SMART CAR PLATE RECOGNITION SYSTEM USING MULTI-TASK LEARNING

KHOR YIN LOON

UNIVERSITI TUNKU ABDUL RAHMAN

SMART CAR PLATE RECOGNITION SYSTEM USING MULTI-TASK LEARNING

KHOR YIN LOON

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Electrical and Electronic Engineering with Honours

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

May 2024

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : ______________________

Name : Khor Yin Loon

ID No. : 1903505

Date : 20/5/2024

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"SMART CAR PLATE RECOGNITION SYSTEM USING MULTI-TASK LEARNING"** was prepared by **KHOR YIN LOON** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Electrical and Electronic Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _________________________

Supervisor : Ir. Ts. Dr. Tham Mau Luen

Date : 20 May 2024

Signature : _________________________

Co-Supervisor : Ir. Dr. Chang Yoong Choon

Date : 14 May 24

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor and co-supervisor, Ir. Ts. Dr. Tham Mau Luen and Ir. Dr. Chang Yoong Choon, for their invaluable advice, guidance and enormous patience throughout the development of the research.

In addition, I would also like to express my gratitude to my loving parents and friends who had helped and given me encouragement, motivation and support while I have been conducting this project.

ABSTRACT

Automatic License Plate Recognition (ALPR) systems are crucial in extracting vehicle information. However, ALPR alone is insufficient for robust vehicle owner identification, especially in the event of misidentification or covered license plates (LPs). Acknowledging the significance of vehicle colour in enhancing identification accuracy, this project proposes a more secure and comprehensive approach by integrating Vehicle Colour Recognition (VCR) with LP detection and Optical Character Recognition (OCR) tasks. Unlike the conventional two-stage ALPR systems, this solution introduces a novel one-stage YOLO-based multi-task model. It incorporates additional object detection heads onto the YOLO backbone, allowing for parallel processing and efficient real-time detection for all three tasks. The proposed model achieves spectacular results with mean Average Precision (mAP) scores of 0.778, 0.963, and 0.881 for OCR, LP detection, and VCR, respectively. Promisingly, this model is comparable to single-head, single-task models, which are trained solely for each task. It outperforms a single-head multi-task model, which naively shares all tasks using one single head. Specifically, the model is 1.77x faster than the conventional approach, which involves inference of single-task models for OCR, LP, and VCR sequentially. Experimental results demonstrate that the proposed solution is robust in simultaneously addressing OCR, LP detection, and VCR within a unified, single-stage framework.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

L_{LP}	LP head loss function
L_{OCR}	OCR head loss function
L_{VCR}	VCR head loss function
L_{box}	bounding box loss
L_{cls}	classification loss
L_{detect}	object detection loss function
L_{dfl}	distribution focal loss
L_{total}	total loss function

AI	artificial intelligence
ALPR	Automatic License Plate Recognition
AOLP	Application-Oriented License Plate
AP	Average Precision
CCTV	closed-circuit television
CNN	convolutional neural network
COCO	Common Objects in Context
ConvFLU	convolutional feature leaky unit
CSP	cross-stage partial connections
DFL	distribution focal loss
FLOPs	floating-point operations per second
FPN	feature pyramid structure
FPS	frames per second
GAN	Generative Adversarial Network
GBM	Gated Bridging Mechanism
HSV	Hue, Saturation and Value
LP	license plate
LPR	License Plate Recognition
mAP	mean Average Precision
MTL	Multi-task learning
NLP	natural language processing
OCR	Optical Character Recognition

P	precision
PBFS	Pruning-Based Feature Sharing
PR	Precision-Recall
R	recall
R-CNN	Region-based Convolutional Neural Network
RoI	regions of interest
RTSP	Real-Time Streaming Protocol
SOTA	state-of-the-art
SSD	Single-Shot Multibox Detector
STR	Scene Text Recognition
SVM	support vector machine
TDP	Thermal Design Power
TRDG	Text Recognition Data Generation
UCSD	University of California, San Diego
VCoR	vehicle colour dataset
VCR	Vehicle Colour Recognition
YOLO	You Only Look Once

CHAPTER 1

INTRODUCTION

1.1 General Introduction

Automatic License Plate Recognition (ALPR) systems are designed to automatically detect, recognize, and record license plate (LP) numbers from images or video streams captured by cameras. Typically, most ALPR systems operate through a two-stage process. The first stage involves LP detection, where the system locates LPs from the captured images or video frames. Subsequently, the second stage employs Optical Character Recognition (OCR) to extract the alphanumeric characters present on the detected LPs (Tham and Tan, 2021). ALPR systems are frequently integrated into diverse fields, playing a crucial role in law enforcement, toll collection, parking management, and various other sectors. These systems offer highly effective, precise, and automated approaches for identifying vehicles and extracting associated information.

Currently, real-time vehicle recognition primarily relies on ALPR. Recognising colour as a fundamental aspect of vehicle recognition, Vehicle Colour Recognition (VCR) should be integrated into ALPR. The integration of VCR into ALPR systems facilitates more reliable and precise vehicle identification. This fusion enriches the extracted vehicle information, enabling a more comprehensive identification process (Chen, Bai and Liu, 2014). In law enforcement, the fusion enhances the ability to track and identify vehicles involved in criminal activities or traffic violations. For instance, police departments can swiftly identify vehicles linked to criminal activities by accessing both LP information and specific details about the car's colour and model.

Multi-task learning (MTL) offers an effective approach to perform multiple tasks in a unified model. This approach leverages shared representations across tasks to improve generalisation performance and accuracy (Crawshaw, 2020). Given the shared essence of object localisation and recognition within ALPR and VCR, unifying them into a singular one-stage object detection problem through MTL becomes feasible.

1.2 Importance of the Study

Most existing works on ALPR and VCR employ a two-stage approach, where object detection and object recognition tasks, are conducted sequentially. Although this method has showcased significant success, its sequential nature often results in slower inference times. The requirement for two distinct phases of detection and recognition, each demanding significant computational resources and complex algorithms, poses limitations in real-time applications where speed and efficiency are crucial and paramount. Fortunately, both ALPR and VCR fundamentally revolve around object detection within images or video frames. A single one-stage multi-task object detection model is feasible to accomplish both tasks, leading to faster inference times and reduced computational overhead.

MTL presents an opportunity to leverage shared representations across different tasks, which leads to better generalisation. The multi-task model captures common features shared, enabling knowledge transfer to improve performance on related tasks. Beyond transfer learning, MTL reduces overall model complexity, resulting in more efficient resource utilisation. By eliminating the need for sequential tasks processing, MTL streamlines the inference pipeline, reducing latency and enabling real-time performance.

1.3 Problem Statement

While ALPR and VCR are crucial for comprehensive vehicle recognition, they remain two-stage approach, leading to slower inference times and increased computational overhead. An efficient one-stage object detection framework is paramount in OCR and VCR, especially in real-time applications. Furthermore, the two tasks remain separated, limiting accuracy through loss of information sharing among tasks. To address these challenges, there is a need to consolidate ALPR and VCR into a unified one-stage object detection model.

This fusion necessitates proper branching, ensuring minimal interference but sufficient shared representations across tasks. Appropriate hyperparameters, such as loss function, is crucial for optimising each task contribution, ensuring accurate and reliable vehicle identification. The model architecture should be carefully designed and considered to achieve exceptional performance. Another challenge includes the absence of multi-labelled dataset covering the labelling of all tasks for multi-task model training. Existing datasets typically focus on individual tasks, imposing barrier to the development and training of multi-task model.

1.4 Aim and Objectives

Recognizing a car plate number using any deep learning technique may not be safe enough for owner identification. A more robust approach would involve recognising a combination of car colour and car plate number. Therefore, this project aims to develop a car plate recognition system using MTL. The objectives of the study are as follows:

1. To develop a multi-task learning model that can simultaneously perform car colour and car plate number recognition.
2. To implement the developed AI model in a real-world scenario.
3. To compare the performance of the developed multi-task artificial intelligence (AI) model with conventional multi-AI models.

1.5 Scope and Limitation of the Study

In this project, ALPR and VCR are merged into a unified single-stage object detection model. This paradigm shift aims to streamline the process by removing sequential stages, thus boosting the inference speed significantly. By treating these intertwined tasks as a holistic object detection problem, the proposed methodology seeks to not only improve efficiency but also retain accuracy, paving the way for real-time applications in the domain of car plate recognition systems.

There is an absence of unified approach to combine ALPR and VCR throughout the literature study, implementing difficulty in the multi-task model development. Model architecture is carefully considered to strike balance between shared representations and task-specific features. To the best of the knowledge, there is no publicly available multi-labelled dataset that covers the labelling of all tasks. Manual data collection and labelling resembling actual implementation are crucial for training multi-task model.

1.6 Contribution of the Study

This study contributes to a multi-task model in a real-world setup, integrating ALPR and VCR for a more comprehensive vehicle recognition. The main contributions of this study are summarised as follows:

1. An in-house multi-labelled dataset that covers VCR, LP detection, and OCR simultaneously is curated for model development. There is an absence of multi-labelled dataset covering the labelling of all tasks.
2. Meanwhile, a lightweight single-stage model that integrates the three tasks into a unified framework is developed. This is especially advantageous for multi-tasking scenarios that require real-time processing.
3. Extensive experiments are conducted to compare the performance of the proposed model with existing state-of-the-art methods in terms of accuracy and inference speed.

1.7 Outline of the Report

The outline of this report consists of five chapters, providing readers adequate information of relevant studies and that of the developed project. This report is organised as follows:

Chapter 1 discusses the general introduction, problem area, aims and objectives, scope and limitation as well as the contribution of the study, to provide a clear overview of the project.

Chapter 2 reviews a large number of relevant research related to ALPR, VCR and MTL. The theory and related works are discussed and evaluated, highlighting the current gaps in related research.

Chapter 3 describes the methodology and work plan of the project to provide all necessary details for replicating the project. The required resources, both hardware and software, are listed together with the model pipeline for model development.

Chapter 4 demonstrates and analyses the performance of the proposed model with thorough discussion. Result interpretations are provided for different hyperparameter tuning including accuracy, speed and number of parameters of the proposed multi-task model.

Chapter 5 concludes the achievements of this project aligning with the stated aims and objectives. Recommendations on future works are provided as well based on the potentials and limitations of current project.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

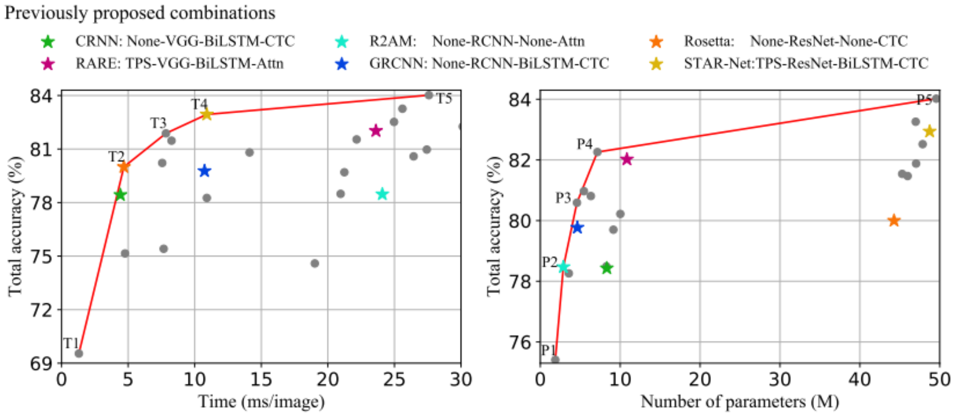
ALPR systems play a pivotal role by automating LPs detection and subsequent OCR to extract alphanumeric information. However, this method fails particularly when LPs cannot be detected accurately, changed or obscured. VCR offers complementary information for vehicle recognition and should be integrated into ALPR. However, existing methodologies predominantly adopt a two-stage approach for both ALPR and VCR, resulting in increased latency. Fortunately, MTL offers a promising avenue to consolidate LP detection, OCR, and VCR into a unified framework, thereby addressing latency issue. This literature review explores the evolution of ALPR and VCR, as well as the potential of MTL in enhancing ALPR systems. Additionally, the study reviews object detection models as the backbone of the project, and explores the available public datasets.

2.2 Automatic License Plate Recognition (ALPR)

ALPR has witnessed significant advancements in recent years, particularly in the realms of LP detection and OCR. Past works have focused on developing robust algorithms for LP detection and OCR, leveraging techniques ranging from traditional image processing to deep learning approaches (Shashirangana et al., 2021). Generally, ALPR systems employ a two-stage approach, where LP regions are first detected in the image, followed by OCR to extract the alphanumeric characters.

With the advancements in deep learning, several object detection models including Region-based Convolutional Neural Network (R-CNN), Single-Shot Multibox Detector (SSD), and You Only Look Once (YOLO) are introduced for LP detection. Tu and Du (2022) suggest a hierarchical structure with multiple levels of R-CNN to handle sub-tasks like vehicle detection and license plate recognition, which reduces the overall computational requirements with improved accuracy. Considering the slow inference speed due to its multi-stage architecture, a lightweight single-stage SSD model is proposed to localise license plates before passing to OCR model for real-time edge computing applications (Awalgaonkar, Bartakke and Chaugule, 2021). Similarly, Al-batat et al. (2022) introduce single-stage YOLO models for vehicle, LP and alphanumeric characters detection separately, utilising output from the previous stage.

On the other hand, the character recognition methods in the subsequent stage vary among different studies. One popular method is to pass the extracted LPs to state-of-the-art OCR models such as Tesseract OCR and EasyOCR, which are easy to use due to their robustness and open-source nature (Tham and Tan, 2021). However, these OCR models may struggle with the variation in font styles and designs within LPs. These nuances may not be fully addressed by generic OCR models, leading to suboptimal performance. Rather than utilising these OCR models directly, they should be trained with relevant dataset to familiarise themselves with the unique characteristics of LPs for character recognition. Acknowledging that OCR forms a subset of Scene Text Recognition (STR) model specialising in structured text extraction, the study on a unified four-stage STR framework offers valuable insights. Most existing STR models fit into four stages consisting of transformation, feature extraction, sequence modelling and prediction for effective text



#	Trans.	Feat.	Seq.	Pred.	Acc. %	Time ms	params $\times 10^6$
T1	None	VGG	None	CTC	69.5	1.3	5.6
T2	None	**ResNet**	None	CTC	80.0	4.7	46.0
T3	None	ResNet	**BiLSTM**	CTC	81.9	7.8	48.7
T4	**TPS**	ResNet	BiLSTM	CTC	82.9	10.9	49.6
T5	TPS	ResNet	BiLSTM	**Attn**	84.0	27.6	49.6

(a) Accuracy versus time trade-off curve and its frontier combinations

#	Trans.	Feat.	Seq.	Pred.	Acc. %	Time ms	params $\times 10^6$
P1	None	RCNN	None	CTC	75.4	7.7	1.9
P2	None	RCNN	None	**Attn**	78.5	24.1	2.9
P3	**TPS**	RCNN	None	Attn	80.6	26.4	4.6
P4	TPS	RCNN	**BiLSTM**	Attn	82.3	30.1	7.2
P5	TPS	**ResNet**	BiLSTM	Attn	84.0	27.6	49.6

(b) Accuracy versus memory trade-off curve and its fronti combinations

Figure 2.1: Two-Types of Trade-Offs of STR Module Combinations (Baek et al., 2019)

recognition. The research work analyses the contributions of individual modules to performance in terms of accuracy, speed, and memory demand, using a consistent set of datasets (Baek et al., 2019). Figure 2.1 illustrates the trade-off plots of all STR module combinations. Alternatively, some studies advocate for treating LP character recognition as an object detection task, utilizing separate object detector for each character within the LP (Henry, Ahn and Lee, 2020). A notable drawback of the two-stage ALPR approach is its susceptibility to errors in the LP detection stage, impacting character recognition accuracy. Additionally, this multi-stage process often results in reduced inference speed.

Substantial quantity of high-quality datasets is essential for effective ALPR models training. However, the scarcity of publicly available Malaysian LP datasets and the resource-intensive nature in collecting real license plate images possess challenges to the success of ALPR system. To address this issue, a novel synthetic dataset generator for Malaysian LPs is introduced to bridge the research gap (Asaad, Faizabadi and Mohd Zaki, 2023). Specifically, it employs the Text Recognition Data Generation (TRDG) module to simulate real-world Malaysian LPs, considering factors such as font type, background, margins, and letter spacing (Belval E, 2020). Additionally, augmentation techniques are applied to replicate and reflect potential real-world variations. Another approach involves generating license plate images from a small set of real images using Generative Adversarial Network (GAN), where over 9000 realistic LP images are generated from merely 159 web-scraped real LP images (Han et al., 2020). Both proposed synthetic dataset generators significantly improve the accuracy of ALPR on real LPs.

2.3 Vehicle Colour Recognition (VCR)

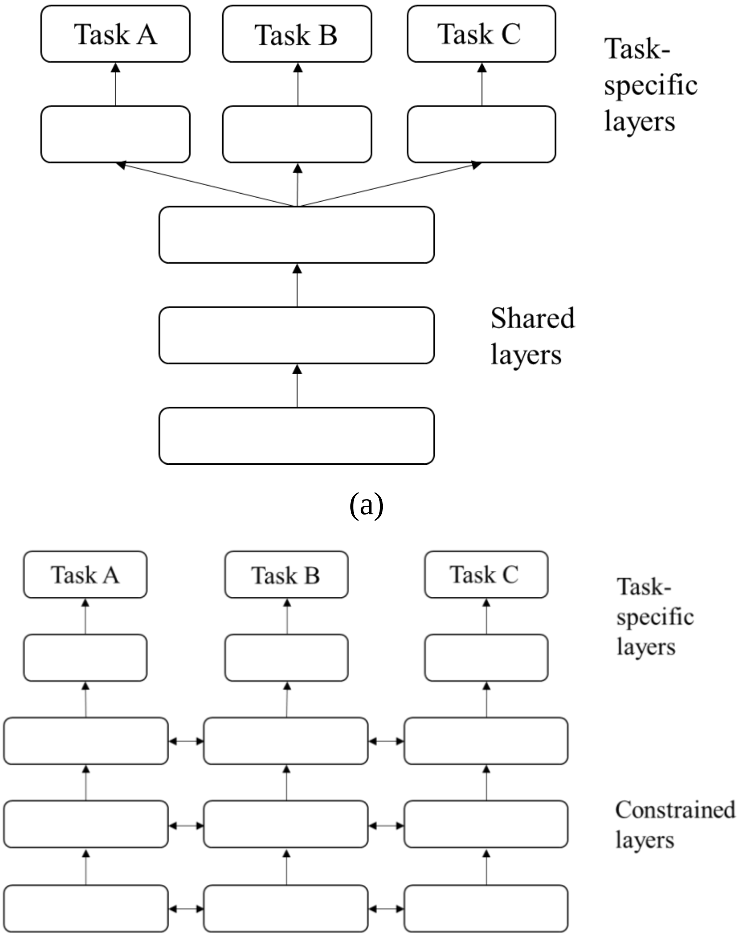
VCR has evolved from traditional methods such as support vector machine (SVM) to the recent end-to-end trainable convolutional neural network (CNN). Razalli et al. (2020) propose Hue, Saturation and Value (HSV) colour segmentation to extract the colour information from detected emergency vehicle lights, which are then fed into an SVM classifier for colour classification. Recent advancements have predominantly embraced deep learning techniques, for both vehicle detection and colour classification tasks.

For instance, a pretrained YOLOv5 is fine-tuned to detect and crop vehicles before passing into RepVGG for vehicle make, model, year and colour recognition (Panetta et al., 2021). A similar framework is adopted by Tariq et al. (2021), where Faster R-CNN is used to first detect vehicles using a regional proposal network, followed by colour classification in a single-stage approach.

Despite these advancements in VCR, the majority of existing frameworks still adhere to a two-stage process involving vehicle detection followed by colour classification. Remarkably, there has been minimal effort directed towards simultaneously addressing these tasks within a unified framework. This division into separate stages not only introduces latency but also poses challenges in integrating the outputs seamlessly. Moreover, the absence of publicly available datasets specifically designed for joint vehicle detection and colour classification further discourages the development of one-stage vehicle colour recognition.

2.4 Multi-Task Learning (MTL)

MTL is a machine learning paradigm where a single model is trained to perform multiple tasks simultaneously. This approach is particularly beneficial when the tasks share some common underlying patterns or features, as leveraging these shared representations can lead to better generalization and overall performance (Wu, Zhang and Ré, 2020). MTL models typically fall into two main architectures: hard parameter sharing and soft parameter sharing. In hard parameter sharing, the task-specific heads are connected to a shared backbone. The shared backbone is trained simultaneously to learn features for multiple tasks, reducing the risk of overfitting to any specific task, which leads to improved generalisation performance. Conversely, in soft parameter sharing, each head possesses its own dedicated backbone. The parameters of each backbone are regularised using L1/L2 loss to encourage similarity, thereby facilitating knowledge sharing among the backbones (Crawshaw, 2020). Figure 2.2 illustrates the general overview of the two architectures.



(a)

(b)

Figure 2.2: (a) Illustration of Hard Parameter Sharing (b) Illustration of Soft Parameter Sharing

As mentioned, hard parameter sharing directly shares parameters across all tasks, forcing shared representations, which act as a form of regularization to prevent overfitting. While the shared backbone features computational efficiency, it imposes strict constraints on parameter sharing, limiting effectiveness for tasks with diverse characteristics (Vafaeikia, Namdar and Khalvati, 2020). In contrast, soft parameter sharing introduces flexibility through feature sharing mechanism across backbones, regularising knowledge transfer across different tasks. The model adapts to unique characteristics of each task, leading to better performance where tasks have diverse requirements. However, this flexibility comes with increased computational overhead and risk of overfitting, as each task has its separate parameters (Sun et al., 2020). The comparison of MTL architectures is presented in Table 2.1.

Table 2.1: Comparison of MTL Architectures

MTL Architectures	**Benefits**	**Limitations**
Hard parameter sharing	• Strong regulatisation: Shares a common backbone, which reduces the risk of overfitting and improves generalisation • Practicability: Adding tasks involves branching an additional head from the original model	• Lack of task-specific adaptation: Imposes strict sharing, limiting effectiveness for tasks with diverse characteristics
Soft parameter sharing	• Flexibility: Allows feature sharing across backbones while adapting to unique task characteristics for better performance among diverse task requirements • Selective transfer: Facilitates selective knowledge	• Computational overhead: Increased complexity and risk of overfitting due to separate parameters for each task

MTL Architectures	Benefits	Limitations
	transfer by regularising parameters across tasks.	

Among the two techniques, hard parameter sharing stands out as a more prevalent approach due to its practicality, where adding an additional task involves branching an extra head from the backbone or neck of the original model. This practicability is notably showcased in the HydraNet architecture, a framework notably utilized in Tesla's self-driving cars. In HydraNet, a single backbone branches out into multiple task-specific heads, each dedicated to a distinct task relevant to autonomous driving (Agrawal, 2023). Similarly, Wang et al. (2023) extended the capabilities of the object detection model YOLOv8 by integrating two additional heads into its backbone for drivable area, and lane line segmentation, respectively. Beyond autonomous driving, hard parameter sharing also finds application in disaster management scenarios. Notably, Tham et al. (2021) integrated a MobileNetV2-like disaster classification head into a YOLOv3 victim detection model, which further extends into edge computing application (Wong et al., 2022; Tham et al., 2023). This integration reduces computational power, allowing the deployment of the multi-task model on resource-constrained devices.

On the other hand, soft parameter sharing involves complex parameter sharing across the backbones, introducing enormous space of possible parameter sharing architectures. To address this issue, Mao and Li (2021) propose Gated Bridging Mechanism (GBM) for selective information exchange and filtering between tasks, which yields better performance than conventional mechanisms, in natural language processing (NLP) applications. Similarly, Chen et al. (2022) introduce Pruning-Based Feature Sharing (PBFS) which integrates model pruning into soft parameter sharing, allowing tasks to select parameters from a shared subnet based on their needs and prune noise parameters. Their work extends soft parameter sharing applications into classification and regression tasks with optimal results, underscoring the effectiveness of PBFS in knowledge transfer. Additionally, soft parameter sharing finds applications in computer vision field where convolutional feature leaky unit (ConvFLU) is introduced to selectively transfer beneficial features between tasks while filtering out irrelevant information (Zhao et al., 2020).

In the context of this research, MTL offers an opportunity to merge the two-stage process of ALPR and VCR into a unified, one-stage model. For example, the study by Huang et al. (2021) aligns with the objective, where a multi-task model is developed for LP detection and OCR. The multi-task model begins with a ResNet-50 with a feature pyramid structure (FPN) serving as the backbone, with different layers of the FPN branching out for the two tasks. Notably, this multi-task model achieves one-stage ALPR, since LP detection and OCR are performed simultaneously. However, there is currently no unified model that could perform ALPR and VCR simultaneously, in a one-stage approach.

2.5 Object Detection Models

Both ALPR and VCR tasks revolve around object detection within images or video frames. Hence, selecting a suitable object detector is crucial to unify them into a single object detection framework, laying the foundation for the multi-task model. Throughout the studies, several state-of-the-art (SOTA) object detection models, such as R-CNN, YOLO and SSD, are proposed to address these challenges.

2.5.1 Region-Based Convolutional Neural Network (R-CNN)

R-CNN is a two-stage object detection model which extracts region proposals to CNN for feature extraction and SVM for classification. It proposes a simpler alternative to complex ensemble systems that rely on multiple image features and context through region proposals. Traditional object detectors are computationally expensive where regions of interest (RoI) with different spatial interest and aspect ratios are selected before passing to CNN. To address this issue, R-CNN extracts only around 2000 region proposals using selective search algorithm, reducing the computational need for CNN feature extraction. SVM then utilises these extracted features for subsequent bounding box regression and object classification. Figure 2.3 illustrates the R-CNN

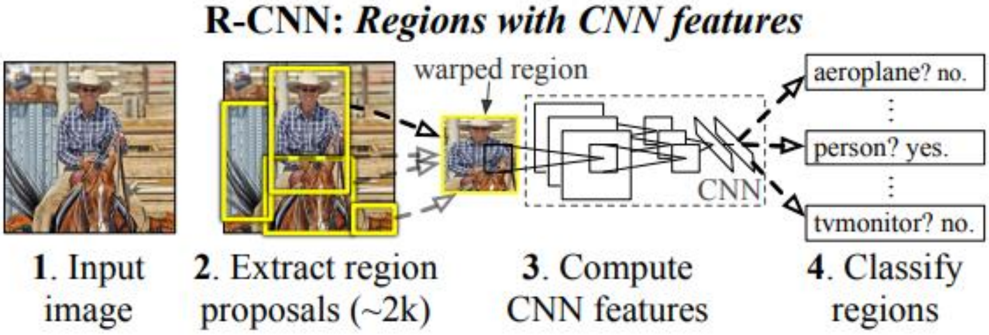


Figure 2.3: R-CNN Model Overview (Girshick et al., 2013)

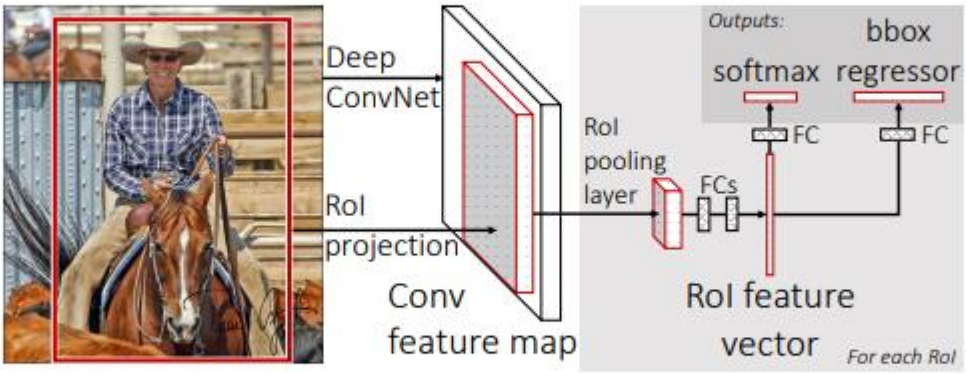


Figure 2.4: Fast R-CNN Model Overview (Girshick, 2015)

model overview. This system is efficient due to the shared CNN computations across all categories with low-dimensional feature vectors computed (Girshick et al., 2013). Despite these advancements, R-CNN still suffers from slow training and inference time, in which each region proposal must be processed separately through CNN, leading to increased computational power. Notably, the fixed selective search algorithm in region proposal module limits the model learning process, leading to possible generation of bad region proposals (Mijwil et al., 2022).

2.5.2 Fast R-CNN

Realising the slow R-CNN architecture, the same authors innovate upon the previous work where Fast R-CNN performs convolution operation once per image for feature map generation. A RoI pooling layer then extracts feature vector from the feature map into RoI feature vector for subsequent bounding box regression and object classification. Benefiting from the improved feature extraction and object detection framework, Fast R-CNN benchmarks faster training speed and higher detection accuracy than R-CNN on the same baseline dataset (Girshick, 2015). However, Fast R-CNN still relies on selective search for generating region proposals, which remains a bottleneck in terms of speed. Figure 2.4 demonstrates the Fast R-CNN model architecture.

2.5.3 Faster R-CNN

Despite the advancements made in R-CNN and Fast R-CNN, these object detection models still depend on region proposal algorithms for object localisation. The computational expensive and slow nature of region proposal algorithms limit the training and inference speed of object detection models, especially for real-time applications. For efficient region proposal generation, (Ren et al., 2015) introduce Region Proposal Network (RPN) that share convolutional features with detection networks, elevating the cost for region proposals. The proposed object detection model combines RPN and Fast R-CNN detector, addressing the region proposal computation through sliding window approach, as shown in Figure 2.5. Overall, Faster R-CNN achieves improved speed and accuracy by leveraging the shared convolutional features with RPN, overcoming the limitations of its predecessors.

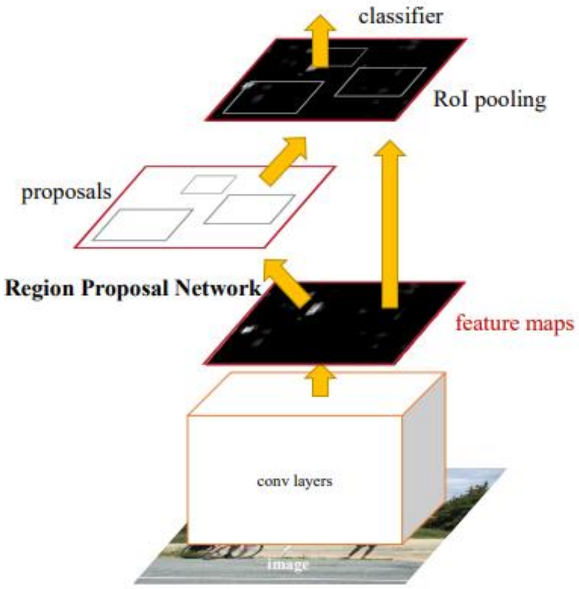


Figure 2.5: Faster R-CNN Overview (Ren et al., 2015)

2.5.4 You Only Look Once (YOLO)

Conventional region proposal methods are computationally expensive, fostering the development of YOLO as a single-stage object detector. YOLO frames detection as a regression problem, predicting bounding boxes and class probabilities directly from full images in one evaluation. Unlike other detection systems, YOLO sees the entire image during training and testing, which allows it to learn contextual information about object classes and their appearance. The system divides input image into grid cells which is responsible for detecting objects whose centre falls within it. Each grid cell predicts bounding boxes and confidence scores, reflecting the likelihood of an object's presence and the prediction accuracy. To provide class-specific confidence scores for each box, each grid cell predicts conditional class probabilities, given that an object is present. The architecture uses a single CNN to predict multiple bounding boxes and class probabilities simultaneously, enabling end-to-end training and real-time speeds (Redmon et al., 2015). To date, YOLO family has evolved through multiple generations until YOLOv9 which is the latest instalment in the YOLO series, underscoring the robustness of YOLO model in object detection. The comparison results of several SOTA object detection models are shown in Figure 2.6.

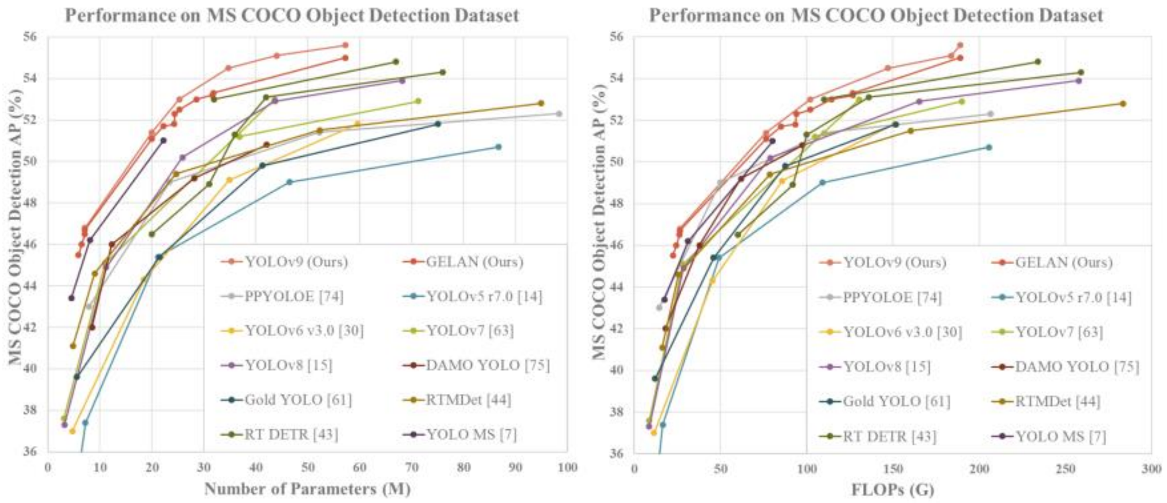


Figure 2.6: Comparisons of Object Detection Models on Microsoft COCO Dataset (Wang, Yeh and Liao, 2024)

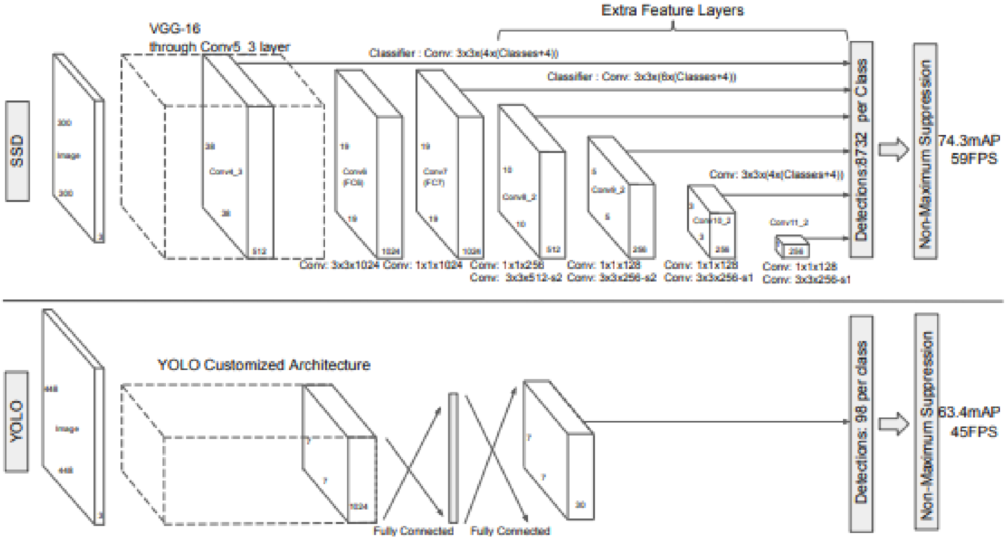


Figure 2.7: Comparison of SSD and YOLOv1 Overview (Liu et al., 2015)

2.5.5 Single Shot MultiBox Detector (SSD)

Similar to YOLO, SSD is a single-stage object detector that eliminates the need for region proposals and feature resampling, making it faster and simpler. SSD has similar architecture as YOLO, where SSD divides each input image into grids of cells for bounding box and score generation in each grid. However, SSD is equipped with auxiliary structures for model enhancement in terms of both speed and accuracy. Considering that YOLO struggles with small object detection due to insufficient spatial resolution, SSD adds convolutional feature layers with different sizes to the backbone, generating multi-scale feature maps for detection. For each spatial location in the feature maps, SSD predicts bounding boxes with respect to different aspect ratio using separate predictors. With these modifications, SSD is able to achieve a higher accuracy and inference speed compared to Faster RCNN and YOLOv1, even with lower resolution input (Liu et al., 2015). A comparison of SDD and YOLOv1 architecture is illustrated in Figure 2.7.

2.5.6 Comparison of Object Detection Models

Table 2.2 presents a comparison of object detection models studied.

Table 2.2: Comparison of Object Detection Models

Object Detection Model	**Detection Method**	**Limitation**
R-CNN	Two-stage	Computationally expensive due to selective search in region proposals
Fast R-CNN	Two-stage	Computationally expensive due to selective search in region proposals
Faster R-CNN	Two-stage	High model complexity due to RPN
YOLO	Single-stage	Difficulty in detecting small objects due to insufficient spatial resolution
SSD	Single-stage	High model complexity due to multiple convolutional feature maps at different scales

2.6 Datasets Available

Many studies have been conducted on Malaysian ALPR systems with good recognition accuracy (Marzuki et al., 2019; Tham and Tan, 2021). However, as described in Section 2.2, there is still a lack of large publicly available dataset with high quality and reliability for ALPR task. Acknowledging this issue, a research work introduces public RodoSol-ALPR dataset with Mercosur LPs, which is a new unified LP standard in Mercosur countries, captured at toll booths (Laroca et al., 2022). Notably, such dataset is only applicable to Mercosur countries with similar LP layout, whereby other countries require algorithm changes to achieve similar results. Henry et al. (2020) propose a two-stage YOLOv3-based multinational ALPR system trained on diverse combinations of public dataset from five countries, demonstrating high accuracy and robustness without additional information needed. However, the evaluation on datasets from only five countries is insufficient to represent the full diversity of global license plate designs, leading to poor generalisation on other countries. On the other hand, Malaysian ALPR dataset comprises a combination of University of California, San Diego (UCSD) and MIMOS datasets, both of which are not publicly available, hindering Malaysian ALPR system development (Asaad, Faizabadi and Mohd Zaki, 2023).

VCR differs from ALPR in which the dataset is not limited by geographical distribution, allowing a more generalised approach by utilising public datasets from any regions for robust model training. Chen et al. (2014) proposes a dataset consisting of 15601 images with eight colour classes, which is challenging due to noises imposed by illumination variation, haze and overexposure. The scarcity of colour classes motivates Panetta et al. (2021) to create vehicle colour dataset (VCoR) with over 10k images and 15 colour classes. However, these datasets are limited to colour classification usage, while research works often require manual data collection and annotation for VCR task, which is treated as an object detection problem (Tariq, Khan and Ghani Khan, 2021). Overall, there is currently no publicly available multi-task dataset covering both ALPR and VCR tasks within a unified framework. The comparison of datasets available for both ALPR and VCR tasks is presented in Table 2.3.

Table 2.3: Dataset Comparison

Task	Dataset	Size	Limitation
ALPR	RodoSol-ALPR	20000	• Consists only vehicle images with Mercosur LPs
	KarPlate	4267	• Focuses on Korean LPs
	Application-Oriented License Plate (AOLP) Dataset	2049	• Focuses on Taiwanese LPs
	Medialab License Plate Recognition (LPR) Database	716	• Focuses on Greek LPs
	Caltech Cars (Rear) 1999	126	• Focuses on

Task	Dataset	Size	Limitation
	Dataset		LPs from the United States of America (USA)
	University of Zagreb Dataset	510	• Focuses on Croatian LPs
	UCSD and MIMOS Datasets	20105	• Not publicly available
VCR	Image Dataset by Chen, Bai and Liu (2014)	15601	• Limited to colour classification usage • Noisy images due to bad weather
	VCoR	Over 10k images	• Limited to colour classification usage

2.7 Summary

Generally, two-stage approach involves object detection, followed by object classification. In the first stage, object detection models like YOLOv5 and Faster R-CNN localise objects of interest for subsequent classification tasks. For license plate recognition, ALPR predominantly relies on state-of-the-art OCR models to extract alphanumeric information. Similarly, VCR employs classification models like RepVGG to determine vehicle colours. Although there exists one-stage approach for both ALPR and VCR, a unified model that could perform ALPR and VCR simultaneously is still absent. Past works in MTL demonstrates the potential of hard parameter sharing in bridging the research gap, particularly in developing a unified model for both tasks seamlessly. However, there is still an absence of public multi-task dataset available for both ALPR and VCR tasks.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

In this work, a multi-task model is proposed to address ALPR and VCR simultaneously. Specifically, both ALPR and VCR tasks are approached as object detection problems, unifying the two tasks into a one-stage end-to-end object detection model. Additional heads are integrated onto YOLOv8 backbone with weighted loss functions, enhancing feature generalisation and task-specific learning. During this phase, a multi-labelled dataset containing ALPR and VCR tasks is curated for multi-task model development.

3.2 Hardware

Training an AI model requires a robust and well-configured setup, where Intel NUC paired with Sonnet eGPU is leveraged for developing this project. Coupled with the flexible customisation options offered by Ubuntu, this setup provides a reliable environment to tackle the project requirements effectively.

3.2.1 Intel NUC 10i7 Mini PC

During the development phase, the project relies on Intel NUC 10i7 Mini PC, as shown in Figure 3.1. This CPU boasts 64-bit, six-core performance and incorporates a substantial 12 MB Intel Smart Cache, ensuring efficient handling of tasks. With support for Hyper-Threading, the 10th generation Intel i7 Core Processor efficiently manages up to 12 threads concurrently. It

Figure 3.1: Intel NUC 10i7 Mini PC

dynamically scales up to an impressive Turbo Boost of 4.7 GHz, ensuring swift execution of tasks with a base speed of 1.1 GHz. Notably, its Thermal Design Power (TDP) is rated at a modest 25W, offering significant energy efficiency compared to conventional desktops or laptops (Anon., 2021). The mini PC's compact form factor makes it highly portable, while its 64GB of RAM and 2TB SSD ensure uninterrupted performance throughout the project's lifecycle.

3.2.2 Sonnet eGPU Breakaway Box 750

Deep learning demands substantial computational power for model training. To facilitate this project, Sonnet eGPU Breakaway Box 750, as depicted in Figure 3.2, is employed. This eGPU connects a high-performance NVIDIA GeForce RTX 1080 Ti to Intel NUC via Thunderbolt 3 port, thereby enabling GPU acceleration to enhance the computational capacity required for training deep learning models. The eGPU is equipped with a sizable built-in fan, which operates at variable speeds and is temperature-controlled, ensuring quiet and effective cooling for the installed GPU card (Anon., 2024a).

3.3 Software

Throughout the project, several framework and software including Python, OpenCV, PyTorch, Google Colaboratory and YOLOv8 are utilised for multi-task model development.

Figure 3.2: Sonnet eGPU Breakaway Box 750

3.3.1 Python

Python, with its logo shown in Figure 3.3, is the programming language of choice for developing this deep learning project. It is highly suitable for deep learning applications owing to its flexibility, user-friendliness, and an extensive array of robust libraries and frameworks, including PyTorch and OpenCV (Sayantini, 2019). Furthermore, Python has a large active community of developers, ensuring large number of tutorials and documentation readily available. This community support greatly facilitates development, enabling swift resolution of common issues and access to guidance during model development.

3.3.2 OpenCV

OpenCV as shown in Figure 3.4, is an open-source software library for computer vision and machine learning. OpenCV is known for its vast algorithms for image processing operations, machine learning applications and computer vision tasks. With its support for Python interface and compatibility with Linux operating system, it is used thoroughly in the project development. Moreover, OpenCV has a large user community, reflecting its widespread adoption and reliability, especially in real-time applications where speed is paramount (Anon., 2024).

Figure 3.3: Python Logo

Figure 3.4: OpenCV Logo

3.3.3 PyTorch

PyTorch, depicted by its logo in Figure 3.5, stands out as an open-source machine learning framework employed for training deep neural networks. The fundamental building block of PyTorch is tensor, which is similar to an array or matrix. These tensors encapsulate model inputs, outputs, and parameters. The various tensors available within this library elevate the neural network development process. Notably, PyTorch is constructed using Python and boasts a robust ecosystem of tools, including ONNX, along with support for GPU training (Jye, 2022).

3.3.4 Google Colaboratory

Google Colaboratory, also known as Google Colab, is a browser-hosted Jupyter notebook service that eliminates the need for installation or setup. It offers free computing resources, including GPU and TPU, allowing users to execute computationally intensive tasks, such as machine learning, at impressive speeds. The platform's seamless integration with Google Drive and effortless sharing features makes it a convenient and accessible choice for project development (Aminu, 2023). In this project, the powerful GPU capabilities of Google Colab are harnessed to accelerate model training, ensuring a faster and more efficient process. Its logo is shown in Figure 3.6.

Figure 3.5: PyTorch Logo

Figure 3.6: Google Colaboratory Logo

3.3.5 YOLOv8

YOLOv8, developed by Ultralytics, is a cutting-edge, SOTA object detection model. YOLO is a one-stage object detector that analyses and makes predictions for bounding boxes and object labels within image in one go. In contrast to typical two-stage object detection algorithms, YOLO significantly enhances the speed of overall system. Due to its unique design, YOLO is highly effective for object identification, capable of detecting objects in real-time with high accuracy. With its speed, accuracy, and ease of use, YOLOv8 is an excellent choice for object detection requirement in this project (Jocher, Chaurasia and Qiu, 2023).

3.4 Work Plan

3.4.1 Dataset

The dataset used for multi-task training is obtained predominantly from a residential area, consisting of 1555 images. This dataset collected for model training is insufficient, particularly for OCR tasks, where the characters are small and diverse. Realising this issue, synthetic data generation techniques are employed to create five synthetic variations for each real data instance (Belval E, 2020). These variations focus on modifying LP and OCR components, generating replicas of the original data. Notably, the VCR image-label pairs remain unchanged, as VCR task is not subject to modification during the synthetic data generation process. This results in a fivefold increase in the number of data instances for LP and OCR tasks. Figure 3.7 illustrates the synthetic data generation pipeline.

Each image is annotated precisely on characters, license plates and cars' colour, where several instances for each class may present in the same image as illustrated in Figure 3.8. The dataset is then split into train, test and valid sets, following a ratio of 7:2:1. Data augmentation is applied solely on training set, resulting in 3x original number of training data. The distribution of instances for each task is presented in Table 3.1.

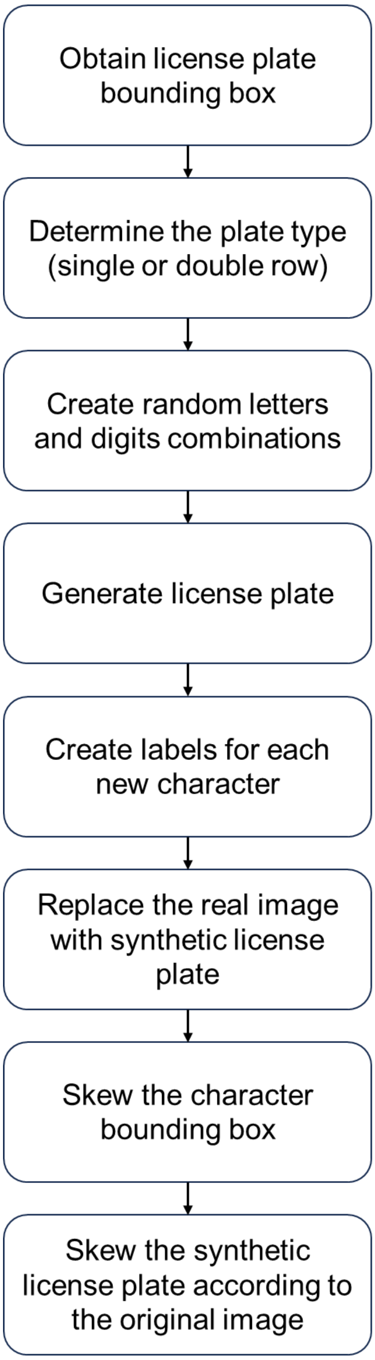

Obtain license plate bounding box
Determine the plate type (single or double row)
Create random letters and digits combinations
Generate license plate
Create labels for each new character
Replace the real image with synthetic license plate
Skew the character bounding box
Skew the synthetic license plate according to the original image

Figure 3.7: Synthetic Dataset Generation Flowchart

Table 3.1: Total Objects Annotated for Each Task

	Synthetic Data			**Real Data**		
Class Labels	**Train**	**Validation**	**Test**	**Train**	**Validation**	**Test**
LP Detection	16926	1613	804	3510	327	184
OCR	118482	11291	5628	22536	2132	1113
VCR	16929	1583	815	2456	325	192

	Synthetic Data			**Real Data**		
Class Labels	**Train**	**Validation**	**Test**	**Train**	**Validation**	**Test**
Total	152337	14487	7247	28502	2784	1489

In the model training pipeline, two-step approach is adopted, where synthetic dataset serves as the foundation for model pretraining, followed by fine-tuning using real data. However, the synthetic data generation process introduces an imbalance within the dataset, with a ratio of 1:5 for VCR to LP and OCR image-label pairs. To mitigate this issue, the loss function for each task is balanced, which is further discussed in subsequent section.

3.4.2 Model Architecture

There are multiple popular object detection models, including R-CNN, SSD, and YOLO. Among these, YOLO has emerged as the *de facto* object detection model for real-time applications due to its high inference speed and accuracy (Wong et al., 2023). In this project, YOLOv8 is chosen as the foundation framework for the multi-task model due to its user-friendliness and well-documented workflows that streamline training and deployment (Jocher, Chaurasia and Qiu, 2023). The smallest YOLOv8 variant, YOLOv8n, is chosen, since it is unwise to use larger model when dealing with limited dataset, as it may lead to overfitting.

YOLOv8 builds upon the previous versions of YOLO algorithms, where its architecture comprises of two main parts: backbone and head. It utilises a modified version of CSPDarknet53 as backbone with cross-stage partial connections (CSP) to enhance information flow between layers. The series of convolutional layers in the head takes feature maps from P3, P4 and P5 of the backbone as inputs. In YOLOv8, C2f module is adopted to extract features at three different scales, replacing traditional YOLO neck. Lastly, three detection head models are used to detect objects based on the three different features, specialising in small, medium and large object detection respectively. The head models consist of fully connected layers with anchor-free detection, making the model more adaptive to object shapes and sizes.

In this project, additional heads are incorporated onto YOLOv8 backbone, forming a multi-head YOLOv8 model to solve the aforementioned

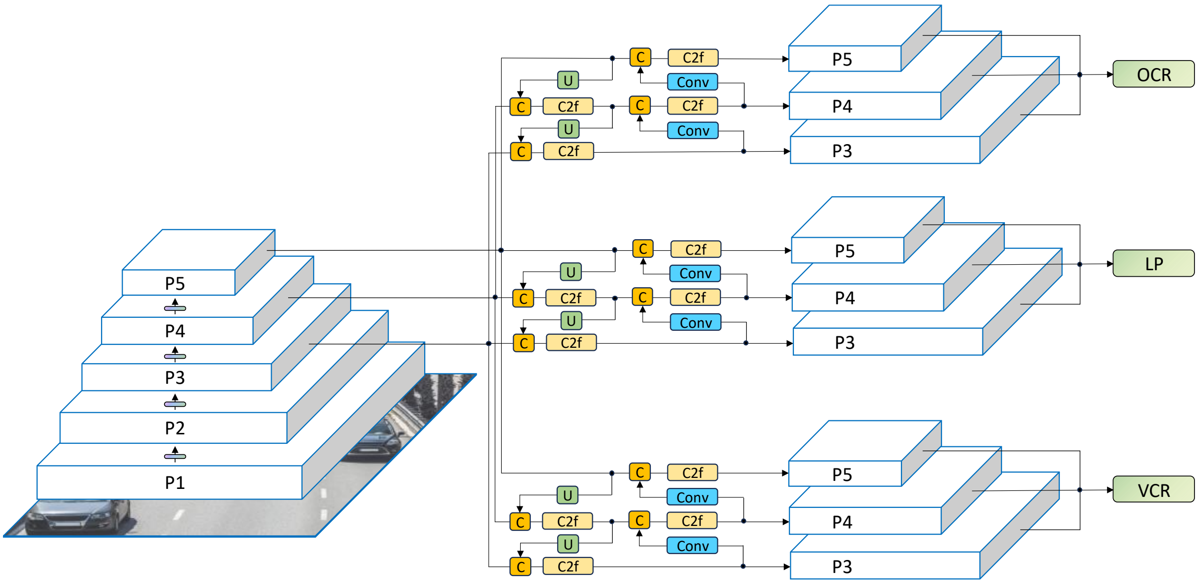


Figure 3.9: Multi-Task YOLOv8 Architecture for OCR, LP Detection and VCR

tasks. Through empirical study, the optimal number of heads is determined to be three, dedicated to i) OCR, ii) LP detection and iii) VCR. These three heads are selected based on object size and task characteristics. The shared backbone facilitates knowledge transfer across multiple tasks, while the task-specific head is tailored to the specific requirements of respective task, enhancing model accuracy. Together, a unified multi-task model for ALPR and VCR is formed. Figure 3.9 illustrates the multi-task model architecture.

3.4.3 Loss Functions

Several methods have been explored to optimize multiple loss functions simultaneously from a multi-objective optimisation perspective. The simplest approach is to naively sum them up into a single scalar loss value (Gong et al., 2019). However, not all objectives hold the same level of priority. Hence, another common approach is to perform a weighted sum of losses for each task (Xin et al., 2022). Other techniques for multi-task optimisation exist, which involves incorporating additional regularisation or normalisation to mitigate gradient conflicts. For instance, GradNorm is a method that adjusts gradient norms across all tasks during training through a novel gradient loss (Chen et al., 2018). However, these methods often come with additional computational overheads without a guaranteed performance improvement compared to the weighted sum approach. To ensure scalability and practicality, weighted sum approach is employed for multi-loss optimisation.

The proposed multi-task YOLO consists of three heads, each corresponding to one of the three tasks: OCR, LP detection and VCR. Let L_{OCR}, L_{LP} and L_{VCR} denote the loss function for each task-specific head. Since all three tasks are treated as object detection problems, each head can adopt the YOLOv8 object detection loss function, L_{detect} as its loss function. The anchorless YOLOv8 detect head employs a decoupled approach to transform high-dimensional features into three outputs using convolutional layers. Each output includes class predictions and bounding boxes relative to its resolution. The detect head loss function, L_{detect} is computed using the Equation (3.1) shown below:

$$L_{detect} = L_{box} + L_{cls} + L_{dfl} \tag{3.1}$$

where L_{box} represents the bounding box loss, L_{cls} represents the classification loss and L_{dfl} represents the distribution focal loss (DFL). With this, each of the three heads adopt L_{detect} for its loss function L_{OCR} , L_{LP} and L_{VCR} , respectively. Finally, the total loss function can be expressed as the weighted sum of each individual loss as in Equation (3.2):

$$L_{total} = \sum_i w_i L_i \tag{3.2}$$

where the two constraints for the weightage w_i of respective loss are: $w_i > 0$ and $\sum_i w_i = 1$, $i \in \{OCR, LP, VCR\}$.

As described in Section 3.4.1, the multi-task model is prone to overfitting the VCR task due to the class imbalance issue. To address this issue, the loss weight for VCR is set to 0.1, effectively reducing the significance of VCR task during training to overcome the risk of overfitting. OCR is prioritised by allocating higher attention to it. Through empirical testing, the weightage for OCR, LP detection and VCR is determined to be 0.6, 0.3 and 0.1, respectively. The multi-task loss function during fine-tuning is represented by Equation (3.3):

$$L_{total} = 0.6L_{OCR} + 0.3L_{LP} + 0.1L_{VCR} \quad (3.3)$$

3.4.4 Augmentation Settings and Hyperparameters

To improve the robustness and performance of the proposed multi-task YOLO model, data augmentation techniques are applied for better model generalisation. Table 3.2 tabulates the configuration of each augmentation setting.

Table 3.2: Augmentation Settings

Argument	Type	Value	Range
hsv_h	float	0.015	0.0 - 1.0
hsv_s	float	0.7	0.0 - 1.0
hsv_v	float	0.4	0.0 - 1.0
degrees	float	0	-180 - +180
translate	float	0.1	0.0 - 1.0
scale	float	0.5	>=0.0
shear	float	0	-180 - +180
perspective	float	0	0.1 - 0.001
flipud	float	0	0.0 - 1.0
fliplr	float	0.5	0.0 - 1.0
bgr	float	0	0.0 - 1.0
mosaic	float	1	0.0 - 1.0
mixup	float	0	0.0 - 1.0
copy_paste	float	0	0.0 - 1.0
auto_augment	str	randaugment	-
erasing	float	0.4	0.0 - 0.9
crop_fraction	float	1	0.1 - 1.0

3.5 Performance Evaluation

The performance for each task is evaluated using mean Average Precision (mAP), which is the average of Average Precision (AP) across all classes in a model. On the other hand, AP is derived from precision (P) and recall (R). The formula for precision and recall are shown in Equation (3.4) and (3.5) respectively.

$$P = \frac{TP}{TP+FP} \tag{3.4}$$

$$R = \frac{TP}{TP+FN} \tag{3.5}$$

where TP, FP and FN, represents true positive, false positive and false negative count respectively.

Given that AP is the area under the PR curve, AP can be represented by Equation (3.6) shown below:

$$AP = \int_{R=0}^{1} P(R)dR \tag{3.6}$$

Finally, mAP is simply obtained by averaging AP across all classes as in Equation (3.7).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{3.7}$$

where N is the total number of classes for each task-specific head.

Overall, the multi-task model will be evaluated using frames per second (FPS), providing an insight on the speed of the system in real-time applications. Equation (3.8) outlines the formula for FPS.

$$\text{FPS} = \frac{\text{Number of frames}}{\sum \text{Inference time of each frame}} \tag{3.8}$$

The performance evaluation of the proposed model consists of few parts as detailed below:

- mAP of OCR, LP detection and VCR tasks on different model configurations
- FPS of different model configurations trained
- Number of parameters and floating-point operations per second (FLOPs) of different model configurations

3.6 Gantt Chart

Table 3.3: Gantt Chart

		Final Year Project 1														Final Year Project 2													
No.	**Project Activities**	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**
1	Understanding the objectives and problem statement of the topic	🟨	🟨																										
2	Study the current achievement of car plate recognition system		🟨	🟨																									
3	Study the fundamentals of multi-task learning		🟨	🟨	🟨																								
4	Research on optical character recognition				🟨	🟨	🟨																						
5	Research on vehicle colour recognition						🟨	🟨																					
6	Research on possible implementations of multi-task learning							🟨	🟨	🟨	🟨																		
7	Progress report writing										🟨	🟨	🟨	🟨	🟨														
8	Data collection and annotation															🟨	🟨	🟨	🟨										
9	Develop the basis of multi-task car plate recognition system																		🟨	🟨	🟨								

10	Model training and testing															■						🟨	🟨							
11	Research and generate synthetic data															■							🟨	🟨	🟨					
12	Fine tune the car plate recognition system															■									🟨	🟨	🟨			
13	Car plate recognition system performance evaluation															■											🟨	🟨		
14	Final report writing															■											🟨	🟨	🟨	🟨

3.7 Summary

A one-stage YOLO-based multi-task model is designed to address ALPR system by unifying OCR, LP detection and VCR within a single framework. The multi-task model is first pretrained on synthetic data and fine-tuned on real data, leveraging transfer learning technique for OCR task. To address data imbalance issue during pretraining phase, weighted sum approach is adopted for multi-task optimisation. Extensive experiments will be conducted to compare the performance of the proposed model with existing state-of-the-art methods in terms of both accuracy and inference speed.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

Synthetic LP generation is done strictly according to the Malaysia vehicle LP rules with data augmentation to mimic the real-world scenarios. Using the synthetic dataset as pretraining data allows the multi-task model to quickly adapt and learn the LP layout as well as the familiarising with OCR task. Subsequently, the model is fine-tuned with real data to adapt with the real-life conditions. The performance of proposed multi-task model is predominantly evaluated using mAP and FPS. To elevate model performance across all tasks, extensive hyperparameter tuning is conducted. Notably, empirical testing demonstrates that the multi-task model outperforms single-head model in overall.

4.2 Synthetic Dataset

4.2.1 Malaysia Vehicle Registration Plate Rules

Throughout the synthetic data generation process, many factors are considered to meet the Malaysia LP specifications and replicate real-world scenarios. Common Peninsular Malaysia LPs follow a $\underline{X}$XX YYYY format, where $\underline{X}$ represents the state or territory prefix, X represents the alphabetical sequences and Y represents the number sequence. East Malaysia LPs are slightly different, in which the second character represents the division prefix. This algorithm is altered into $\underline{X}$ YYYY X format on exhaustion, further allowing a vast number of registered plates in specific state or territory. Notably, special plates are introduced during special occasions, replacing the leading alphabets such as LIMO, PATR1OT and PUTRAJAYA. With the exception of taxis, vehicle dealers and diplomats, all Malaysia LPs have white alphanumeric characters on black plate for both front and rear plates, regardless of the vehicle type (Anon., 2024b). However, several exceptions are present in the algorithm as follows:

- Leading zeros are prohibited in the number sequence.

- The letters I and O are excluded from the alphabetical sequences due to their similarities with the numbers 1 and 0.
- The letter Z is excluded and reserved for Malaysian military vehicles.

In the context of this research, synthetic LPs with X YYYY X format and special plates are not generated as each alphanumeric character is treated as an object detection problem. For robust OCR task, letter Z is included during data generation process to accommodate special plates like NAZA. Additionally, LPs with red, blue and white background for taxis, vehicle dealers and diplomats respectively are omitted due to their rare occurrence. Notably, the fonts for normal and special LPs are Franklin Gothic Bold and Calisto MT Italic respectively according to regulations (Anon., 2024b). However, in reality, different fonts are used for normal LPs, including Arial Bold, Bebas Neue, Grand Junction, Helvetica Now Micro and Palo Compressed Medium, which is illegal (Asaad, Faizabadi and Mohd Zaki, 2023). For close resemblance of the real-world LPs, all these unauthorised fonts are included in the synthetic dataset generation. Figure 4.1 shows some samples of synthetic images with different fonts.

4.2.2 Synthetic License Plate Generation

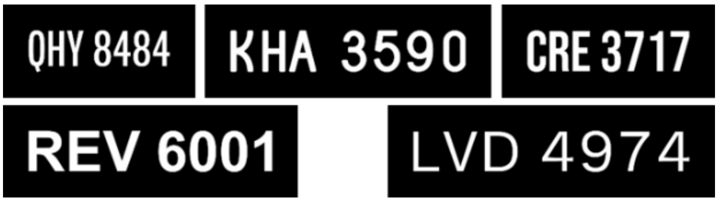
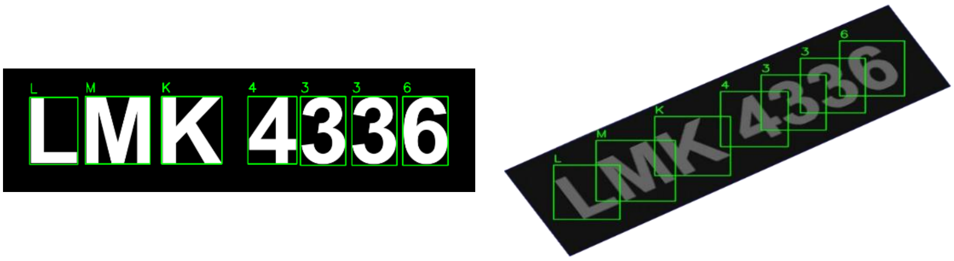


Figure 4.1: Samples of Synthetic Images



(a) (b)

Figure 4.2: (a) Annotated LP before Augmentation (b) Annotated LP after Augmentation

Data annotation process is typically tedious and time-consuming, where annotators have to label large quantities of quality images for good model generalisation. Moreover, human error poses a challenge in data annotation due to mislabelled data and inconsistent annotation, leading to decreased model performance. To automate data annotation process, cv2 library is utilised for contours detection and bounding box generation of all characters on the synthetic image. The synthetic images are ideal and non-realistic to replicate the real-world LPs for further model training. Hence, data augmentation techniques are applied, generating realistic-looking LPs. Finally, these images together with the bounding boxes, are skewed according to the LP's angle in original images and replaced, creating synthetic dataset with modified LPs. Figure 4.2 shows the annotated LP before and after augmentation.

4.3 Model Training Setup

Before training the model, transfer learning technique is applied by initialising the model with pretrained Common Objects in Context (COCO) weights to enhance its training process. Due to data scarcity, the model is pretrained on synthetic dataset to warm up the model for better convergence. After pretraining with synthetic dataset, the model is fine-tuned on real dataset. Notably, the VCR head is frozen during fine-tuning to prevent overfitting, as VCR image-label pairs are not replaced during synthetic data generation. A consistent batch size of 12 is used throughout model training. It is observed that epochs are a tricky hyperparameter for each model configuration.

Determining the appropriate number of epochs poses a challenge for each model configuration. Hence, the best epochs discovered for each model configuration are reported in the result tabulations. All models in this work are trained on the NVIDIA GeForce RTX 1080 Ti Graphic Cards. Table 4.1 shows the complete experimental platform configurations, while Table 4.2 presents the library versions used in the codebase development.

Table 4.1: Experimental Platform Configurations

Names	Configuration
Operating System	Ubuntu 18.04.6
CPU	Intel Core i7-10710U CPU @ 1.10GHz
RAM (GB)	64
GPU	NVIDIA GeForce RTX 1080 Ti
GPU Acceleration Library	CUDA11.4

Table 4.2: Library Versions Used in the Codebase Development

Libraries	Versions
ultralytics	8.0.105
torch	1.13.1+cu117
openvino	2023.1.0
scipy	1.10.1
opencv-python	4.9.0.80
trdg	1.8.0
imutils	0.5.4

4.4 Hyperparameter Tuning

According to Table 4.3, higher image resolution significantly enhances model performance. Specifically, increasing the image resolution ensures a mAP improvement for OCR ranging from 17% to 44%, depending on the model configurations. This improvement is rationalised by the fact that OCR involves detecting small characters in the input image, which is challenging in low-resolution images. Meanwhile, the significance of pretraining model on synthetic data is demonstrated in Table 4.4, where the model accuracy improves. For instance, the mAP for VCR significantly improves as the number of synthetic datasets increases, going from one set to five sets. However, the number of synthetic datasets is capped at five sets as no significant performance improvement is observed beyond 5 sets. With these observations, the ablation study presented in Table 4.5 utilises 960 image resolution with models pretrained on five sets of synthetic data as base models.

Table 4.3: Image Size Effect on Model Accuracy

Model Configuration	**Image Size**	**mAP50**			**Epochs**
		OCR	**LP**	**VCR**	
1 Head, 1 Task					
OCR (Pretrained)	640	0.653	-	-	20+80
	960	0.819	-	-	20+80
LP	640	-	0.956	-	80
	960	-	0.956	-	80
VCR	640	-	-	0.716	80
	960	-	-	0.708	80
1 Head, 3 Tasks (Pretrained)	640	0.550	0.872	0.890	80+80
	960	0.791	0.874	0.891	80+80
3 Head, 3 Tasks (Pretrained)	640	0.628	0.963	0.877	80+80
	960	0.735	0.965	0.862	20+40

Table 4.4: Synthetic Data Effect on Model Accuracy

Model Configuration	**Sets of Pretrained Synthetic Data**	**mAP50**			**Epochs**
		OCR	**LP**	**VCR**	
1 Head, 3 Tasks	1	0.772	0.763	0.793	80+80
	5	0.791	0.874	0.891	80+80
3 Head, 3 Tasks	1	0.781	0.962	0.76	80+80
	5	0.735	0.965	0.862	20+40

Table 4.5: Ablation Study. Shaded Indicates the Optimal Model for All Tasks

	Model Configuration	**mAP50**				**Epochs**
		OCR	**LP**	**VCR**	**Average**	
	1 Head, 1 Task					

	Model Configuration	**mAP50**				**Epochs**
		OCR	**LP**	**VCR**	**Average**	
A	OCR (Pretrained)	0.819	-	-	0.828	20+80
B	LP	-	0.956	-		80
C	VCR	-	-	0.708		80
D	**1 Head, 3 Tasks (Pretrained)**	0.791	0.874	0.891	0.852	80+80
E	**3 Head, 3 Tasks (Pretrained)**	0.735	0.965	0.862	0.854	20+40
E1	E with disabled mosaic	0.755	0.966	0.817	0.846	20+40
E2	E with frozen VCR head, 5:4:1 loss weightage and disabled mosaic	0.755	0.958	0.849	0.854	20+40
E3	E with frozen VCR head, 6:3:1 loss weightage and disabled mosaic	0.778	0.963	0.881	0.874	20+40
E4	E with frozen VCR head, 7:2:1 loss weightage and disabled mosaic	0.764	0.953	0.87	0.862	20+40

In the ablation study to access the effectiveness of the proposed solution, conventional methods represented by Models A, B, C, and D are included as the performance baseline. Models A, B, and C serve as baselines, with each model individually fine-tuned for specific tasks. In contrast, Model D adopts a naive approach, consolidating all tasks under a single head. Interestingly, the proposed solution (Model E3) is comparable with the single-head YOLO model trained for both single and multi-task. Model E3 demonstrates a competitive edge by achieving improvements within a deviation of 24.4% and 10.2% from single-head single-task models (Model A, B and C) and single-head multi-tasks model (Model D), respectively. Model E3 is flexible enough to simultaneously optimise for OCR, LP, and VCR, with the highest average mAP of 0.874 among all model configurations. This underscores model E3's ability to achieve a delicate balance between model complexity and generalisation.

The dedicated loss design in Equation (3.3) is demonstrated to well-suit the multi-task model. Across models E1 to E4, best performance is achieved when the weights for L_{OCR}, L_{LP} and L_{VCR} are adjusted to be 0.6, 0.3 and 0.1, respectively. Any other ratio would degrade the overall performance. Meanwhile, mosaic data augmentation is disabled since it is primarily beneficial for large objects. Disabling mosaic augmentation slightly improves OCR and LP detection while adversely affecting VCR performance, as shown in the transition from model E to model E1. However, this issue can be compensated through the dedicated loss function, as demonstrated by the increase in mAP for VCR in model E3. Once again, this underscores the advantage of the custom loss function.

4.5 Output Demonstration

This section demonstrates the output of each head from the multi-task model, including OCR, LP detection and VCR head, which is shown in Figure 4.3, Figure 4.4 and Figure 4.5 respectively.

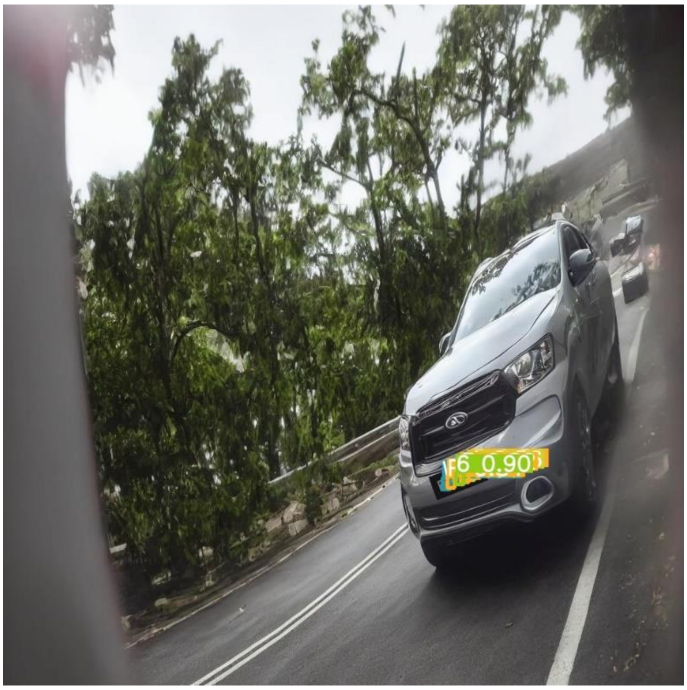


Figure 4.3: OCR Head Output

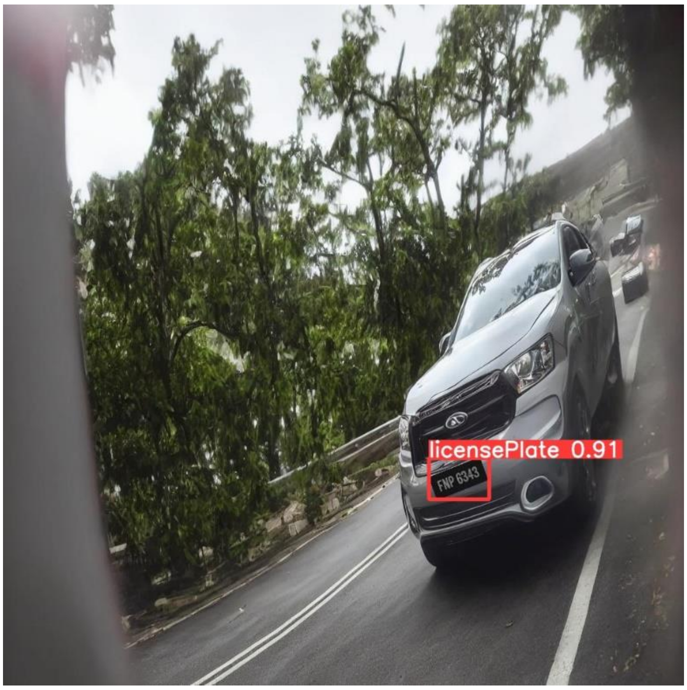


Figure 4.4: LP Detection Head Output

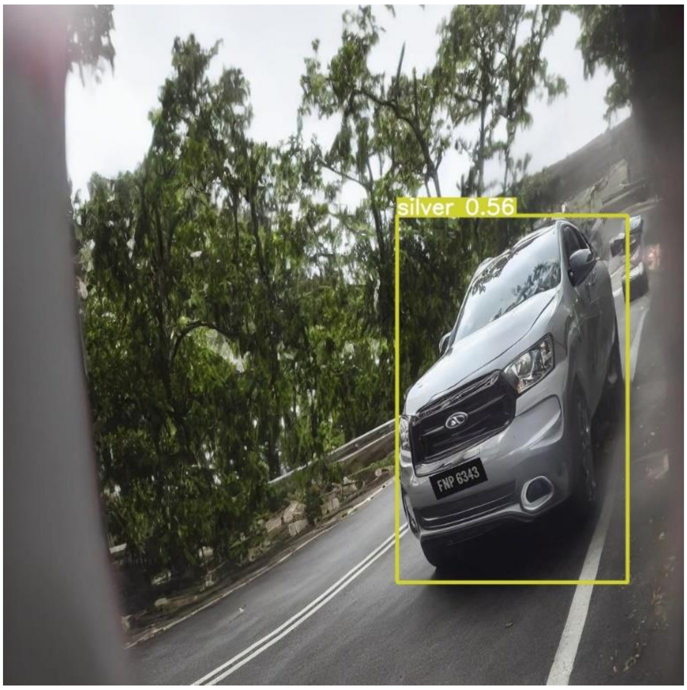


Figure 4.5: VCR Head Output

4.6 Performance Evaluation

Based on Table 4.5, the proposed model achieves the highest average mAP of 0.874 among all models, with mAP for OCR, LP detection and VCR as 0.778, 0.963, and 0.881 respectively. Notably, the proposed model is implemented for real-time applications, in which inference speed is the key evaluation metric. Table 4.6 compares the FPS of different model configurations. The conventional approach involves sequential execution of each single-head single-task YOLO, as shown in setup: Model A + B + C. The inference time is significantly slow because sequential processing imposes a cumulative time overhead, hindering real-time applications. On the other hand, the naive approach of using single-head for multi-tasking (Model D) exhibits the fastest inference time. However, this efficiency comes at the cost of compromised accuracy, indicating a potential trade-off between speed and accuracy. In contrast, the multi-head multi-task Model E3 has a higher inference speed than the conventional sequential execution approach. Although Model E3 has a lower FPS than the naive multi-task model D, it has the highest average mAP among all model configurations. This strikes a balance between efficiency and prediction, surpassing existing single-task and multi-task model configurations.

Recognising the role of multithreading in model inference process, Model E3 is deployed in a multithreaded manner, as described by Wong (2024). Table 4.7 compares the FPS of two video sources with both single- and multi-stream threading configuration. Multithreading significantly enhances the model inference speed by 45.69% and 69.05% on single-stream prerecorded video setting with GPU and CPU respectively. Notably, multi-stream setting is slightly slower than single-stream as several frames are processed concurrently, consuming additional computational power. In terms of video source, Real-Time Streaming Protocol (RTSP) has slower inference speed where video is streamed over a server rather than obtained locally. However, RTSP has flexible configurations in how the video stream is managed and delivered to cater specific needs and further optimisation. Hence, it remains as the *de facto* standard for closed-circuit television (CCTV), in which the multi-task model will be implemented in the future.

Table 4.6: FPS of Different Models

Model Configuration	**FPS (on GPU)**	**FPS (on CPU)**	**mAP50 (average)**
Model A + B + C (sequential execution)	18.868	5.376	0.828

Model Configuration	FPS (on GPU)	FPS (on CPU)	mAP50 (average)
Model D	53.452	16.522	0.852
Model E3	36.225	9.510	0.874

Table 4.7: FPS of Model E3 on Different Sources and Multithreading Configurations

Source	Multithread	FPS (on GPU)	FPS (on CPU)
Prerecorded video	Single-stream	52.7780	16.0770
	Multi-stream	40.6866	8.9300
RTSP	Single-stream	23.1716	9.3468
	Multi-stream	22.4547	2.6978

Table 4.8 compares the parameter size and FLOPs between conventional and proposed methods. Undoubtedly, the sequential execution of Model A, B and C exhibits the highest number of parameters and FLOPs due to the combination of three models. Meanwhile, Model D, as a single-head single-model setup showcases the lowest number of parameters and FLOPs. Although Model E3 utilises higher number of parameters and FLOPs than the naive multi-task Model D, it is still lightweight compared to the conventional sequential execution approach.

Table 4.8: Number of Parameters and FLOPs of Different Models

Model Configuration	Number of Parameters (M)	FLOPs (GFLOPs)
Model A + B + C (sequential execution)	9.041	24.623
Model D	3.019	8.237
Model E3	6.495	18.175

Figure 4.6 illustrates an example of ALPR using the proposed multi-task model. Overall, the model performs well to detect the vehicle colour and license plate. Minor error is observed on OCR where the model recognises as “8ABK138” rather than “ABK1388”. Similar and repeated alphanumeric characters confuse the model, resulting in an additional “8” in front of “A” and

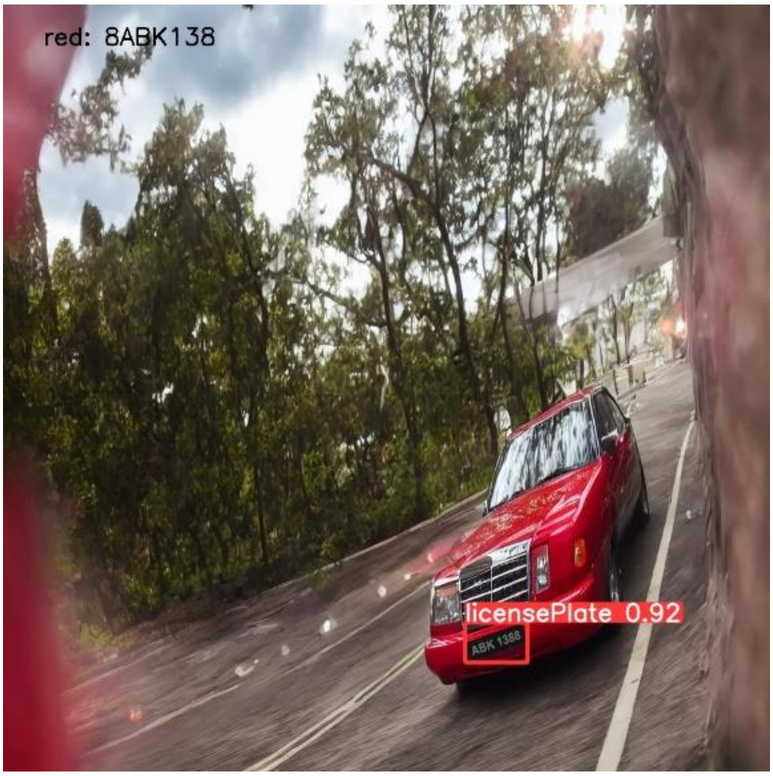


Figure 4.6: Failure OCR Case

missing “8” at the end, especially when the characters are small. Considering this issue, geofencing is empirically adopted for the model to perform OCR only when the license plate is near to the camera. For real-time application, license plate tracking is applied, maintaining unique ID for each detected license plate and subsequent OCR task. On the other hand, the performance of VCR is evaluated using the Precision-Recall (PR) Curve as shown in Figure 4.7.

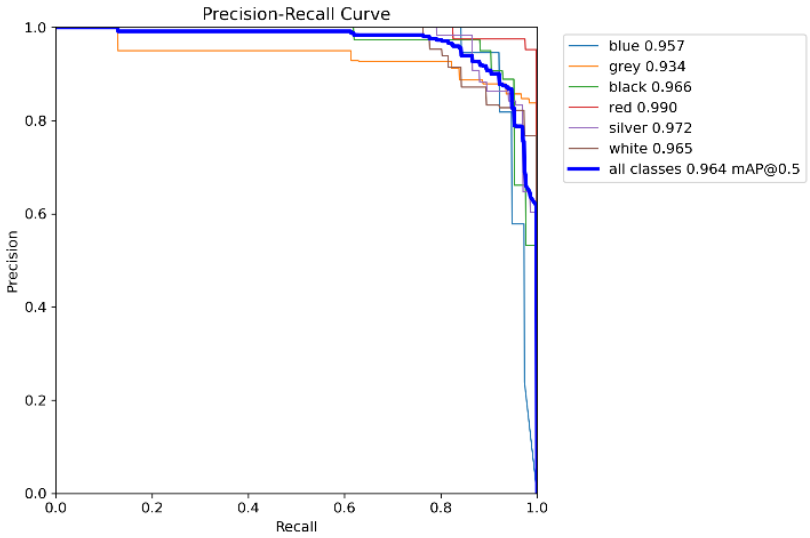


Figure 4.7: PR Curve for VCR Head

4.7 Summary

The proposed multi-task model achieves comparable performance with both single-head single-task and single-head multi-task model. Initially, this project leverages transfer learning technique with synthetic data to improve OCR task due to its small and diverse characteristics. For further model optimisation, extensive hyperparameter tuning is conducted involving number of epochs, weighted loss function and data augmentation. As such, the proposed model achieves the highest average mAP among all models. Although this model incorporates additional heads onto the backbone, it is still lightweight with comparable FPS for real-world applications.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

In conclusion, a novel multi-task YOLOv8 model is designed for OCR, LP detection and VCR. The model demonstrated exceptional performance, achieving high mAP scores of 0.778, 0.963, and 0.881 for OCR, LP detection, and VCR tasks, respectively. It outperforms conventional two-stage systems and strikes a balance between accuracy and inference speed for single-head models, making it suitable for real-world applications. Given the multi-head nature of the proposed model, it is still lightweight for execution on edge devices.

Despite the challenge posed by a limited dataset comprising only 1555 images, various strategies are employed, including pretraining on synthetic data and hyperparameter tuning, to enhance the model’s performance. Notably, OCR tasks exhibit slightly lower accuracy compared to LP detection and VCR tasks due to the small and diverse alphanumeric characters. To address this issue, geofencing is adopted to perform OCR only when the license plate is within a specified region, enhancing OCR performance.

This research contributes to the advancement of ALPR systems, offering a comprehensive solution for vehicle identification that can be beneficial for law enforcement and security surveillance.

5.2 Recommendations for Future Work

In future research, the proposed model can be further optimised by incorporating OpenVINO. OpenVINO, known for accelerating and deploying deep learning models for efficient execution on edge devices, presents a promising avenue for improving the model’s efficiency. Furthermore, the model’s capabilities will be extended to recognise car make and model, providing a more comprehensive understanding of the observed vehicles. Additional dataset will be curated by leveraging pseudo-labelling approach with current model, reducing human intervention and enhancing model performance simultaneously. Specifically, dataset distribution should be taken into account, in which real-life conditions and geographical distribution the model will be deployed. The optimised model holds significant potential for deployment in real-world scenarios, such as residential areas or shopping malls, contributing to advanced ALPR systems.

REFERENCES

Agrawal, A., 2023. Autonomous Self Driving System Using Deep Learning Appraoches and Impact. *Proceedings of the International Conference on Circuit Power and Computing Technologies, ICCPCT 2023*, pp.961–970. https://doi.org/10.1109/ICCPCT58313.2023.10245567.

Al-batat, R., Angelopoulou, A., Premkumar, S., Hemanth, J. and Kapetanios, E., 2022. An End-to-End Automated License Plate Recognition System Using YOLO Based Vehicle and License Plate Detection with Vehicle Classification. *Sensors 2022, Vol. 22, Page 9477*, [online] 22(23), p.9477. https://doi.org/10.3390/S22239477.

Aminu, A., 2023. *Google Colab vs Jupyter Notebook: Software Comparison*. [online] Available at: <https://www.techrepublic.com/article/google-colab-vs-jupyter-notebook/> [Accessed 29 March 2024].

Anon. 2021. *Intel® NUC Products NUC10i3FN/NUC10i5FN/ NUC10i7FN Technical Product Specification*. Intel.

Anon. 2024. *About - OpenCV*. [online] OpenCV. Available at: <https://opencv.org/about/> [Accessed 29 March 2024].

Anon. 2024a. *eGPU Breakaway Box 750/750ex*. [online] Sonnet. Available at: <https://www.sonnettech.com/product/egpu-breakaway-box/overview.html> [Accessed 29 March 2024].

Anon. 2024b. *JPJ Car Plate Specifications*. [online] CYC Malaysia. Available at: <https://www.bilton.my/pages/jpj-car-plate-specifications> [Accessed 24 April 2024].

Asaad, A., Faizabadi, A.R. and Mohd Zaki, H.F., 2023. Synthetic License Plate Generation: A Novel Approach For Effective License Plate Recognition In Malaysia. *International Conference on Engineering Technologies and Applied Sciences: Shaping the Future of Technology through Smart Computing and Engineering, ICETAS 2023*. https://doi.org/10.1109/ICETAS59148.2023.10346262.

Awalgaonkar, N., Bartakke, P. and Chaugule, R., 2021. Automatic License Plate Recognition System Using SSD. *2021 International Symposium of Asian Control Association on Intelligent Robotics and Industrial Automation, IRIA 2021*, pp.394–399. https://doi.org/10.1109/IRIA53009.2021.9588707.

Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J. and Lee, H., 2019. What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. *Proceedings of the IEEE International Conference on Computer Vision*, [online] 2019-October, pp.4714–4722. https://doi.org/10.1109/ICCV.2019.00481.

Belval E, 2020. *TextRecognitionDataGenerator: A synthetic data generator for text recognition*. [online] Available at: <https://github.com/Belval/TextRecognitionDataGenerator> [Accessed 17 March 2024].

Chen, P., Bai, X. and Liu, W., 2014. Vehicle color recognition on urban road by feature context. *IEEE Transactions on Intelligent Transportation Systems*, 15(5), pp.2340–2346. https://doi.org/10.1109/TITS.2014.2308897.

Chen, Y., Yu, J., Zhao, Y., Chen, J. and Du, X., 2022. Task's Choice: Pruning-Based Feature Sharing (PBFS) for Multi-Task Learning. *Entropy 2022, Vol. 24, Page 432*, [online] 24(3), p.432. https://doi.org/10.3390/E24030432.

Chen, Z., Badrinarayanan, V., Lee, C.-Y. and Rabinovich, A., 2018. *GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks*. Available at: <https://proceedings.mlr.press/v80/chen18a.html> [Accessed 1 April 2024].

Crawshaw, M., 2020. Multi-Task Learning with Deep Neural Networks: A Survey. [online] Available at: <https://arxiv.org/abs/2009.09796v1> [Accessed 18 March 2024].

Girshick, R., 2015. Fast R-CNN. [online] Available at: <http://arxiv.org/abs/1504.08083>.

Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2013. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, [online] pp.580–587. https://doi.org/10.1109/CVPR.2014.81.

Gong, T., Lee, T., Stephenson, C., Renduchintala, V., Padhy, S., Ndirango, A., Keskin, G. and Elibol, O.H., 2019. A Comparison of Loss Weighting Strategies for Multi task Learning in Deep Neural Networks. *IEEE Access*, 7, pp.141627–141632. https://doi.org/10.1109/ACCESS.2019.2943604.

Han, B.G., Lee, J.T., Lim, K.T. and Choi, D.H., 2020. License Plate Image Generation using Generative Adversarial Networks for End-To-End License Plate Character Recognition from a Small Set of Real Images. *Applied Sciences 2020, Vol. 10, Page 2780*, [online] 10(8), p.2780. https://doi.org/10.3390/APP10082780.

Henry, C., Ahn, S.Y. and Lee, S.W., 2020. Multinational License Plate Recognition Using Generalized Character Sequence Detection. *IEEE Access*, [online] 8, pp.35185–35199. https://doi.org/10.1109/ACCESS.2020.2974973.

Huang, Q., Cai, Z. and Lan, T., 2021. A Single Neural Network for Mixed Style License Plate Detection and Recognition. *IEEE Access*, 9, pp.21777–21785. https://doi.org/10.1109/ACCESS.2021.3055243.

Jocher, G., Chaurasia, A. and Qiu, J., 2023. *Ultralytics YOLO*. Available at: <https://github.com/ultralytics/ultralytics>.

Jye, S.-R., 2022. *What Is PyTorch?* [online] Built In. Available at: <https://builtin.com/machine-learning/pytorch> [Accessed 29 March 2024].

Laroca, R., Cardoso, E. V., Lucio, D.R., Estevam, V. and Menotti, D., 2022. On the Cross-dataset Generalization in License Plate Recognition. *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, [online] 5, pp.166–178. https://doi.org/10.5220/0010846800003124.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C., 2015. SSD: Single Shot MultiBox Detector. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, [online] 9905 LNCS, pp.21–37. https://doi.org/10.1007/978-3-319-46448-0_2.

Mao, R. and Li, X., 2021. Bridging Towers of Multi-task Learning with a Gating Mechanism for Aspect-based Sentiment Analysis and Sequential Metaphor Identification. *35th AAAI Conference on Artificial Intelligence, AAAI 2021*, 15, pp.13534–13542. https://doi.org/10.1609/AAAI.V35I15.17596.

Marzuki, P., Syafeeza, A.R., Wong, Y.C., Hamid, N.A., Nur Alisa, A. and Ibrahim, M.M., 2019. A design of license plate recognition system using convolutional neural network. *International Journal of Electrical and Computer Engineering*, 9(3), pp.2196–2204. https://doi.org/10.11591/IJECE.V9I3.PP2196-2204.

Mijwil, M.M., Aggarwal, K., Doshi, R., Hiran, K.K. and Gök, M., 2022. The Distinction between R-CNN and Fast R-CNN in Image Analysis: A Performance Comparison. *Asian Journal of Applied Sciences*, [online] 10(5), pp.2321–0893. https://doi.org/10.24203/AJAS.V10I5.7064.

Panetta, K., Kezebou, L., Oludare, V., Intriligator, J. and Agaian, S., 2021. Artificial Intelligence for Text-Based Vehicle Search, Recognition, and Continuous Localization in Traffic Videos. *AI 2021, Vol. 2, Pages 684-704*, [online] 2(4), pp.684–704. https://doi.org/10.3390/AI2040041.

Razalli, H., Ramli, R. and Alkawaz, M.H., 2020. Emergency Vehicle Recognition and Classification Method Using HSV Color Segmentation. *Proceedings - 2020 16th IEEE International Colloquium on Signal Processing and its Applications, CSPA 2020*, pp.284–289. https://doi.org/10.1109/CSPA48992.2020.9068695.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2015. You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, [online] 2016-December, pp.779–788. https://doi.org/10.1109/CVPR.2016.91.

Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, [online] 39(6), pp.1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.

Sayantini, D., 2019. *Deep Learning With Python - A Comprehensive Guide to Deep Learning*. [online] Medium. Available at: <https://medium.com/edureka/deep-learning-with-python-2adbf6e9437d> [Accessed 29 March 2024].

Shashirangana, J., Padmasiri, H., Meedeniya, D. and Perera, C., 2021. Automated license plate recognition: A survey on methods and techniques. *IEEE Access*, 9, pp.11203–11225. https://doi.org/10.1109/ACCESS.2020.3047929.

Sun, T., Shao, Y., Li, X., Liu, P., Yan, H., Qiu, X. and Huang, X., 2020. Learning Sparse Sharing Architectures for Multiple Tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 34(05), pp.8936–8943. https://doi.org/10.1609/AAAI.V34I05.6424.

Tariq, A., Khan, M.Z. and Ghani Khan, M.U., 2021. Real Time Vehicle Detection and Colour Recognition using tuned Features of Faster-RCNN. *2021 1st International Conference on Artificial Intelligence and Data Analytics, CAIDA 2021*, pp.262–267. https://doi.org/10.1109/CAIDA51941.2021.9425106.

Tham, M.L. and Tan, W.K., 2021. IoT Based License Plate Recognition System Using Deep Learning and OpenVINO. *ACM International Conference Proceeding Series*, [online] pp.7–14. https://doi.org/10.1145/3502814.3502816.

Tham, M.L., Wong, Y.J., Kwan, B.H., Ng, X.H. and Owada, Y., 2023. Artificial Intelligence of Things (AIoT) for Disaster Monitoring using Wireless Mesh Network. *ACM International Conference Proceeding Series*, pp.234–239. https://doi.org/10.1145/3584871.3584905.

Tham, M.L., Wong, Y.J., Kwan, B.H., Owada, Y., Sein, M.M. and Chang, Y.C., 2021. Joint Disaster Classification and Victim Detection using Multi-Task Learning. *2021 IEEE 12th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2021*, pp.407–412. https://doi.org/10.1109/UEMCON53757.2021.9666576.

Tu, C. and Du, S., 2022. A hierarchical RCNN for vehicle and vehicle license plate detection and recognition. *International Journal of Electrical and Computer Engineering (IJECE*, 12(1), pp.731–737. https://doi.org/10.11591/ijece.v12i1.pp731-737.

Vafaeikia, P., Namdar, K. and Khalvati, F., 2020. A Brief Review of Deep Multi-task Learning and Auxiliary Task Learning. [online] Available at: <https://arxiv.org/abs/2007.01126v1> [Accessed 20 April 2024].

Wang, C.-Y., Yeh, I.-H. and Liao, H.-Y.M., 2024. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. [online] Available at: <https://arxiv.org/abs/2402.13616v2> [Accessed 27 April 2024].

Wang, J., Wu, Q.M.J. and Zhang, N., 2023. You Only Look at Once for Real-time and Generic Multi-Task. [online] Available at: <https://arxiv.org/abs/2310.01641v3> [Accessed 19 March 2024].

Wong, Y.J., 2024. *Efficient YOLOv8 Inferencing using Multithreading*. https://doi.org/10.5281/zenodo.10792741.

Wong, Y.J., Huang Lee, K., Tham, M.L. and Kwan, B.H., 2023. Multi-Camera Face Detection and Recognition in Unconstrained Environment. *2023 IEEE World AI IoT Congress, AIIoT 2023*, pp.548–553. https://doi.org/10.1109/AIIOT58121.2023.10174362.

Wong, Y.J., Tham, M.L., Kwan, B.H., Gnanamuthu, E.M.A. and Owada, Y., 2022. An Optimized Multi-Task Learning Model for Disaster Classification and Victim Detection in Federated Learning Environments. *IEEE Access*, 10, pp.115930–115944. https://doi.org/10.1109/ACCESS.2022.3218655.

Wu, S., Zhang, H.R. and Ré, C., 2020. Understanding and Improving Information Transfer in Multi-Task Learning. *8th International Conference on Learning Representations, ICLR 2020*. [online] Available at: <https://arxiv.org/abs/2005.00944v1> [Accessed 18 March 2024].

Xin, D., Ghorbani, B., Gilmer, J., Garg, A. and Firat, O., 2022. Do Current Multi-Task Optimization Methods in Deep Learning Even Help? *Advances in Neural Information Processing Systems*, 35, pp.13597–13609.

Zhao, R., Liu, T., Xiao, J., Lun, D.P.K. and Lam, K.M., 2020. Deep Multi-task Learning for Facial Expression Recognition and Synthesis Based on Selective Feature Sharing. *Proceedings - International Conference on Pattern Recognition*, [online] pp.4412–4419. https://doi.org/10.1109/ICPR48806.2021.9413000.