datasets, evaluation metrics, and comparison techniques. Commonly used datasets like SVHN and LPRNet offer diverse sets of images captured under various conditions. Evaluation metrics such as accuracy, precision, recall, and F1-score provide quantitative measures of model performance, aiding in rigorous comparative analysis (Zhang & Liu, 2021).

Despite significant advancements, benchmarking VLPR models encounters challenges related to dataset diversity, occlusion, lighting conditions, and environmental variations.

Addressing these challenges is crucial for enhancing the reliability and generalization capabilities of VLPR systems (Chen et al., 2023).

2.13 Summary

The related works to the research title were reviewed and discussed in this chapter which the cites to the author of the research, study paper, or journal. All the discussed works were used to determine the proposed technique and design the methodology of this research.

CHAPTER 3

Methodology

3.1 Introduction

The methodology in developing the proposed method will be stated and explained in this chapter. In this chapter, the research frameworks for this project were discussed in detail. The system framework, experimental setup, benchmarking is also discussed. A summary of this chapter is included at the end of this chapter. Research Framework is the detailed structure of this project. This project was divided into two phases, research phase and implementation phase. The research phase emphasizes preliminary research in determining the background, aim and objectives of this project. It also includes literature review on previous related work and the framework design of this project. Implementation phase is the development of the system, the evaluation of the system, and the conclusion of the project. The system framework is about the working process of the developed system. It discussed each process in detail. The system architecture was categorized into three stages: input, process, and output. Experimental setup mentioned the setup used in developing and evaluating the developed system. The benchmarking discussed how the developed system was evaluated.

3.2 Research Framework

A research framework is a structured plan or outline that guides the process of conducting research. It typically includes the research question or problem statement, the theoretical foundation or conceptual framework, the methodology, and the result. It provides a roadmap for researchers to follow in order to systematically investigate a particular topic or issue. Table 3.1 show the research framework of this project

Table 3.1: Research Framework

Research Phase	
Preliminary Research	<ul style="list-style-type: none">• Research on current problem and issue of the• Aims, objective and scope were identified
Literature Review	<ul style="list-style-type: none">• Study about the related information and previous work related to the• Figure out the initial idea to start the project and focus on techniques that can be used
Framework Design	<ul style="list-style-type: none">• Outline of the framework of this project was started.• Discuss about the chosen techniques in detail
Implementation Phase	
Implementation	<ul style="list-style-type: none">• The selected techniques will be implemented to develop the system.• System testing will be carried out
Evaluation	<ul style="list-style-type: none">• The result of the system will be evaluated and analyzed
Conclusion	<ul style="list-style-type: none">• The conclusion will be made

3.2.1 Research Phase

Preliminary research was conducted in this phase. The current issues of vehicle license plate detection using image segmentation and character recognition are being studied and discussed. The issues were discussed in the problem background, and a clearer statement was stated in the problem statement. The aim of this project is to present a model that can detect and recognize the vehicle license plate during challenging condition such as foggy or muddy license plate. The

initial idea to start the project is figured out in this phase after reviewing all the related literature and reviews. After that, the techniques to solve the current problem are then decided and the outline of the framework is planned.

3.2.2 Implementation Phase

In Implementation Phase, the proposed method is developed, implemented, and tested, and finally the evaluation on the proposed method will be conducted. The technique development and implementation will be the first stage in this phase. After developing the proposed method, the method will then undergo system testing to ensure that the proposed method meets the requirements. Then, the result of the method will be evaluated and analyzed based on the initial objective of the project. Finally, conclusions will be made regarding the result of the project.

3.3 System Framework

The overview of the system framework for the vehicle license plate recognition can be split into three parts, which are Input, Process, and Output. Figure 18 shows the overview of the system framework for vehicle license plate recognition.

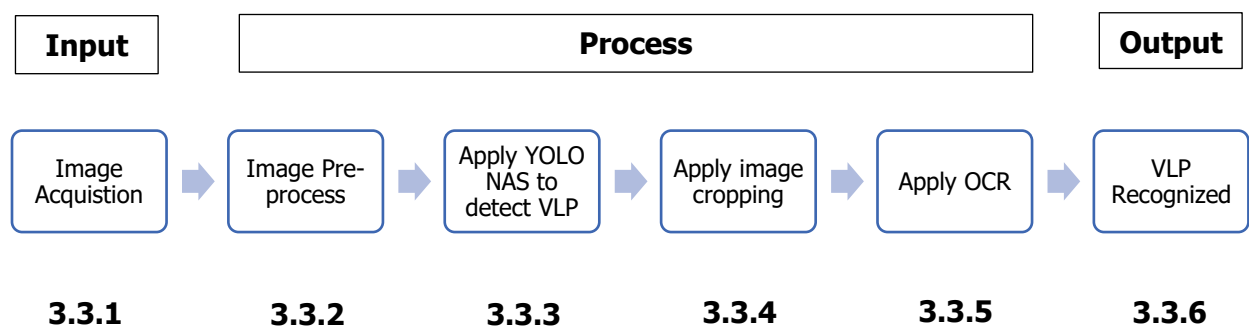


Figure 3.1: System framework

3.3.1 Input

The images will be obtained from Roboflow provided by project-p5nyc as dataset for training. The model will be trained using the 1013 images dataset provided. After the model is developed, the model will be tested on detecting vehicle license plates under normal, obscured, hazy, and raining conditions.

3.3.2 Image Pre-processing

The input image will undergo image pre-processing before the detection process. The pre-processing are super-resolution, contrast adjustment, and image sharpening.

3.3.2.1 Super Resolution

The images will be pre-processed to reduce noises and increase the success rate of vehicle license plate detection. Super-resolution is used to increase the resolution of the image to create a higher quality and crisp image.

Table 3.2: Algorithm of Super-resolution

Algorithm 1: Image super-resolution	
Input	: Image of vehicle
Output	: higher resolution image
<pre>1 Input image of vehicle 2 Apply super-resolution algorithm. 3 from super_resolution import cartoon_upsampling_4x super_img = cartoon_upsampling_4x(img) 4 return enhanced image</pre>	

Low resolution images can be modeled from high resolution images using the below formula, where D is the degradation function, I_y is the high resolution image, I_x is the low resolution image, and σ is the noise.

$$I_x = D(I_y; \sigma) \quad (3.1)$$

The super-resolution used is an super resolution model included in the python library which made by Feng Wang. The algorithm was created using the model proposed by Ledig *et al* (2017) and using deep convolutional neural network. When conducting super-resolution, mean square error (MSE) loss between the higher resolution image and the ground truth image is minimized to ensure the higher resolution image is truth. This is convenient as minimizing MSE will maximize the peak-to-noise-ratio (PSNR) which is used to evaluate the super-resolution algorithm. However, Ledig *et al* (2017) method uses GAN-based network optimized for perceptual loss which is the combination of adversarial loss and content loss.

3.3.2.2 Contrast Adjustment

Table 3.3: Algorithm of Contrast Adjustment

Algorithm 2: Contrast Adjustment	
Input	: Image of vehicle
Output	: higher contrast image
1 Input image of vehicle 2 Apply contrast adjustment. 3 alpha = 0.8 beta = 30 4 Img=cv2.convertScaleAbs(img, alpha=alpha, beta=beta) 5 return enhanced image	

Contrast adjustment of the image is done by the following formula:

$$y = ax + b \quad (3.2)$$

Where x is the input, a is the *alpha* which control the contrast, and b is the *beta* which control the brightness of the image.

3.3.2.3 Image Sharpening

The image will be sharpened using filter2D function in python. The filter2D function applies a convolution mathematical operation to an image. It takes an image, and a small matrix called a kernel. It slides this kernel over the image, performing a calculation at each position. This calculation involves multiplying the values in the kernel with the corresponding pixel values in the image and then summing up these products. Image sharpening can be done by using the correct value in the 3x3 kernel.

Table 3.4: Algorithm of Image Sharpening

Algorithm 3: Image Sharpening	
Input	: Image of vehicle
Output	: Sharpened Image
<pre>1 Input image of vehicle 2 Using kernel = np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]]) 3 img = cv2.filter2D(img, -1, kernel) 4 return sharpened image</pre>	

A kernel is a small matrix used for filtering effects such as blurring, sharpening, edge detection, etc. This is done by doing a convolution between the kernel and the image. The general expression for a convolution is:

$$\begin{aligned}g(x, y) &= \omega * f(x, y) \\ &= \sum_{i=-a}^a \sum_{j=-b}^b \omega(i, j) f(x - i, y - j)\end{aligned}\quad (3.3)$$

Where $g(x, y)$ is the filtered image, $f(x, y)$ is the original image, ω is the filter kernel. In the filter kernel, every elements are considered by $-a \leq i \leq a$ and $-b \leq j \leq b$

3.3.3 VLP Detection using YOLO-NAS

After the image is processed, YOLO-NAS will be applied to detect the area of the vehicle license plate correctly.

Table 3.5: Algorithm of License Plate Detection

Algorithm 4: License plate detection	
Input	: vehicle image
Output:	: detected license plate
<ol style="list-style-type: none">1. Input image2. Declare the pre-trained YOLO-NAS Model3. Declare the prediction using the best prediction model Apply the best prediction model on the input image4. return detected license plate	

The YOLO-NAS model was trained using the dataset before conducting the testing. The YOLO-NAS model was trained using the parameters where the model will determine and detect a license plate and ignore other existing objects.

3.3.4 Extraction of VLP

After the VLP has been detected by YOLO-NAS, the cropping technique is conducted on the original image using the coordinate given by the YOLO-NAS

Table 3.6: Algorithm of Image Cropping

Algorithm 5: Image Cropping	
Input	: Coordinate
Output:	: Cropped Image
<ol style="list-style-type: none">1. Input image2. Input coordinate for the boundary box3. Apply image cropping algorithm4. <code>crop_box = (int(bbox[0]), int(bbox[1]), int(bbox[2]), int(bbox[3]))</code>5. <code>img2 = img_detect[crop_box[1]:crop_box[3], crop_box[0]:crop_box[2]]</code>6. return cropped image	

The boundary box for the cropping algorithm can be obtained from the detect result of the YOLO-NAS using the function: `bboxes = detection.prediction.bboxes_xyxy`. This function provides the values of the xy coordinate of the detection boundary box.

3.3.5 Optical Character Recognition

Now, the image that contains only the vehicle license plate can be used to recognize the license number using the OCR method.

Table 3.7: Algorithm of Optical Character Recognition

Algorithm 6: Optical Character Recognition	
Input	: Cropped Image
Output:	: Recognized Character
<ol style="list-style-type: none">1. Input cropped image2. Declare the Easy OCR algorithm3. Apply the OCR on the image4. return recognized character	

Easy OCR is a user-friendly OCR in python library that simplifies the process of extracting text. The OCR uses edge detection, corner detection, optical character recognition.

The kernel for Sobel operator is:

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \quad (3.4)$$

The formula for corner detection is:

$$R = \det(M) - k \cdot \text{trace}^2(M) \quad (3.5)$$

Where M is the structure tensor of the image patch around the pixel, $\det(M)$ is the determinant of M , $\text{trace}(M)$ is the trace of M , k is an empirically determined constant.

3.3.6 Output

The method can recognize the vehicle license plate number.

3.4 Experiment Setup

There are several preparation that are needed for the experiment setup. Table x shows the preparation needed, the requirement, and the applications.

Table 3.6: Experiment Setup

Preparation	Requirement	Application
Images of vehicles	<ul style="list-style-type: none">• Images that contain Malaysia vehicle• Each images must have visible vehicle license plate	<ul style="list-style-type: none">• Data needed during the implementation phase to test the performance and accuracy of the proposed method
Google Collab	<ul style="list-style-type: none">• Model: Visual Studio Code version 1.75• System Bit: 64-bit	<ul style="list-style-type: none">• Used to program, train and test the proposed method
Data Set for training	<ul style="list-style-type: none">• Images dataset that contains vehicle with visible license plate number• Dataset must contain at least 1000 images	<ul style="list-style-type: none">• Used to develop and train the model.
Data Set for testing	<ul style="list-style-type: none">• Image dataset that contains visible license plate on various occasion	<ul style="list-style-type: none">• Used to test the model.• 50 images for normal• 10 images for obscured, hazy and rain
Performance Testing	<ul style="list-style-type: none">• Test the model in multiple iterations• Run the model under same conditions	<ul style="list-style-type: none">• To test the consistency of the model in running the model

The hardware used to facilitate this project is a 15.6 inch performance laptop with the following technical specifications:

- Processor: Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz
- Graphic Card: NVIDIA GeForce GTX 1650
- RAM: 16 GB, 2933 MHZ
- Operating System: Windows 10 Home, 64-bit
- Harddisk: Lexar 500GB SSD

3.5 Benchmarking

To validate the proposed prototype method, there are a few tests conducted for the chosen parameters.

Table 3.7: Benchmarking

Type of Test	Description	Parameters
Performance Test	<ul style="list-style-type: none"> • The usage is monitored using the monitor tab provided in Google Collab • The performance and usage of the GPU and RAM during the experiment will be recorded. • The execution time of the process will be record. 	<ul style="list-style-type: none"> • GPU usage • RAM Usage • Execution Time
Detection Test	<ul style="list-style-type: none"> • Observe the vehicle license plate area detected by the object detection model 	<ul style="list-style-type: none"> • The vehicle license plate is detected. • The minimum success rate of detection test is 80% (Vig <i>et al</i>, 2023).

Recognition Accuracy Test	<ul style="list-style-type: none"> To test the accuracy of the output whether the vehicle license plate number within the area is recognized accurately 	<ul style="list-style-type: none"> Each character on the license plate is detected. The minimum accuracy of the recognition rate of normal input images is 96% (Sultan <i>et al</i>, 2023). The minimum accuracy of the recognition rate of challenging input images is 88% (Sultan <i>et al</i>, 2023).
---------------------------	--	---

3.6 Summary

This chapter discusses the research framework which included the research phase and implementation phase, and system framework in detail. The experimental setup and benchmarking parameters were also mentioned in this chapter.

CHAPTER 4

System Design and Implementation

4.1 Overview

In this chapter, the design of the optical character recognition based vehicle license plate recognition using YOLO NAS and Watershed is discussed in detail. The overall system design will be explained using the Unified Modelling Language (UML) Diagram. There are three types of UML diagrams that will be shown for this project, which are Use Case Diagram, Sequence Diagram, and Class Diagram. The Use Case Diagram is used to show the outline of the interactions between the user and the proposed model system. The Sequence Diagram is used to show the interaction between the models in a sequential order. The Class Diagram is used to describe the static structure of the system.

4.2 Use Case Diagram

A Use Case Diagram illustrates the scenarios in the systems that interacts with the user effectively. Figure 4.1 shows the used diagram of the vehicle license plate recognition system.

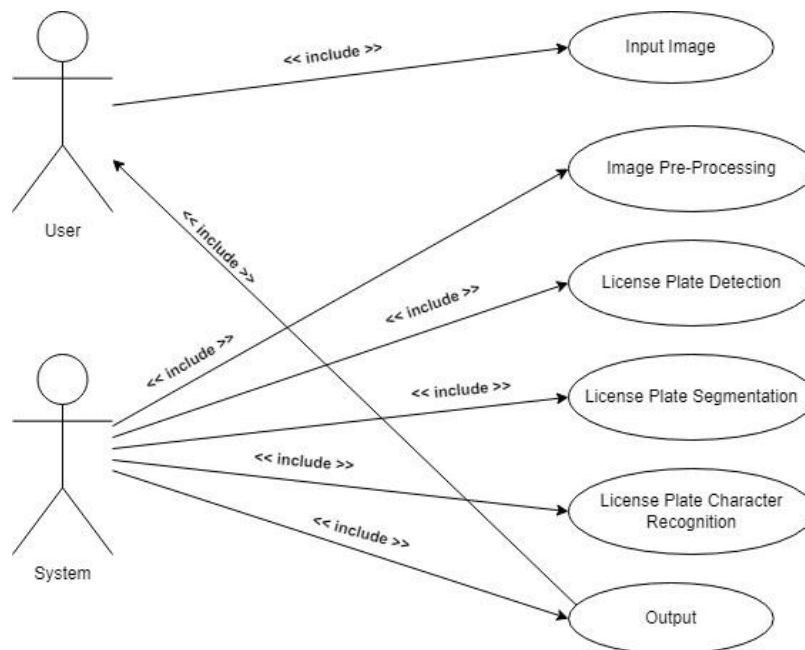


Figure 4.1: Use Case Diagram of the vehicle license plate recognition system.

From Figure 4.1 above, the relationship between the major cases in the system are stated clearly. Users are prompt to input an image into the system. The system then processes the input image using image pre-processing, license plate detection, license plate segmentation, license plate character recognition, and lastly the output after processing the image. The final output result will be shown on the computer screen for the user along with the original input image.

4.3 Class Diagram

Figure 4.2 shows the UML class diagram of the system. There are 12 classes in the system such as Trainer, models, PPYOloELoss.

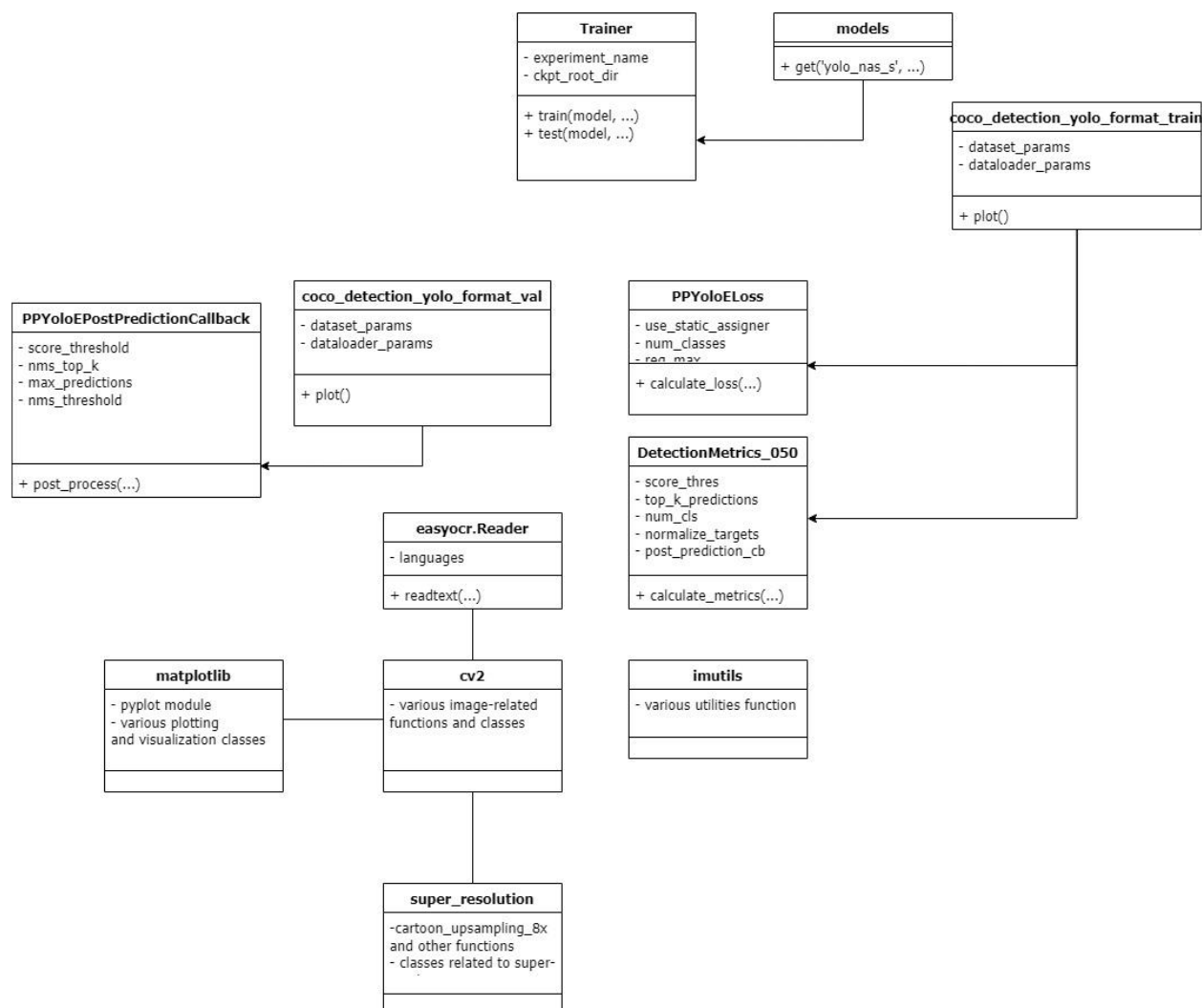


Figure 4.2: Class Diagram

a. Trainer Class

Manages the training and testing processes for machine learning models, specifying experiment details and checkpoint storage.

a. PPYoloELoss Class

- a. Computes the loss for a YOLO-based model during the training process, incorporating aspects like static assigners and regularization.

b. DetectionMetrics_050 Class:

- a. Calculates detection metrics, including score thresholds and top-k predictions, providing insights into model performance.

c. PPYoloEPostPredictionCallback Class

- a. Defines post-prediction processing, including score thresholds and non-maximum suppression, enhancing the precision of YOLO predictions.

d. easyocr.Reader Class

- a. Utilizes the EasyOCR library for Optical Character Recognition (OCR), extracting text from images in various languages.

e. super_resolution Module

- a. Contains functions for super-resolution tasks, such as 8x cartoon upscaling, contributing to image enhancement.

f. matplotlib Module

- a. Utilizes the Matplotlib library for data visualization, enabling the creation of plots and charts for analysis.

g. imutils Module

- a. Incorporates utility functions for basic image processing operations, enhancing the code's readability and efficiency.

h. cv2 Module

Integrates OpenCV functionalities for computer vision tasks, offering a wide range of image processing and manipulation capabilities.

4.4 Sequence Diagram

Figure 4.3 shows the sequence diagram of the license plate recognition system.

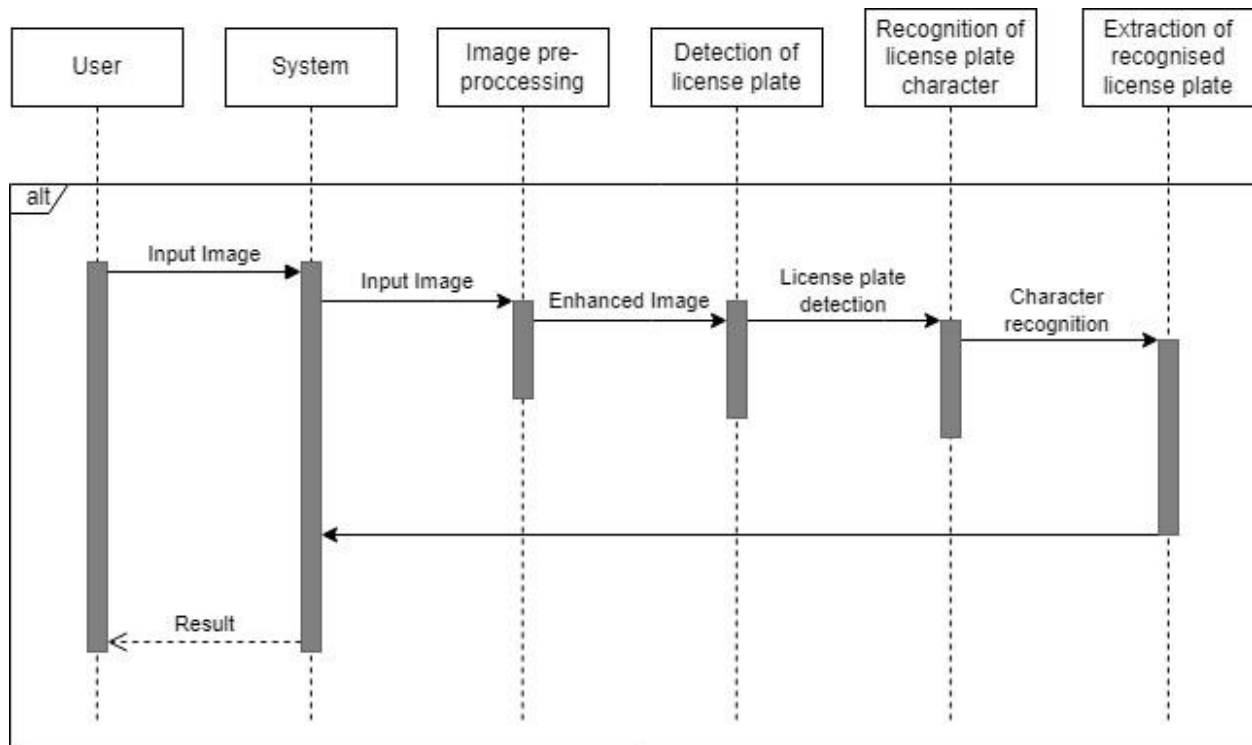


Figure 4.3: Sequence Diagram of license plate recognition system.

Referring to Figure 4.3 above, all the vertical lines represent the interactions in the systems. The process begins with the user input an image into the system for the license plate recognition, the input image is then pre-processed to enhance reduce noise and sharpen the image as well as adjusting the contrast to make it easier for the system to analyze the image. The pre-processed image is then analyzed to detect the presence of any license plate in the image. Once the license plate is detected successfully, the system will proceed to segment out the license plate and discard the rest of the image to reduce any unnecessary information for the character detection process. The segmented license plate image is then analyzed to identify any individual character on the

license plate, and each detected character will be recognized to determine its actual alphanumeric value. Finally, the recognized characters are then combined to form the complete detected license plate number and will be displayed to the user.

4.5 Flowchart

A flowchart is a visual representation of a process or algorithm using various symbols connected by arrows to show the flow of steps. It helps to understand the sequence of actions, decision points, and loops within a system or procedure. Figure 4.4 shows the flowchart of the system.

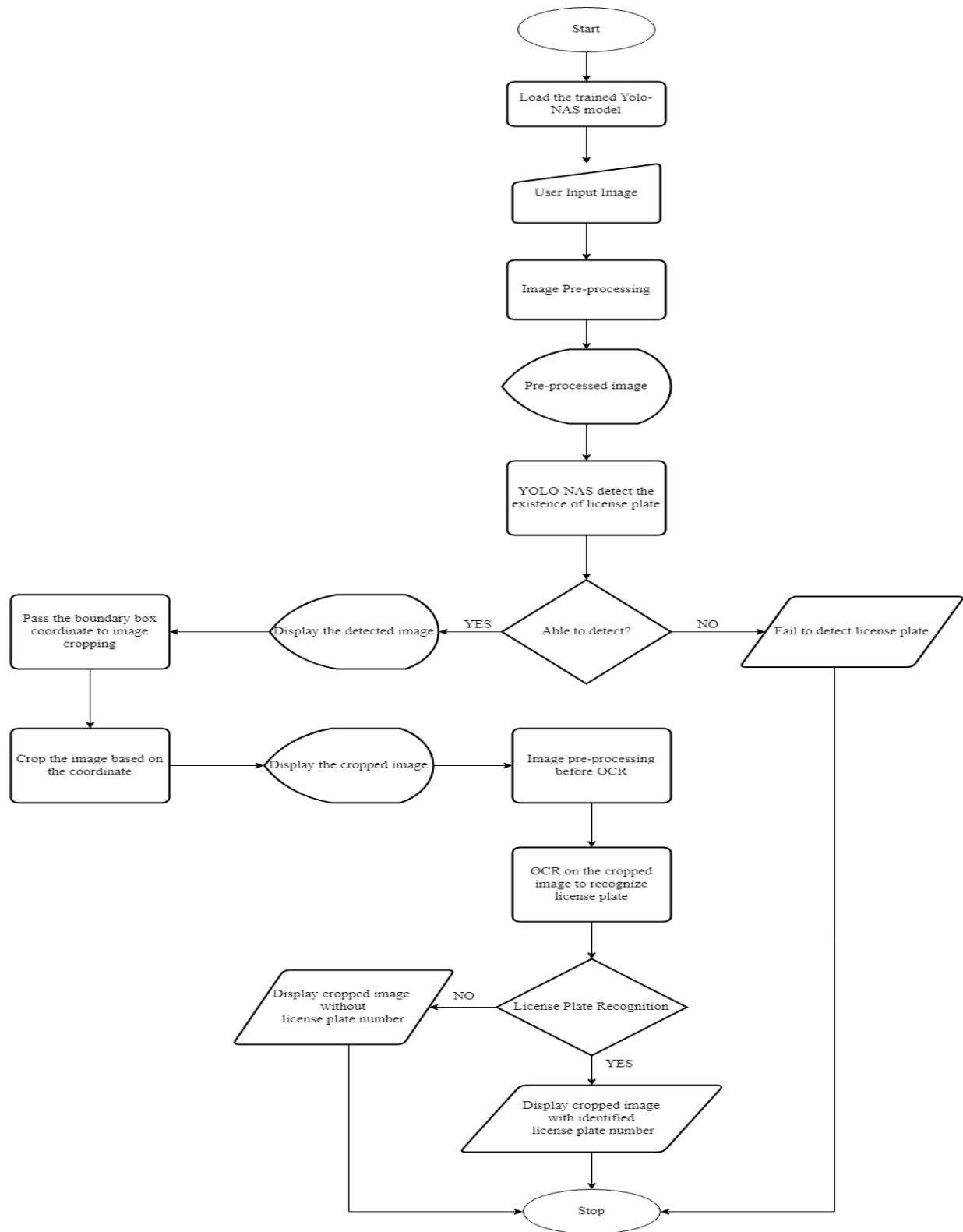


Figure 4.4: Flowchart of the model

The flowchart of the purpose model is as follows:

The user loads the trained YOLO-NAS model by loading its best trained model. Then the user inputs an image into the model for the license plate recognition. The model first apply image pre-process onto the image which contains image enhancing through super-resolution and increasing the contrast of the image. Then the processed image will pass to the YOLO-NAS to detect the license plate within the image. Once the license plate is determined, it will display the image along with the border of the license plate detected by YOLO-NAS and the model will pass the value of the coordinate for the border to the next step.

After the coordinates are determined, the image cropping process starts. Using the coordinates given by the YOLO-NAS, the image containing only the license plate are able to be crop up successfully. Then, the cropped image will go through sharpening process and contrast increasing process for the optical character recognition. Then, EasyOCR will be use on the processed image to determine and recognized any character in the license plate. Then, the model will show the cropped image with the recognized character and the character in text form for the user to compare. The user can compare whether the recognized character are correct by comparing the recognized character with the character on the original license plate in the cropped image

4.6 Algorithm

The license plate recognition model takes an image and pre-process it, detect license plate using YOLO-NAS, cropping the detected license plate using the coordinate determined from YOLO-NAS, enhancing the image before recognizing the license plate, and displaying the recognized license plate. The model is first trained with the dataset obtained, then can be applied to recognize the license plate. Table 4.1 shows the overall flow of the proposed model.

Table 4.1: Algorithm of the overall flow of the proposed model

Algorithm 1 : License Plate Detection Model	
Input	: Vehicle Image
Output	: License Plate Number
	<ol style="list-style-type: none">1. Get input image2. Image pre-processing3. License plate detection4. License plate cropping5. Image enhancing6. License plate recognition7. return license plate number

Table 4.2 shows the algorithm when using the trained license plate detection model with the flows. The input is a vehicle image containing a license plate, and the model will output the license plate number. The license plate detection is using YOLO-NAS that is trained before running this model. Table 4 shows the algorithm when training the YOLO-NAS model.

Table 4.2: Algorithm of the training of the proposed model

Algorithm 2 : YOLO-NAS model training	
Input	: Vehicle dataset
Output	: License plate detection model
	<ol style="list-style-type: none">1. Setup the dataset2. Pass the require parameters3. Initialize the model4. Train the model5. return Best trained model

After the model is trained, the model can then be used in detecting license plate. The testing dataset image will be input into the model to detect the license plate. The input image first needs to pass through pre-processing.

Table 4.3: Algorithm of Image Pre-processing

Algorithm 3	: Image pre-processing
Input	: Testing vehicle dataset
Output	: Pre-processed image
	<ol style="list-style-type: none"> 1. Input image 2. Initialize super-resolution on the image 3. Contrast adjustment 4. Sharpening 5. return pre-processed image

The pre-processing is an important step in the license plate detection process. The image go through super-resolution to increase the resolution of the image to prevent the loss of information, then go through contrast adjustments to overcome some color that might affect the detection process, and lastly sharpness to increase the sharpness of the image.



(a) Original Image



(b) Pre-processed image

Figure 4.5: Comparison of original image and pre-processd image

After the pre-processing process, the image then passes to the license plate detection process. The license plate detection process is done by using the pre-trained YOLO-NAS model to detect the license plate inside the pre-processed image.

Table 4.4: Algorithm of the license plate detection

Algorithm 4 : License plate detection	
Input	: Pre-processed image
Output	: Detected license plate image. License plate boundary box coordinate
<ol style="list-style-type: none">1. Initialize the model2. Pass the require parameters3. Input the pre-processed image4. Initialize license plate detection with best model5. return detected license plate image. license plate boundary box coordinate	



Figure 4.6: License plate detected in the pre-processed image by YOLO-NAS

After the license plate boundary box coordinate is obtained, the model further uses the boundary box to crop out the license plate from the original images. The cropped image will consist of only the area within the boundary box given.

Table 4.5: Algorithm of the image cropping

Algorithm 5	: License plate cropping
Input	: License plate boundary box coordinate Testing vehicle image
Output	: Cropped license plate image
	<ol style="list-style-type: none"> 1. Input the coordinate and the image 2. Initialize cropping algorithm using the coordinate given 3. $bbox = bboxes[0]$ 4. $crop_box = (int(bbox[0]), int(bbox[1]), int(bbox[2]), int(bbox[3]))$ 5. $img2 = img_detect[crop_box[1]:crop_box[3], crop_box[0]:crop_box[2]]$ 6. return cropped image

After the image is cropped, the image will pass to the OCR to recognize the vehicle license plate. This is the last process in the algorithm, where the output will be the recognized vehicle license plate number.

Table 4.6: Algorithm of the OCR

Algorithm 6	: License plate number recognition
Input	: Cropped license plate image
Output	: license plate number
	<ol style="list-style-type: none"> 1. Input the cropped image 2. Initialize the EasyOCR algorithm 3. Characters in the license plate are recognized 4. return license plate number

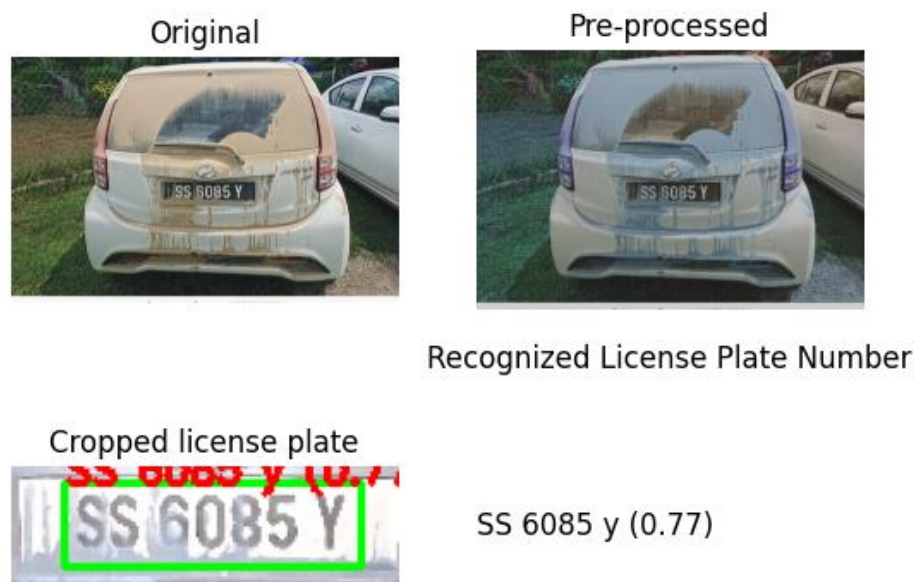


Figure 4.7: License plate number recognized by EasyOCR

Figure 4.7 shows the example of the completed algorithm after the license plate number is recognized by EasyOCR.

4.7 Summary

In this chapter, the overall system flow was illustrated using UML diagrams such as Use Case Diagram, Sequence Diagram and Class Diagram. Besides that, Flowchart has also been used to illustrate the flows of the proposed model for a better understanding of the model. Lastly, the algorithm of every process in the proposed model are also illustrated and explained in detail with figures showing the result of each algorithm.

Chapter 5

Result and Discussion

5.1 Overview

In this chapter, three experiments will be conducted to test and ensure the results are correct and accurate. The experiments that will be conducted are performance test, detection test, and recognition accuracy test. These experiments are conducted to evaluate the system based on the objective.

5.2 Experimental Setup

The experimental set up is used to identify the independent and dependent variables in each test. For each test, the depended variables and the independent variables were listed and prepared. The results from each test were listed in a table form the interpreted. The aim of conducting the three test were to find out whether the proposed method is appropriate for each condition. The dataset prepared for the model are sorted into four dataset: (1) normal dataset which consist of license plate with clear and no occlusion; (2) obscure dataset consisting of license plate with obscure such as dirt and snow; (3) hazy dataset which contain license plate under foggy or hazy conditions; (4) rain dataset which contain license plate in raining situation. The datasets will be conducted in three iterations to obtain the average value of the model for the analysis. The number of images selected for the experiment in 4 different datasets are 50 images in normal dataset, 10 images in obscured dataset, 10 images in hazy dataset, and 10 rain dataset. There are also extra images that are selected in testing the model to test the limitation of the model.

5.2.1 Performance Test

Performance test is the test where the computer usages and the execution time were identified and recorded when executing the model. The performance test focuses on the execution time for the pre-processing process, execution time for the detection process, the total execution time, the RAM usage, and the GPU usage. The average value for the parameters will be calculated

using the result from the 3 iterations, with the exclusion of dataset that the model failed to detect the license plate. The dataset that is failed to detect by the model will still be recorded but is considered invalid and will not be used to calculate the average value.

5.2.2 Detection Test

Detection tests the process where the model is executed and expected to be able to detect the license plate in the dataset. The detection test aims to record down the number of license plates that the model success in identify, which is then used to calculate the percentage of the successful detection. For images that are too obscure or too foggy which can be identified are not taken as dataset but will be used to test the limitations of the model. This is to prevent images that are unrealistic for the model to affect the testing result. The objective for the detection test is to check the percentage of the success license plate detection of the model, which does not take account of character recognizability of the license plate.

5.2.3 Recognition Accuracy Test

The accuracy test is to test the accuracy of the model in recognizing the characters in the license plate. This accuracy test is done by dividing the correct recognized characters by the model with the number of characters on the license plate. The formula is as follows:

$$\frac{\text{Number of correct recognized}}{\text{Number of characters}} \times 100\% = \text{accuracy \%}$$

In the accuracy test, the result of the 3 iterations is recorded and compared with the number of characters to identify the correct recognized characters by the model. The average value of the accuracy percentage is calculated to identify the model satisfied the objective. The accuracy will only depend on the character recognized based on the actual characters on the license plate, any false predicted character that are not a part of the character on the license plate are ignored from the accuracy test.

5.3 Result and Analysis

In this section, the results from the experiment are shown, analyzed, and discussed in each subsection for each test. The data obtained from each test are tabulated for better visualization and understanding of the performance of the model. The data from each iterations and condition are analyzed and evaluated.

5.3.1 Result from Performance Test

In the performance test, the data collected are the execution time for pre-processing, detection, total time, average RAM usage, and average GPU usage. The execution time for pre-processing is high due to the super-resolution process, where the image was enhanced to higher resolution to assist the model in detecting and recognizing the license plate.

Table 5.1: Result of Performance Test on Normal Image Dataset

Dataset	Execution Time (s)									Computer Usage					
	Time taken for Pre-processing			Time taken for detection			Total time			RAM usage (GB)			GPU usage (GB)		
	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Image 1	37.9	37.5	37.6	6.7	7.3	6.4	44.6	44.8	44.0	2.9	3.1	3.2	0.8	0.8	0.8
Image 2	38.5	37.4	37.5	5.9	7.2	6.6	44.4	44.5	44.1	3.3	3.3	3.4	0.6	0.6	0.6
Image 3	37.4	37.9	37.2	5.9	6.6	5.9	43.3	44.6	43.0	3.4	3.5	3.4	1.0	1.0	1.0
Image 4	35.8	34.9	34.4	6.4	6.3	5.7	42.2	41.2	40.1	3.8	3.7	3.7	0.6	0.6	0.6
Image 5	34.8	36.1	35.6	5.2	5.1	4.7	40.0	41.2	40.7	3.4	3.4	3.4	0.7	0.7	0.7
Image 6	36.9	38.1	37.3	6.4	7.0	6.7	43.4	45.1	44.0	3.8	3.5	3.6	0.6	0.6	0.6
Image 7	38.5	37.2	37.4	6.8	6.5	7.5	45.3	43.7	44.9	3.6	3.4	3.4	0.7	0.7	0.7
Image 8	36.6	37.2	36.8	6.3	6.9	7.1	42.9	44.1	43.9	3.5	3.6	3.6	0.6	0.6	0.6
Image 9	37.0	37.1	37.4	8.1	7.7	7.1	45.2	44.8	44.5	3.8	3.7	3.7	0.9	0.9	0.9
Image 10	36.9	36.7	36.5	6.9	6.8	6.8	43.8	43.5	43.7	3.8	3.7	3.7	0.5	0.5	0.5
Image 11	10.8	10.6	12.1	8.7	5.3	6.5	19.5	15.9	18.6	2.6	2.9	2.9	0.6	0.6	0.6
Image 12	45.2	46.1	44.7	7.7	8.7	8.0	52.9	54.8	52.7	3.5	3.5	3.4	0.6	0.5	0.5
Image 13	45.7	45.9	45.1	8.3	8.4	9.1	55.2	54.3	54.2	3.8	3.6	3.6	0.5	0.5	0.5

Image 14	10.5	10.4	10.5	4.6	5.3	5.7	15.2	15.6	15.2	3.6	3.6	3.6	0.5	0.5	0.5
Image 15	12.5	12.3	12.3	4.9	5.5	5.1	17.5	17.8	17.4	3.6	3.6	3.6	0.5	0.5	0.5
Image 16	8.6	8.5	8.7	5.1	5.6	5.1	13.8	14.1	13.8	3.6	3.6	3.6	0.8	0.8	0.8
Image 17	12.5	12.4	12.3	5.0	4.8	4.8	17.5	17.2	17.1	3.6	3.6	3.6	0.6	0.6	0.6
Image 18	12.3	12.3	12.4	4.9	4.7	4.8	17.2	17.1	17.2	3.6	3.6	3.6	0.5	0.5	0.5
Image 19	19.3	19.3	21.3	5.9	5.8	6.2	25.2	25.0	27.5	3.6	3.6	3.8	0.6	0.6	0.6
Image 20	18.8	20.5	19.6	5.7	5.8	6.5	24.5	26.2	26.1	3.8	3.7	3.7	0.5	0.5	0.5
Image 21	8.6	8.0	8.4	4.5	5.3	4.8	13.1	13.3	13.2	3.7	3.7	3.7	0.5	0.5	0.5
Image 22	6.4	7.2	6.2	5.6	4.0	4.8	11.9	11.2	10.9	3.7	3.7	3.7	0.5	0.5	0.5
Image 23	21.9	21.3	21.3	5.5	5.7	5.6	27.5	26.9	26.9	3.7	3.6	3.7	0.6	0.6	0.5
Image 24	9.8	8.9	9.0	4.0	4.1	4.5	13.8	13.0	13.5	3.7	3.7	3.6	0.5	0.5	0.5
Image 25	10.0	9.8	8.5	5.2	4.6	5.1	15.2	14.4	13.6	3.6	3.6	3.6	0.5	0.5	0.5
Image 26	18.6	18.1	17.5	5.7	6.1	5.9	24.3	24.2	23.4	3.7	3.7	3.7	0.6	0.6	0.6
Image 27	11.9	11.9	12.4	4.6	5.0	5.4	16.6	17.0	17.6	3.7	3.7	3.7	0.6	0.6	0.6
Image 28	10.8	11.6	11.4	5.1	4.5	5.4	15.9	16.1	16.8	3.7	3.7	3.7	0.6	0.6	0.6
Image 29	11.9	10.9	10.5	5.2	5.5	5.1	17.0	16.4	15.7	3.7	3.7	3.7	0.5	0.5	0.5
Image 30	8.5	8.6	7.5	5.1	4.7	5.4	13.6	13.2	12.8	3.7	3.7	3.7	0.6	0.6	0.6
Image 31	20.3	18.3	18.8	6.2	6.7	7.1	26.5	25.0	25.9	3.7	3.7	3.8	1.5	1.5	1.5
Image 32	8.9	10.0	8.8	5.2	4.4	4.4	14.2	14.4	13.1	3.7	3.7	3.7	0.6	0.6	0.6

Image 33	7.3	8.2	8.4	5.3	4.4	4.3	12.6	12.6	12.7	3.7	3.7	3.7	0.5	0.6	0.6
Image 34	13.4	12.6	12.5	6.2	5.6	5.2	19.6	18.2	17.7	3.7	3.7	3.5	0.5	0.5	0.5
Image 35	36.9	37.1	35.9	7.8	7.7	7.8	44.8	44.8	43.7	3.6	3.5	3.7	0.5	0.5	0.5
Image 36	8.9	8.5	9.5	4.7	5.7	4.7	13.6	14.2	14.2	3.7	3.6	3.6	0.5	0.5	0.5
Image 37	7.8	7.5	9.1	4.7	4.9	4.4	12.4	12.4	13.5	3.6	3.6	3.6	0.5	0.5	0.5
Image 38	11.1	10.3	10.2	4.9	5.0	4.7	16.1	15.3	14.9	3.6	3.6	3.5	0.5	0.5	0.5
Image 39	90.7	92.4	92.3	15.2	13.8	13.9	105.9	106.2	106.2	3.6	3.6	3.7	0.8	0.8	0.7
Image 40	95.0	92.1	90.4	11.4	11.0	11.0	106.4	103.1	101.4	3.6	3.5	3.7	0.7	0.7	0.7
Image 41	90.4	92.2	92.1	13.8	13.1	13.4	103.6	105.3	105.5	4.7	4.3	4.8	0.4	0.6	0.7
Image 42	90.7	89.8	90.1	12.6	12.3	12.5	103.3	102.1	102.6	3.5	3.4	3.5	0.7	0.8	0.8
Image 43	91.8	92.5	91.8	15.9	15.5	15.9	107.7	108.1	107.1	3.8	3.9	4.0	0.8	0.8	0.8
Image 44	94.3	92.0	94.6	15.3	14.6	14.7	109.6	106.7	109.3	3.5	3.5	3.8	0.9	0.9	0.9
Image 45	93.2	93.6	92.1	12.3	12.8	11.9	105.5	106.4	104.0	3.6	3.6	3.5	0.8	0.8	0.9
Image 46	95.2	94.7	94.8	13.5	14.8	14.2	108.7	109.5	109.0	3.5	3.5	3.5	0.7	0.7	0.7
Image 47	92.8	92.5	91.7	10.5	12.7	10.0	103.3	105.2	102.7	3.6	3.6	3.7	0.7	0.7	0.7
Image 48	95.6	97.2	96.3	11.2	12.5	13.2	106.8	109.7	109.5	3.8	3.8	3.8	0.8	0.8	0.8
Image 49	93.2	96.7	95.5	13.4	15.2	15.3	106.6	111.9	110.8	3.6	3.5	3.6	0.7	0.7	0.7
Image 50	93.8	94.5	94.7	15.1	15.8	14.8	108.9	110.3	109.5	3.7	3.7	3.7	0.7	0.7	0.8

From the table, it is shown that the total time when running the model before image 39 was below 50 seconds while the total time increases starting from image 39. This is due to the involvement of higher resolution images in the dataset. It is noticeable that the pre-processing time is causing the higher amount of total time due to the long time required in super-resolution.

Table 5.2: Result of Performance Test on Obscured Image Dataset

Dataset	Execution Time (s)									Computer Usage					
	Time taken for Pre-processing			Time taken for detection			Total time			RAM usage (GB)			GPU usage (GB)		
	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Image 1	24.1	26.9	37.7	6.6	5.8	4.9	30.7	32.7	42.6	3.8	3.8	3.9	0.6	0.7	0.7
Image 2	40.4	42.3	40.0	4.5	5.0	4.6	44.8	47.3	44.7	3.5	3.1	3.1	0.6	0.6	0.6
Image 3	26.5	23.7	24.6	-	-	-	-	-	-	3.4	3.6	3.6	0.7	0.7	0.7
Image 4	234.3	226.8	240.8	5.5	9.1	5.4	239.8	236.0	246.3	3.4	3.4	3.4	0.7	0.6	0.7
Image 5	55.5	48.6	50.2	4.8	5.5	4.6	60.3	54.1	54.8	3.7	3.1	3.2	0.6	0.6	0.6
Image 6	44.2	41.8	38.7	4.4	4.5	4.5	48.6	46.3	43.1	3.7	3.1	3.2	0.6	0.6	0.6
Image 7	31.6	34.6	30.7	6.3	5.5	5.5	37.9	40.1	36.2	3.7	3.7	3.6	0.6	0.6	0.7
Image 8	29.1	31.2	29.2	5.9	6.0	5.2	35.0	37.2	34.4	3.7	3.4	3.4	0.6	0.7	0.7
Image 9	41.6	45.3	43.6	7.8	9.1	8.8	49.4	54.4	52.4	3.8	3.8	3.8	0.7	0.7	0.7
Image 10	33.9	35.8	36.1	8.2	8.6	9.3	42.1	44.4	45.4	3.9	3.8	3.8	0.6	0.6	0.6

In the obscured image dataset, image 3 was unsuccessful in detecting license plate using the model. Hence, the time taken for detection and total time was non-available. Image 4 was having a much higher time for pre-processing and total time, while other images achieved balanced total time.

Table 5.3: Result of Performance Test on Hazy Image Dataset

Dataset	Execution Time (s)									Computer Usage					
	Time taken for Pre-processing			Time taken for detection			Total time			RAM usage (GB)			GPU usage (GB)		
	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Image 1	62.4	40.6	36.8	5.8	5.8	7.7	68.2	46.4	44.5	3	3	3.1	0.6	0.7	0.7
Image 2	36.8	38.9	40.4	10.7	10.3	10.5	47.5	49.2	50.9	3.5	3.1	3.1	0.6	0.6	0.6

Image 3	37.1	36.6	36.9	6.1	8.1	7.4	43.2	44.6	44.3	3.7	3.8	3.8	0.6	0.6	0.6
Image 4	35.1	33.6	36.4	-	-	-	-	-	-	3.7	3.8	3.8	0.6	0.7	0.6
Image 5	37.3	38.4	38.7	5.0	5.9	4.5	42.3	44.3	43.1	3.8	3.8	3.7	0.6	0.7	0.7
Image 6	39.7	37.4	40.0	4.1	6.0	4.5	43.8	43.4	44.4	3.7	3.7	3.7	0.6	0.7	0.6
Image 7	39.5	39.8	41.2	5.4	4.3	4.9	44.9	44.1	46.1	3.8	3.8	3.8	0.7	0.7	0.7
Image 8	40.2	39.3	38.9	6.2	4.3	4.5	46.4	43.6	43.4	3.6	3.6	3.6	0.7	0.8	0.8
Image 9	38.1	37.2	37.5	5.2	4.8	4.9	43.3	42	42.4	3.7	3.7	3.8	0.8	0.7	0.7
Image 10	37.1	38.8	40.3	4.7	3.9	5.2	41.8	42.7	45.5	3.7	3.7	3.7	0.6	0.7	0.7

Similar to obscured table set, the hazy image dataset also had image that the model failed in detecting the license plate. Overall, the total time for the images averaging between 40 seconds to 50 seconds.

Table 5.4: Result of Performance Test on Rain Image Dataset

Dataset	Execution Time (s)									Computer Usage					
	Time taken for Pre-processing			Time taken for detection			Total time			RAM usage (GB)			GPU usage (GB)		
	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Image 1	37.8	37.0	38.2	8.5	9.1	9.2	46.3	46.1	47.3	3.0	2.9	3.1	0.7	0.7	0.7
Image 2	37.5	37.4	37.3	9.1	8.6	8.5	46.6	45.9	45.9	4.0	3.6	5.0	0.7	0.7	0.7
Image 3	36.7	36.4	37.1	9.4	9.5	9.7	46.1	45.9	46.8	3.4	3.3	3.3	0.7	0.7	0.7
Image 4	35.6	36.7	36.3	9.3	9.3	9.5	44.9	45.8	45.8	3.4	3.5	3.5	0.6	0.6	0.7
Image 5	36.4	37.5	36.7	9.5	8.8	8.7	45.5	46.3	45.5	3.7	3.6	3.5	0.6	0.6	0.6
Image 6	35.2	36.3	37.0	10.0	9.4	9.2	45.2	45.8	46.1	3.7	3.6	3.7	0.6	0.6	0.6
Image 7	38.1	36.2	36.8	9.3	9.7	9.6	47.4	45.9	46.4	3.7	3.7	3.7	0.6	0.6	0.6
Image 8	36.3	37.4	36.8	9.6	9.9	9.4	45.9	47.3	46.2	3.7	3.7	3.7	0.6	0.6	0.6
Image 9	45.8	39.6	39.4	10.5	11.3	10.7	56.2	50.8	50.1	3.7	3.3	3.5	0.7	0.7	0.7
Image 10	39.1	40.2	39.8	8.8	9.2	8.9	47.9	49.4	48.7	3.5	3.5	3.6	0.6	0.8	0.7

In the rain image dataset testing, the detection for this dataset is higher than the detection time of the others dataset. This is due to the raining effect causing the model slowed down is detecting and recognizing the license plate.

Table 5.5: Average Value of Performance Test

Dataset	Average Time (s)			Average Computer Usage	
	Pre-processing	Detection	Total Time	RAM usage (GB)	GPU usage (GB)
Normal	38.3	7.7	45.9	3.8	0.7
Obscured	56.3	5.5	59.4	3.5	0.6
Hazy	39.0	5.4	40.9	3.6	0.7
Rain	37.6	9.4	47.0	3.6	0.7

The highest average pre-processing time is obscured dataset, and the lowest in rain dataset. The average detection time ranking from lowest to highest are hazy dataset at 5.4 seconds, obscured dataset at 5.5 seconds, normal dataset at 7.7 seconds, and rain dataset at 9.4 seconds. In the performance test for the normal image dataset, the data obtained is consistent. The value of RAM usage ranges from 3.5 GB to 3.8 GB, while the GPU usage only consists of 0.6 GB and 0.7 GB.

The average RAM usage for normal dataset is 4.6 GB, while the average for obscured dataset is 3.4 and for hazy dataset is 3.5. The reason behind the high RAM usage in normal dataset is probably due to the higher computer usage in the background during the testing.

In a normal dataset, images 39 to 50 yield a higher preprocessing time, detection time, and total time than the previous images. This is due to the image selected starting from here are images that have higher resolution than the previous, causing the time taken for the model to increase. Figure 5.1 shows image 46 in then normal dataset which has higher resolution than the previous images.



Figure 5.1: Image 46 in the normal dataset

In the obscured dataset, there is an outlier which is image 4 that caused a high pre-processing time of 226.8 seconds to 240.8 seconds. This is potentially due to the super-resolution process for image 4 requiring a higher processing time. The model was unable to detect the license plate within image 3, which was removed from the calculation for the average value.



Figure 5.2: Image 4 in the obscure dataset

Based on Table 5.3, the performance of the model in hazy dataset is consistent. The average pre-processing time is consistent at an average value of 46.6, with the highest pre-processing time of 62.4 seconds that is slightly higher than the average value during the first iteration of image 1. There was also an image that the model unable to detect the license plate which was also removed from the calculation of the average values in Table 5.4.

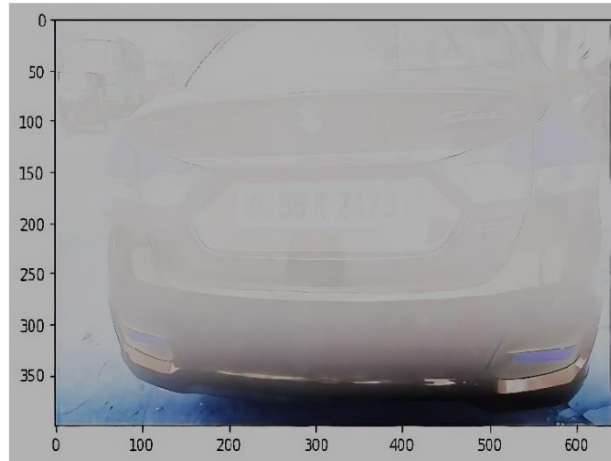


Figure 5.3: Image 3 in the hazy dataset

In the rain dataset, the average detection time is longer than the other dataset. This is caused by the raining effect in the images that caused the model needing the higher time in detecting the license plate. Figure 5.4 shows image 9 in normal dataset which took the longest total time for the model due to the rain effect and the higher contrast of the overall image.

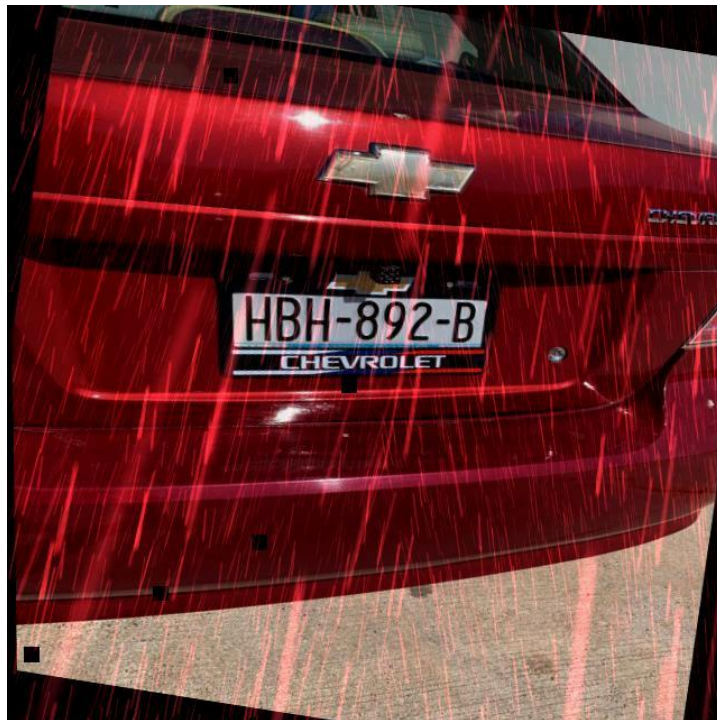


Figure 5.4: Image 9 in rain dataset

Overall, the average detection time and the average computer usage are within the desired range.

5.3.2 Result from Detection Test

The model was used on the 50 normal image dataset selected from the original dataset, and the total number of successful detected license plate was recorded down to calculate the success detection rate of the model. The rain dataset included two extra images to test the limitation of the model.

Table 5.6: Success Rate of Detection Test on Normal Dataset

Number of Image	50
Number of License Plate Detected	50
Success Rate %	100.00

Table 5.7: Success Rate of Detection Test on Obscured Dataset

Number of Image	10
Number of License Plate Detected	9
Success Rate %	90.0

Table 5.8: Success Rate of Detection Test on Hazy Dataset



Number of Image	10
Number of License Plate Detected	8
Success Rate %	80.0




Table 5.9: Success Rate of Detection Test on Rain Dataset


Number of Image	12
Number of License Plate Detected	10
Success Rate%	83.33

For the detection test, the model performed superb for the normal dataset with the accuracy percentage of 100% and performed great in obscure dataset and rain dataset with the accuracy of 90% and 83.33% respectively and performed fine in hazy dataset with 80% detection rate. As mentioned during the performance test, both the obscure dataset and the hazy dataset had 1 testing image that the model failed to detect the license plate which lowered the accuracy of the detection test. The hazy dataset had Image 10 able to detect license plate using the model but unable to recognize the character inside the license plate. The model was also tested on images that had similar or worse conditions than the dataset that the model failed to detect the license plate, which showed limitation of the model. In the testing of the rain dataset, the first 10 images that were originally included in the dataset succeeded in detecting the license plate. However, the two additional images were unable to detect the license plate inside the image.

Table 5.10: Example of Image with worse conditions

Dataset	Image	Description
Obscure Dataset		Image that the model failed to detect during the testing due to the snow and lower resolution picture.
		Image with dirty stain having similar color with the character which caused the model fail to identify the character.

<p>Hazy Dataset</p>		<p>Image that is too foggy/hazy causing the model unable to detect the dataset</p>
		<p>Image with worse condition with the previous which the model unable to identify the license plate</p>
<p>Rain Dataset</p>		<p>Additional image that the model failed in detection the license plate</p>

		Additional image that the model failed in detection the license plate
--	---	---

5.3.3 Result from Recognition Accuracy Test

In the accuracy test, the testing on the model was determined the recognition accuracy of the model on the license plate number detected. Hence, the image in the dataset where the model failed in detecting the license plate was not included in calculating the overall accuracy of the model, the accuracy percentage was calculated solely on the accurate recognized characters in the license plate number.

Table 5.11: Result of Recognition Accuracy Test on Normal Dataset

Dataset	Actual Plate	Recognized Plate	Number of character	Recognized character	Accuracy (%)
1	CAI 4893	CAI 4893	7	7	100.00
2	SAB 1633 C	SAB 1633 C	8	8	100.00
3	WMV 8993	WMV 8993	7	7	100.00
4	MBH 2222	WBH 2222	7	6	85.71
5	HQ 1991	HQ 1991	6	6	100.00
6	VFX 7126	VFX 7126	7	7	100.00
7	PEN 15	PEN 15	5	5	100.00
8	WB 8204 U	WB 8204U	7	7	100.00
9	FD 3171	FD 3171	5	5	100.00
10	HWA 2737	HWA 2737	7	7	100.00
11	W 4575 K	W 4575 K	6	6	100.0

12	VDG 7341	VdG 7341	7	7	100.0
13	ADV8	ADV8	4	4	100.0
14	CZ17 K0D	CZi7 Kod	7	5	71.4
15	MK-35-32	MK-35-32	8	8	100.0
16	DX65 AWD	0X65 AhD	7	5	71.4
17	KL01CA2555	KL01CA2555	10	10	100.0
18	DL 7C N 5617	DL7C N 5617	9	9	100.0
19	LR33 TEE	LR33 TEE	7	7	100.0
20	15-LK-10898	15-LK-10898	11	11	100.0
21	V12 LAF	V12 LAF	6	6	100.0
22	E4 GLE	E4 GLE	5	5	100.0
23	JA62 UAR	JA6Z UAR	7	6	85.7
24	IM4U 555	IMAU 555	7	6	85.7
25	PG MN112	PG MN112	7	7	100.0
26	DAN 54P	DAN 54P	6	6	100.0
27	RX61 GDU	RXG1 GDU	7	6	85.7
28	SDN 7484 U	SDN7484U	8	8	100.0
29	G526 JHD	6526 JHD	7	6	85.7
30	EAB 0001	Bab000]	7	5	71.4
31	PUI8 BES	PuI8 BES	7	7	100.0
32	ALR 486	ALR 486	6	6	100.0
33	DZI7 YXR	DzI7 YXR	7	7	100.0
34	AHY 2407	AHV 2407	7	6	85.7
35	HWC 7890	HIG 7890	7	5	71.4
36	MH 15BD8877	MH 15B08877	10	9	90.0
37	C HI0 0SE	cHIO 0SE	7	5	71.4
38	DL7C N 5617	DL7C N 5617	9	9	100.0
39	A DB4566	A DB4566	7	7	100.0
40	A D11911	011910	7	4	57.1
41	A D20088	a Dzo088	7	5	71.4
42	A D41988	A D41998	7	7	100.0

43	A DC5992	A DC5992	7	7	100.0
44	A DB8617	A DB8617	7	7	100.0
45	A D08815	A D08815	7	6	85.7
46	A D35199	A D35199	7	7	100.0
47	A DB9447	A DB9447	7	7	100.0
48	A D09991	A DD9991	7	6	85.7
49	C D02109	C D02109	7	7	100.0
50	A D12238	A D12238	7	7	100.0
Average					93.22

Table 5.12: Result of Recognition Accuracy Test on Obscured Dataset

Dataset	Actual Plate	Recognized Plate	Number of character	Recognized character	Accuracy (%)
1	VF 21595	VE 21595	7	6	85.71
2	VDU 3228	VDU 3228	7	7	100.00
3	-	-	-	-	0
4	V AD4M	AD4M	5	4	80.00
5	SS 6085 Y	SS 6085 y	7	7	100.00
6	JNA 3526	JNA 3528	7	6	85.71
7	XN 27852	YN 27852	7	6	85.71
8	A E87J2	A E87J2	6	6	100.0
9	A 8P369	A 8P369	6	6	100.0
10	A U5754	A U5754	6	6	100.0
Average					93.0

Table 5.13: Result of Recognition Accuracy Test on Hazy Dataset

Dataset	Actual Plate	Recognized Plate	Number of character	Recognized character	Accuracy (%)
1	21 BH 0001 AA	21 BH 0001AA	10	10	100.00
2	WB 06 F 5977	HB 06 F 5977	9	8	88.89
3	RJ14WC7432	RJ14HC7432	10	9	90.00
4	-	-	-	-	0
5	RJ29CA5938	RJ29Ca5938	10	10	100.00
6	RJ27CD0805	RJ27C00805	10	9	90.00
7	HR26EM1198	HRZ6EM1198	10	9	90.0
8	RJ21CB9889	RJZ7CB9889	10	8	80.0
9	RJ45CE2714	RJ45CE2714	19	19	100.0
10	HP 63B 6147	HP 63B6147	9	9	100.0
Average					93.21

Table 5.14: Result of Recognition Accuracy Test on Rain Dataset

Dataset	Actual Plate	Recognized Plate	Number of character	Recognized character	Accuracy (%)
1	HFX-925-A	HFX-925-A	9	9	100.0
2	HAD-450-E	HAD-450-E	9	9	100.00
3	RDF-977-B	RDF4977-B	9	8	88.9
4	GZM-459-B	GZM-459-B	9	9	100.00
5	A3A-AA Y	A3a Va	7	3	42.9
6	NGM-76-76	NGM-76-76	9	9	100.0
7	HER-889-0	HER 78891g	9	6	66.7
8	HCY-204-E	HCY 204-E	9	8	88.9
9	HBH-892-B	HBH-892-B	9	9	100.0
10	S200 OUK	[200 QU	7	5	71.4

	Average	85.9
--	----------------	-------------

Based on the tables, the accuracy test for the normal dataset, obscured dataset, hazy dataset, and rain dataset achieved accuracy of 93.22%, 93.0%, 93.21%, and 85.9%. Both obscured and hazy dataset had one image excluded in calculating the average accuracy percentage due to the images were unable to detect the license plate by the model. Figure 5.5 shows one of the result after running the model.

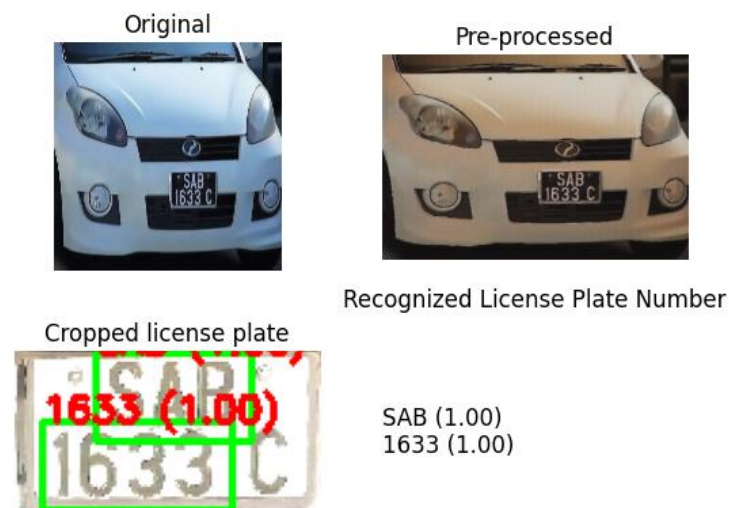


Figure 5.5: Result of Accuracy test of normal dataset

For the characters that are detected with the incorrect letter case, but the correct letter is still considered as a success recognized character. There are some cases where the character detected is not a part of the actual character on the license plate, in which the recognized character based on the character on the license plate are only considered and any others are ignored.

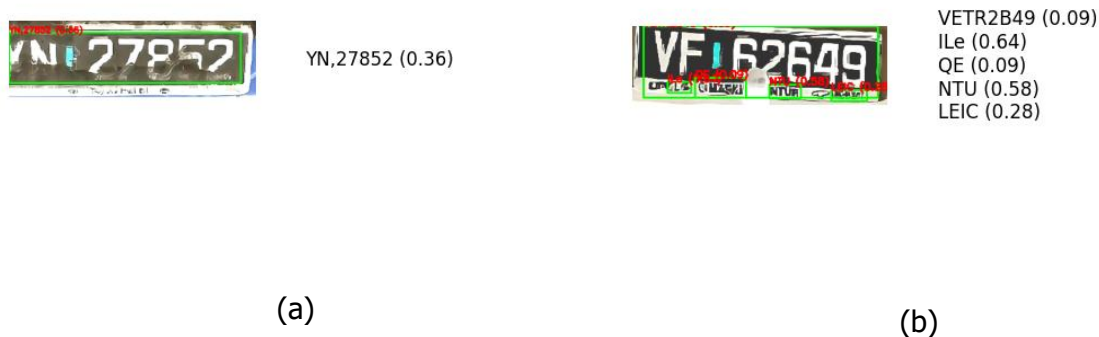


Figure 5.6: (a) Image of non-character recognized as character. (b) Image of several non-related characters been recognized as character

The elements that might look like a character are recognized by the OCR as a character. This is due to the algorithm of how the OCR recognizes a character, which is through detecting the outline of the any elements and identifying whether the outline looks like a character.

5.4 Comparison of proposed model with other methods

The performance of the proposed model is compared to previous models under 3 different circumstances: (1) Success rate in detecting license plate; (2) Accuracy in recognizing license plate under normal circumstances; (3) Accuracy in recognizing license plate under challenging circumstances. The comparison made between existing model and the proposed model is based on the results stated by the researcher, which the dataset used between the models are different.

For the first comparison, the model involved is a method involved YOLO with Siamese neural network (SNN) and Multi-task learning (MTL) to detect non-standard license plate and standard license plate (Vig *et al*, 2023). The comparison is to find out whether the proposed model is on par or better than the existing model.

Table 5.15: The comparison of Success Detection Rate under normal circumstances

Model	Accuracy
YOLO+SNN+MTL	99.2
Proposed model	100.0

Compared to the YOLO+SNN+MTL method, the proposed model had better results. The proposed method had a 100.0 % success rate in detecting license plates under normal circumstances, this is due to the low number of datasets in testing the proposed model where the dataset used had their license plate detected by the proposed model. There is more testing needed on the proposed model to ensure that the proposed model can perform better than existing models.

To compare the accuracy of the recognition under normal circumstances, results from other models were obtained to compare with the proposed model. The first model obtained is a highly trained model that used only YOLOv5 in detecting and recognizing the license plate (Asaad *et. al*, 2022). The model was trained with 58,000 images and tested with 1200 dataset. The model 2 obtained for comparison is a model that combines Faster RCNN, morphological operations and deep learning network (Sultan *et al*, 2023). The model was highly optimized and tested on dataset of more than 100,000 images.

Table 5.17: The comparison of recognition accuracy under normal circumstances

Model	Accuracy (%)
Model 1	98.1
Model 2	99.8
Proposed model	93.22

Comparing the accuracy for license plate recognition under normal circumstances, the proposed model performed poorer than the previous models. This is due to the model was not optimized enough in optical character recognition, although the model succeeds in detecting the license plate, but it was not performing on par with previous models in recognition.

Table 5.18: The comparison of recognition accuracy under challenging circumstances

Model	Accuracy (%)
Model 1	95.95
Model 2	88.80
Proposed model	90.7

For accuracy under challenging circumstances, the accuracy for the proposed model is obtained by calculating the average of the combination of obscure dataset, hazy dataset, and rain dataset. This is due to most researcher having these dataset combined under challenging datasets. The proposed model yields an accuracy of 90.7 %, satisfying the objective of the project and outperforming the model 2.

Overall, the proposed model yields great results after comparing with the existing model. However, the comparison result still needs a better justification by running the proposed model and other existing models on the same dataset.

5.5 Summary

In this chapter, there were three test experimented on the proposed model in recognizing vehicle license plate under obscured conditions. All the results obtained from the experiment were discussed and evaluated. Overall, the results obtained from all three tests are within the benchmark value which showed that the proposed model performance is on-par with previous model.

Chapter 6

Conclusion

6.1 Overview

This chapter contains the conclusion on the proposed method and the results of this projects. This chapter also includes the contribution of this projects toward the current technological knowledge. Besides, the limitations in this projects are also discussed and recommendations for future works are discussed as well.

6.2 Conclusion

As a summary to this project, the Vehicle License Plate Recognition (VLPR) project is conducted to propose a more robust model in detecting and recognizing vehicle license plate under challenging conditions such as occluded license plate or hazy weather conditions. The project was built in Google Collab notebook using python programming language and the libraries in python.

This project is conducted to improve the accuracy of the vehicle license plate recognition using the combination of existing algorithms with the state-of-the-art detection model. There were two objective in this project, and both were achieved throughout the project. The first objective was to design a vehicle license plate recognition model using object detection model and optical character recognition model. The newly developed state-of-the-art detection model, YOLO-NAS, was used as the object detection model and EasyOCR was used as the optical character recognition model in this project. The model was successfully developed, and the results obtained after running the experiment satisfied the requirement. Based on the result, the model was performing consider slightly better than the existing model mentioned in the project for the comparison. The proposed model was able to detect every vehicle image with obvious license plate and perform well in detecting license plate in dataset that contain challenging conditions.

The second objective of this project is to recognize the vehicle license plate accurately during challenging conditions. The model was used to recognize the vehicle license plate under challenging conditions to determine whether the model was able to recognize the character and number in the license plate correctly. The challenging conditions vehicle images contain occlusion which may cause the model to recognize false information or even unable to recognize the license

plate. Overall, this objective was achieved as the performance of the model under such conditions was within the targets.

In conclusion, the vehicle license plate recognition model proposed was performing on-par with the existing method and performed better in certain conditions.

6.3 Contribution

There are two contributions to society in this project. The contributions are:

1. To design a vehicle license plate recognition model using YOLO-NAS object detection.

A VLPR model was designed and developed using YOLO-NAS along with image preprocessing. A detection test was carried out on the model to test the success detection rate of the model on vehicle license plate image under normal circumstances, obscured circumstances, hazy circumstances, and raining circumstances. The proposed model had achieved a 100% success vehicle license plate detection rate under normal circumstances, 90% under obscured circumstances, 80% under hazy circumstances and 83% under rainy circumstances. This will help the other researchers in identifying the better techniques in creating VLPR techniques that can achieve 100% successful rate and accuracy license plate detection in any circumstances.

2. To recognize the vehicle license plate number using the designed model with optical character recognition.

The vehicle license plate number was identified and determined using the model with the help of optical character recognition. The recognition accuracy test was conducted on the designed model to test the accuracy of the recognition of the model in the four circumstances. The designed model was able to recognize vehicle license plate numbers under normal circumstances at 93.22% accuracy, 93.00% accuracy under obscured circumstances, 93.21 under hazy circumstances, and 85.90% under raining circumstances. This will allow the recognizing license plate numbers in image in a large number achieve a higher efficiency using the model.

6.4 Limitations

In this project, there were several limitations that caused the result from the project to be not accurate when comparing to other existing model.

1. Limitation in dataset for training.

There were limitations in training the model due to inability to access the larger and more frequently used dataset.

2. Limited computer power in training.

The computers used in training the model were not powerful enough to run machine learning on a larger dataset. The model may achieve better results if the model was trained with more dataset.

3. Different dataset in comparison

The proposed model was used to compare with the existing model to determine whether the proposed model was on par or better than the existing model. However the dataset used for the proposed model and the existing model are different, this is due to the difficulties in obtaining the same dataset among the researchers. Comparing the models under the same dataset should yield a different result.

6.5 Future work

The proposed model has room for improvement that can result in a more robust technique in recognizing vehicle license plates in challenging conditions. This opens avenues for researchers to further develop the proposed model to achieve a better outcome.

1. Larger dataset in training

The success rate in detecting a license plate is based on how well the model was trained. Hence, it is recommended to use a large dataset in training the YOLO-NAS to increase the stability of the detection.

2. Better image pre-processing techniques

The image pre-processing process is an important factor that contributes to the success chance in detecting and recognizing the license plate. Developing better pre-processing techniques will increase the performance of the model.

3. Large dataset for testing

Larger datasets in testing the model to obtain a stable result of the model.

4. Different object detection model and optical character recognition model

Implementing different models that are on-par or better to build a better VLPR model.

5. Able to use it for moving image/video.

The proposed model can be further developed and improved to be useable in using on moving images such as .gif, or video that contain a moving vehicle.

6.6 Summary

Although the results were not certain and there were still many rooms for improvement in the proposed model, this project however provided the techniques to improve the vehicle license plate recognition system by implementing newer object detection model and optical character recognition model. Moreover, the proposed model had achieved the targets for this projects. More research should be done in the future to further improve the vehicle license plate recognition system.

REFERENCES

- Aleqabie, Hiba J.. (2012). A Hybrid approach for Image Segmentation.
- Al-Masni, M. A., Al-Antari, M. A., Park, J.-M., Gi, G., Kim, T.-Y., Rivera, P., Valarezo, E., Choi, M.-T., Han, S.-M., & Kim, T.-S. (2018). Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system. *Computer Methods and Programs in Biomedicine*, 157, 85–94.
- Asaad, A. (2022). MALAYSIAN AUTOMATIC LICENSE PLATE RECOGNITION USING SINGLE-SHOT OBJECT DETECTION MODEL AT LOW VISIBILITY AND UNCONSTRAINED ENVIRONMENT. <https://perintis.org.my/ejournalperintis/index.php/PeJ/article/view/130>
- Ashraf, A. H., Imran, M., Qahtani, A. M., Alsufyani, A., Almutiry, O., Mahmood, A., Attique, M., & Habib, M. (2022). Weapons detection for security and video surveillance using CNN and YOLO-V5s. *CMC-Comput. Mater. Contin*, 70, 2761–2775.
- Benjumea, A., Teeti, I., Cuzzolin, F., & Bradley, A. (2021). Yolo-z: Improving small object detection in yolov5 for autonomous vehicles. arXiv preprint arXiv:2112.11798.
- Borman, S., & Stevenson, R. L. (1998). Super-Resolution From Image Sequences - A Review. *Proceedings of the 1998 International Conference on Image Processing*.
- Buades, A., Coll, B., & Morel, J. M. (2005). A Review of Image Denoising Algorithms, with a New One. *Multiscale Modeling & Simulation*.
- Chen, C., Zheng, Z., Xu, T., Guo, S., Feng, S., Yao, W., & Lan, Y. (2023). YOLO-based UAV technology: A review of the research and its applications. *Drones*, 7(3), 190.
- Chen, J., et al. (2023). Benchmarking Vehicle License Plate Recognition Models under Challenging Conditions. *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2023, 189-196.
- Chen, W., Huang, H., Peng, S., Zhou, C., & Zhang, C. (2021). YOLO-face: A real-time face detector. *The Visual Computer*, 37, 805–813.
- Chen, X., Lu, Y., & Nishihara, A. (2017). Vehicle license plate recognition with novel templates in unconstrained scenarios. *Pattern Recognition Letters*, 94, 9-17.

- Chowdhury, Wasif & Khan, Ashikur & Uddin, Jia. (2018). Vehicle License Plate Detection Using Image Segmentation and Morphological Image Processing. 142-154. 10.1007/978-3-319-67934-1_13.
- Cigla, C., & Alatan, A. A. (2010). Efficient graph-based image segmentation via speeded-up turbo pixels. <https://doi.org/10.1109/icip.2010.5653963>
- Dazlee, N. M. A. A., Khalil, S. A., Abdul-Rahman, S., & Mutalib, S. (2022). Object detection for autonomous vehicles with sensor-based technology using YOLO. *International Journal of Intelligent Systems and Applications in Engineering*, 10(1), 129–134.
- Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a Deep Convolutional Network for Image Super-Resolution. *European Conference on Computer Vision (ECCV)*.
- Dos Reis, D. H., Welfer, D., De Souza Leite Cuadros, M. A., & Gamarra, D. F. T. (2019). Mobile robot navigation using an object recognition software with RGBD images and the YOLO algorithm. *Applied Artificial Intelligence*, 33(14), 1290–1305.
- Du, Y., Pan, N., Xu, Z., Deng, F., Shen, Y., & Kang, H. (2021). Pavement distress detection and classification based on YOLO network. *International Journal of Pavement Engineering*, 22(13), 1659–1672.
- Gerber, C., & Chung, M. (2016). Number Plate Detection with a Multi-Convolutional Neural Network Approach with Optical Character Recognition for Mobile Devices. *J. Inf. Process. Syst.*, 12, 100-108.
- D. Glasner, S. Bagon and M. Irani, "Super-resolution from a single image," 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 2009, pp. 349-356, doi: 10.1109/ICCV.2009.5459271.
- Gong, Y., Zhang, D., & Yang, J. (2015). Robust license plate recognition from blurred images. *Pattern Recognition*, 48(10), 3131-3142.
- Gonzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing* (3rd ed.). Pearson Education.
- He, Kun & Wang, Dan & Tong, Miao & Zhu, Zhijuan. (2020). An Improved GrabCut on Multiscale Features. *Pattern Recognition*. 103. 107292. 10.1016/j.patcog.2020.107292.

- Hsu, Wei-Yen & Lin, Wen-Yen. (2021). Adaptive Fusion of Multi-Scale YOLO for Pedestrian Detection. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3102600.
- Huang, W., He, Y., & Qiao, Y. (2006). License plate location and recognition in vehicle images. IEEE Transactions on Intelligent Transportation Systems, 7(3), 319-329.
- Huang, Y., *et al.* (2017). Diagnosis of Papillary Thyroid Carcinoma in Ultrasonography Images Using Deep Learning. *Physics in Medicine & Biology*.
- Jain, A., Gupta, P., & Gohokar, A. (2020). Recent Trends in Vehicle License Plate Recognition: A Comprehensive Review. *Journal of Artificial Intelligence and Systems*, 1(2), 123-136.
- Jang, won-Dong & Kim, Chang-Su. (2019). Interactive Image Segmentation via Backpropagating Refinement Scheme. 5292-5301. 10.1109/CVPR.2019.00544.
- Khalifa, Othman & Khan, Sheroz & Islam, Md & Suleiman, Ahmad. (2007). Malaysian Vehicle License Plate Recognition. Int. Arab J. Inf. Technol.. 4. 359-364.
- Kim, J., Kwon Lee, J., & Mu Lee, K. (2016). Accurate Image Super-Resolution Using Very Deep Convolutional Networks. IEEE *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Kim, J., & Jung, K. (2019). License Plate Detection and Recognition: A Dataset and Baseline Algorithm. arXiv preprint arXiv:1906.08210
- Kulik, S., & Shtanko, A. (2020). Experiments with neural net object detection system YOLO on small training datasets for intelligent robotics. *In Advanced Technologies in Robotics and Intelligent Systems: Proceedings of ITR 2019* (pp. 157–162). Springer.
- Kumar, P., Swamy, S. N., Kumar, P., Purohit, G., & Raju, K. S. (2021). Real-time, YOLO-based intelligent surveillance and monitoring system using Jetson TX2. *In Data Analytics and Management: Proceedings of ICDAM* (pp. 461–471). Springer.
- Lan, Wenbo & Dang, Jianwu & Wang, Yangping & Wang, Song. (2018). Pedestrian Detection Based on YOLO Network Model. 1547-1551. 10.1109/ICMA.2018.8484698.
- Ledig, C., *et al.* (2017). Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. IEEE *Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Lee, S. H., Moon, J., & Lee, M. (2006). A Region of Interest Based Image Segmentation Method using a Biologically Motivated Selective Attention Model. In The 2006 IEEE International Joint Conference on Neural Network Proceedings.
<https://doi.org/10.1109/ijcnn.2006.246859>
- Li, Q., Ding, X., Wang, X., Chen, L., Son, J., & Song, J.-Y. (2021). Detection and identification of moving objects at busy traffic road based on YOLO v4. *The Journal of the Institute of Internet, Broadcasting and Communication*, 21(1), 141–148.
- Liang, S., Wu, H., Zhen, L., Hua, Q., Garg, S., Kaddoum, G., Hassan, M. M., & Yu, K. (2022). Edge YOLO: Real-time intelligent object detection system based on edge-cloud cooperation in autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(12), 25345–25360.
- Lippi, M., Bonucci, N., Carpio, R. F., Contarini, M., Speranza, S., & Gasparri, A. (2021). A YOLO-based pest detection system for precision agriculture. In *2021 29th Mediterranean Conference on Control and Automation (MED)* (pp. 342–347). IEEE.
- Liu, J., Liu, W., Ma, Y., & Yu, J. (2021). A lightweight and fast vehicle license plate detection model for real-time
- Ma, H., Celik, T., & Li, H. (2021). Fer-YOLO: Detection and classification based on facial expressions. In *Image and Graphics: 11th International Conference, ICIG 2021, Haikou, China, August 6–8, 2021, Proceedings, Part I 11* (pp. 28–39). Springer.
- Nie, Y., Sommella, P., O’Nils, M., Liguori, C., & Lundgren, J. (2019). Automatic detection of melanoma with YOLO deep convolutional neural networks. In 2019 E-Health and Bioengineering Conference (EHB) (pp. 1–4). IEEE.
- Nurhadiyatna, A., & Lončarić, S. (2017). Semantic image segmentation for pedestrian detection.
<https://doi.org/10.1109/ispa.2017.8073587>
- P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010
- Pizer, S. M., et al. (1987). *Adaptive Histogram Equalization and Its Variations*. Computer Vision, Graphics, and Image Processing, 39(3), 355–368.

- Pramanik, S., Chandra, M. G., Sridhar, C. V., Kulkarni, A. D., Sahoo, P. R., Chethan, D. V. V., Sharma, H., Paliwal, A., Navelkar, V., Poojary, S., Shah, P., & Nambiar, M. (2021). A Quantum-Classical Hybrid Method for Image Classification and Segmentation. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2109.14431>
- Pratt, W. K. (2007). *Digital Image Processing* (4th ed.). Wiley.
- PunamThakare,. (2011). A Study of Image Segmentation and Edge Detection Techniques. International Journal on Computer Science and Engineering. 3.
- Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O. R., & Jagersand, M. (2020). U2-Net: Going deeper with nested U-structure for salient object detection. Pattern Recognition, 106, 107404. <https://doi.org/10.1016/j.patcog.2020.107404>
- Qing, Y., Liu, W., Feng, L., & Gao, W. (2021). Improved YOLO network for free-angle remote sensing target detection. Remote Sensing, 13(11), 2171.
- R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014
- Redmon, J., & Farhadi, A. (2016). YOLO9000: Real-time object detection for image recognition. arXiv preprint arXiv:1612.08242.
- Rehman, M., Ali, M., Obayya, M., Asghar, J., Hussain, L., Nour, M., Negm, N., & Hilal, A. M. (2022). Machine learning based skin lesion segmentation method with novel borders and hair removal techniques. PLOS ONE, 17(11), e0275781. <https://doi.org/10.1371/journal.pone.0275781>
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with high accuracy. arXiv preprint arXiv:1506.02095.
- Romano, Y., *et al.* (2017). Super-Resolution Reconstruction of Satellite Images. *IEEE Transactions on Computational Imaging*.
- Rother, Carsten & Kolmogorov, Vladimir & Blake, Andrew. (2004). GrabCut: Interactive Foreground Extraction Using Iterated Graph Cuts. ACM Trans. Graph.. 23. 309-314. 10.1145/1186562.1015720.

- Roy, A. M., Bhaduri, J., Kumar, T., & Raj, K. (2023). Wildect-YOLO: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection. *Ecological Informatics*, 75, 101919.
- Saeed, S., Abdullah, A., Zaman, N., Naqvi, M., Masud, M., & AlZain, M. A. (2022). Hybrid GrabCut Hidden Markov Model for Segmentation. *Computers, Materials & Continua*, 72(1), 851–869. <https://doi.org/10.32604/cmc.2022.024085>
- Sahin, O., & Ozer, S. (2021). YOLOdrone: Improved YOLO architecture for object detection in drone images. In *2021 44th International Conference on Telecommunications and Signal Processing (TSP)* (pp. 361–365). IEEE.
- Samantaray, Milan & Biswal, Anil Kumar & Singh, Debabrata & Samanta, Debabrata & Karuppiah, Marimuthu & Joseph, Niju. (2022). Optical Character Recognition (OCR) based Vehicle's License Plate Recognition System Using Python and OpenCV. 10.1109/ICECA52323.2021.9676015.
- Shinde, S., Kothari, A., & Gupta, V. (2018). YOLO based human action recognition and localization. *Procedia computer science*, 133, 831–838.
- Singh, A., & Kumar, P. (2020). Benchmarking of Vehicle License Plate Detection and Recognition Systems: A Review. *International Journal of Engineering Research & Technology*, 9(1), 255-261.
- Sultan, F.; Khan, K.; Shah, Y.A.; Shahzad, M.; Khan, U.; Mahmood, Z. Towards Automatic License Plate Recognition in Challenging Conditions. *Appl. Sci.* 2023, 13, 3956. <https://doi.org/10.3390/app13063956>
- Sun, J., Ban, X., Han, B., Yang, X., & Yao, C. (2022). Interactive Image Segmentation Based on Feature-Aware Attention. *Symmetry*, 14(11), 2396. <https://doi.org/10.3390/sym14112396>
- Sun, Yi & YUAN, Peisen & SUN, Yuming & Zhai, Zhaoyu. (2020). Hybrid Segmentation Algorithm for Medical Image Segmentation Based On Generating Adversarial Networks, Mutual Information And Multi-scale Information. *IEEE Access*. 8. 1-1. 10.1109/ACCESS.2020.3005384.

- Tappen, M. F., Freeman, W. T., & Adelson, E. H. (2005). Recovering Intrinsic Images from a Single Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Terven, Juan & Cordova-Esparza, Diana-Margarita. (2023). A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond.
- Li, Gang & Xu, Lei & Wang, Jing. (2008). A new method of vehicle license plate location based on mathematical morphology and texture characteristics. 10.1109/ICIEA.2008.4582662.
- Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E., & Liang, Z. (2019). Apple detection during different growth stages in orchards using the improved YOLO-v3 model. *Computers and electronics in agriculture*, 157, 417–426.
- Timofte, R., *et al.* (2019). NTIRE 2019 Challenge on Video Super-Resolution: Methods and Results. Proceedings of the *IEEE Conference on Computer Vision and Pattern Recognition Workshops*.
- Ukhwah, E. N., Yuniarno, E. M., & Suprpto, Y. K. (2019). Asphalt pavement pothole detection using deep learning method based on YOLO neural network. In *2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)* (pp. 35–40). IEEE.
- Vig, Simar & Arora, Archita & Arya, Greeshma. (2023). Automated License Plate Detection and Recognition using Deep Learning.
- Wang, S., Lin, J., & Zhang, D. (2019). A Survey on Recent Advances in Vehicle License Plate Recognition Systems. *IEEE Transactions on Intelligent Transportation Systems*, 20(4), 1316-1336.
- Wang, W., Liu, W., Liu, X., Bai, S., & Ma, Y. (2022). Attention-guided YOLOv7: An accurate and real-time vehicle license plate detector. *Applied Sciences*, 12(13), 6054.
- Wang, Z., Lv, Y., Wu, R., & Zhang, Y. (2023). Review of GrabCut in Image Processing. *Mathematics*, 11(8), 1965. <https://doi.org/10.3390/math11081965>
- Wei, Q., Zhang, H., & Yong, J. (2023). Focused and Collaborative Feedback Integration for Interactive Image Segmentation. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.2303.11880>

- Wenzheng, Zhang & Kai, Yuan. (2021). Building Outline Extraction Directly Using the U2-Net Semantic Segmentation Model from High-Resolution Aerial Images and a Comparison Study. *Remote Sensing*. 13. 10.3390/rs13163187.
- Wu, D., Lv, S., Jiang, M., & Song, H. (2020). Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Computers and Electronics in Agriculture*, 178, 105742.
- Wu, H., Liu, Y., Xu, X., & Gao, Y. (2022). Object Detection Based on the GrabCut Method for Automatic Mask Generation. *Micromachines*, 13(12), 2095. <https://doi.org/10.3390/mi13122095>
- Xinchun, Wei & Li, Xing & Wei, Liu & Lianpeng, Zhang & Dayu, Cheng & Hanyu, Ji & Yang, J., Wright, J., Huang, T. S., & Ma, Y. (2008). Image Super-Resolution Via Sparse Representation. *IEEE Transactions on Image Processing*.
- Yang, W., & Jiachun, Z. (2018). Real-time face detection based on YOLO. In 2018 1st IEEE *International Conference on Knowledge Innovation and Invention (ICKII)* (pp. 221–224). IEEE.
- Yang, W., *et al.* (2019). Single-Image Super-Resolution: A Benchmark. *IEEE Transactions on Signal Processing*.
- Yang, X., Zhang, Z., Huang, Y., Zheng, Y., & Shen, Y. (2022). Using a graph-based image segmentation algorithm for remote vital sign estimation and monitoring. *Scientific Reports*, 12(1). <https://doi.org/10.1038/s41598-022-19198-1>
- Yin, J., Liu, Y., & Chen, W. (2009). Automatic license plate recognition using a genetic algorithm approach. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(3), 309-316.
- Yu, Y., Wang, C., Fu, Q., Kou, R., Huang, F., Zhou, B., Yang, T., & Gao, M. (2023). Techniques and Challenges of Image Segmentation: A Review. *Electronics*, 12(5), 1199. <https://doi.org/10.3390/electronics12051199>

- Zakria, Z., Deng, J., Kumar, R., Khokhar, M. S., Cai, J., & Kumar, J. (2022). Multiscale and direction target detecting in remote sensing images via modified YOLO-v4. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15, 1039–1048.
- Zhang, C., & Liu, Q. (2021). Benchmarking License Plate Detection and Recognition Models: Challenges and Perspectives. *Sensors*, 21(3), 1120.
- Zhang, X., Ye, Q., & Zhang, Z. (2016). A review on automatic vehicle license plate recognition. *IEEE Transactions on Intelligent Transportation Systems*, 17(12), 3490-3503.
- Zhang, Y., Xu, C., Yang, Y., & Jia, W. (2023). Domain-adaptive vehicle license plate recognition via image style transfer. *Pattern Recognition*, 170, 108306.
- Zheng, Qihua & Li, Wenqing & Hu, Weihua & Wu, Guohua. (2012). An Interactive Image Segmentation Algorithm Based on Graph Cut. *Procedia Engineering*. 29. 1420-1424. 10.1016/j.proeng.2012.01.149.
- Zheng, Y., & Zhang, H. (2022). Video analysis in sports by lightweight object detection network under the background of sports industry development. *Computational Intelligence and Neuroscience*, 2022.
- Zhou, J., Li, H., & Tian, Q. (2014). License plate detection and recognition for traffic monitoring. In *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 1044-1049).
- Zhou, X., Yao, C., Wen, H., Xu, Y., Bai, X., & Lu, S. (2017). EAST: An efficient and accurate scene text detector. *arXiv preprint arXiv:1704.07411*.
- Zhu, Q., Qiao, H., & Zhao, Y. (2018). License plate detection and recognition: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 19(2), 578-596