Vehicle-Related Scene Understanding Using Deep Learning

Xiaoxu Liu

A thesis submitted to the Auckland University of Technology

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer and Information Sciences

2023

School of Engineering, Computer and Mathematical Sciences

Abstract

Given diverse and intricate nature of traffic scenes, it becomes imperative to comprehend the scene from multiple perspectives and dimensions. In scenes that entail hierarchical relationships and demand a comprehensive grasp of global context, the evaluation of deep learning models hinges on the capacity to handle high-level semantic representation and processing. The models with superior capabilities in understanding hierarchical relationships and excelling in global and local feature extraction are widely regarded as the ideal choices for addressing the challenges of traffic scene understanding.

In this thesis, we undertake a comprehensive exploration of vehicle-related scene understanding using deep learning, from multiple perspectives. Initially, we delve into semantic segmentation and vehicle tracking from a 2D viewpoint. Subsequently, we extend this analysis from 2D to 3D, estimate scene depth and inter-vehicle distances from 2D images for understanding the scene from a different perspective.

To enhance scene analysis, we investigate the fusion of pose and appearance as features. Additionally, we make efforts to improve the understanding of local and global features within the models. This involves restructuring the models through the incorporation of attention modules and Transformer, as well as replacing tracking algorithms and adding distance estimation vector.

Furthermore, this thesis integrates four distinct tasks: Scene segmentation, vehicle tracking, distance estimation, and depth estimation. These integrated approaches yield a more sophisticated and specific scene understanding, encompass not only a horizontal analysis from a 2D perspective but also a vertical understanding from a 3D perspective.

Keywords: Traffic scene understanding, deep learning, scene segmentation, vehicle tracking, distance estimation, depth estimation, attention module, Transformer.

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: Date: 01 August 2023

Acknowledgment

I would like to express my heartfelt gratitude to staff members in AUT for their exceptional academic guidance and support, especially my supervisors: Wei Qi Yan, Minh Nguyen and Nikola Kasabov. Their expertise has not only nurtured my efficient study habits and skills but also enriched my understanding through regular theoretical lectures.

I extend my thanks to the Auckland University of Technology for providing an outstanding educational platform and fostering a strong academic atmosphere on campus. The support from the student hub has been invaluable in assisting students with their academic and personal lives. Moreover, the facilities, including the library and laboratories, have played a crucial role in facilitating my studies and research for this thesis.

Finally, I want to acknowledge the unwavering encouragement and financial contribution from my family, especially Zack Wang, which has made it possible for me to pursue my education and life in New Zealand. Their support has been an invaluable source of motivation throughout my journey.

Xiaoxu Liu

Auckland, New Zealand

August 2023

Chapter 1 Introduction

This chapter includes discussion of the challenges and complexities involved in scene understanding tasks, including object tracking, semantic segmentation, depth estimation, and distance estimation and explanation of how deep learning has emerged as a powerful approach to tackle them. We will also descript the original contributions the research aims to make to the field of scene understanding using deep learning. Moreover, we will overview the novel architectures, methods, or approaches we plan to introduce or improve upon.

1.1 Background and Motivation

Intelligent surveillance has been marked as a prominent area of research within the field of artificial intelligence. As technology continues to evolve, intelligent surveillance has become more accessible and practical, leading to its extensive adoption on roads and in the automotive industry. By integrating various disciplines, including computer science, engineering, computer graphics, digital image processing, computer vision, and computational intelligence, intelligent surveillance has emerged as a crucial method for ensuring both public and private safety and security (Yan, 2019).

In the realm of achievements, the advent of autonomous vehicles stands as a significant manifestation of intelligent surveillance technology. The successful operation of autonomous driving relies on various essential attributes, and among them is semantic segmentation. This process is specifically designed to aid the “Central Processing Unit" for vehicles in comprehending its surroundings, enabling real-time analysis of each scene and determining the optimal course of action (Gupta et al., 2021).

Autonomous vehicles are required to undertake a diverse range of intricate semantic tasks beyond fundamental semantic understanding. These tasks encompass discerning the intricate relationships between objects in driving scenarios and predicting the behavior of pedestrians, vehicles, and scenes. Autonomous vehicles are use of complex algorithms and decision-making processes to navigate through traffic and handle various driving scene. Environmental changes, such as road construction, detours, or temporary obstacles, may require autonomous system reevaluate its planned route and make real-time decisions to ensure a smooth and safe journey. Hence, a profound comprehension of vehicle-related scene understanding becomes paramount for ensuring the safety of autonomous vehicles, optimizing decision-making processes, and facilitating seamless human-computer interaction (Yan, 2019).

A fundamental component of scene understanding for autonomous vehicles is spatial perception. Spatial perception provides the necessary information for vehicles to perceive its surroundings accurately, makes informed inferences about the scene, and ensures safe and reliable autonomous driving (Guo et al., 2023). Throughout spatial perception, from a 2D perspective, autonomous vehicles need understand the spatial relationships between different elements in the scene; also, the ability to perceive the spatial layout of the scene enables the vehicle to detect and recognize objects in its vicinity. This includes identifying road boundaries, lane markings, traffic lights, and obstacles (Perng et al., 2020; Mamun et al., 2022; Guo et al., 2022; Ci, Xu, Lin & Lu, 2022). From a 3D perspective, spatial perception assists the vehicle to estimate the distance and proximity of visual objects in the scene. This information is crucial for safe navigation, as it allows the vehicle to maintain a safe following distance from other vehicles or avoid collisions with obstacles (Alfred et al., 2023; Ming et al., 2021; Masoumian et al, 2022).

Scene understanding serves as a fundamental pillar of driverless technology, as vehicles can only make informed control decisions when they accurately and autonomously perceive the traffic and road scene environment. It has the ability to provide precise representations and comprehensive understanding of scenes with valuable knowledge of the surroundings, enabling the completion of various tasks in an effective and secure manner (Ignatious, 2023; Liu et al., 2019).

The information contained within video scenes is intricate, diverse, and complex. With the advent of deep learning, employing machine-based perception of scenes, there exists substantial potential to greatly improve the efficiency of video analysis. The recent advancements of deep learning have led to its widespread applications, particularly in computer vision, making it a primary contributor to the prevalence of such representations in this field (Liu, 2019).

Moreover, deep learning offers a significant advantage in the form of transfer learning, benefiting from publicly available datasets and pre-trained networks, which ease the training process for various traffic scenarios. In the context of vehicle-related scenes, deep neural networks aim to replicate high-level abstractions present in the data. Simultaneously, these networks encode the abstractions as robust representations to comprehend single and multiple objects, scenes in images and videos, and events in the video (Sarker et al., 2021; Guo et al., 2021; Zhang et al., 2020; Hu et al., 2020).

Currently, deep learning enables comprehensive scene understanding and explanation from five key dimensions. Firstly, deep neural network possesses the ability to identify the category of each pixel within the scene. Secondly, it can recognize specific regions within the environment, with a focus on boundary positions for enhanced learning. Moreover, deep learning neural network is not only capable of visual object classification but also proficient in recognizing various environmental scenarios, such as pedestrian, streets, intersections, parking lots, highways, and more. Furthermore, it can generate a complete description of events, encompassing left turns, right turns, overtaking, parking, and other relevant actions based on the features present in the image.

In contrast to typical machine learning models, deep neural networks have the capability to acquire more comprehensive semantic information about a scene. There are numerous deep learning-based tools to develop scene understanding from image data. The generated description includes detailed object classes, the names of renowned persons, and the relationships between objects within the scene. Due to its outstanding accuracy in output and high capacity for semantic learning, deep learning approach offers significant advantages in the field of scene interpretation.

In the context of vehicle tracking, deep neural networks exhibit greater invariance to geometric modifications, deformations, and variations in lighting compared to

conventional techniques. This robustness allows the deep learning approach to excel in the task of vehicle tracking.

Furthermore, deep learning models exhibit fault-tolerance, parallel processing capabilities, and self-learning capabilities. These advantageous traits make the model highly proficient in handling environmental information, replication challenges, and ambiguous inference rules. Moreover, it can effectively tackle difficulties arising from size changes and rotational deformations of vehicles. As a result, deep learning models are well-suited for addressing challenges related to autonomous vehicle tracking (Liu et al., 2022).

There are two methods for estimating depth in RGB images to understand scene: (1) Depth estimation from stereo images or videos: Current deep learning techniques can recover depth information from two or more perspectives (Zheng et al., 2022). This involves using two horizontally located sensors in a stereo imaging system. The images captured simultaneously by the cameras are analysed and compared using deep learning and pattern matching algorithms to determine the disparity and depth map. However, stereo-matching can be challenging, especially in cases of blurred or poorly lit scenes. (2) Depth estimation from monocular images: In this approach, depth information is calculated accurately by using a single perspective image. Deep learning and pattern matching algorithms are again employed to infer the depth information from a single RGB image (Lin, Dai &Van, 2020).

Both methods play significant roles in depth estimation for RGB images, each with its advantages and challenges. The use of deep learning and pattern matching algorithms has enabled considerable progress in accurately estimating depth from images, though there are still challenges in dealing with certain scenes, such as poor lighting conditions and blurred scenes.

Traditional visual depth estimation approaches rely heavily on multi-view scene information to recover scene depth from a two-dimensional image through triangle geometric correspondence (Li, Li, Zhao & Fan, 2022). The methods based on deep learning utilize CNNs to recreate scene depths, which has been a popular topic among academics. Traditional methods are computationally demanding and complicated. Convolutional neural network can be pre-trained by using image data and its accompanying benchmark depth data, enabling it to accomplish full-resolution end-to-end image depth estimate during the test.

Distance estimation can be achieved through various methods, with laser detecting and ranging being a prominent approach for obtaining distance information (Zalevsky et al., 2021). Laser-based distance measurement has gained significant interest in the development of Collision Warning Systems. However, LiDAR, while sophisticated, is costly and yields limited results, making it currently suitable only for testing automobiles.

Alternative methods for vehicle detection and distance measurement include ultrasound, infrared, and microwave radar (Aliew, 2022; Özcan et al., 2020). However, each of these approaches has its limitations. Ultrasound and infrared-based distance measures have restrictions, while microwave radar is sensitive to interference, leading to unreliable detection results. Moreover, these methods often struggle to distinguish between different detection targets.

Hardware-based tools like radar and infrared devices present challenges in terms of cost, integration with imaging devices, and limitations in measurement precision. As a cost-effective solution, effective research methods have proposed by discarding expensive distance measurement apparatus and instead inferring distance information from 2D video footage captured during vehicle detection (Mehtab et al., 2021). This approach offers a potential alternative to the hardware-based methods, but it also comes with its own set of challenges and considerations.

Deep learning-based ranging (distance estimation) holds significant potential for various applications that can be seamlessly integrated with existing approaches to yield superior results. By accurately calibrating the camera's internal and external parameters, it becomes possible to determine the distance to the vehicle ahead. This information can then be utilized to provide timely alerts about potential accidents, simulates how objects are projected onto the image plane based on camera parameters or use the visual projection model that based on their appearance and geometric properties to estimate the distance of objects.

In the realm of visual data-based ranging techniques, there are two main branches: Monocular camera-based ranging methods and binocular camera-based ranging methods. Monocular camera ranging relies on initially identifying the target by matching its image with known patterns or features in the scene. This initial identification is typically performed by using visual object detection or recognition algorithms, which can locate and classify objects of interest in the image. Once the target object is identified, the distance estimation is based on its apparent size in the image and the knowledge of the real-world size of the object or its category. This method assumes that the physical size of the object is known or can be estimated from prior knowledge.

Within monocular cameras, there are several approaches. The circumferential ranging method utilizes a fisheye lens, which can result in more extensive lens distortion (Bremer et al., 2023). However, using a fisheye lens can result in more extensive lens distortion compared to traditional rectilinear lenses. Lens distortion can affect the apparent size and shape of objects in the image, leading to inaccuracies in the perceived dimensions of objects. This impacts the accuracy of distance estimation based on apparent size.

Moreover, fisheye lens distortion is nonlinear and more complex than rectilinear lens distortion. Accurately calibrating the fisheye camera to correct for distortion

requires more sophisticated calibration methods and introduces additional computational overhead. Another approach is related to forward-looking camera ranging, characterized by reduced aberration in the front-view lens helps capture more accurate and undistorted images of the scene in front of the vehicle and the camera being mounted beneath the rearview mirror of the vehicle offers a relatively stable and vibration-free location for the camera, which improves the quality of the captured images (Karimanzira et al., 2021). The third approach is based on oblique cameras, which is distinguished by its larger angle of view (Fukushima, Farzad, & Torras, 2017; Cai et al., 2020). Each of these methods has its unique characteristics and applications in distance estimation by using monocular cameras.

In contrast to the distance measurement method used by the forward-facing cameras, the circumferential fisheye camera does not rely on mathematical geometry for distance ranging. The reason is that the lens faces downward, resulting in high aberration coefficients, making traditional geometric distance ranging prone to significant errors. Instead, the distance ranging concept for the circumferential fisheye camera is based on a single-strain matrix and affine transformation.

Binocular camera ranging utilizes a pair of cameras to perceive 3D structure of a scene. This method is inspired by human vision, where our brain makes use of information from both eyes to estimate depth and perceive the world in three dimensions. In a binocular camera system, two cameras are positioned side by side, mimicking the separation between human eyes. Each camera captures a slightly different view of the scene due to their horizontal displacement. The images from the two cameras are then employed to compute the disparity, which is the horizontal difference between corresponding points in the left and right images.

Binocular estimation offers a number of advantages. One notable benefit is that it doesn't require prior recognition of objects, allowing for an unlimited recognition rate. All obstacles can be directly evaluated without the need for pre-existing knowledge.

Moreover, binocular estimation doesn't rely on maintaining a sample database, as it operates without the concept of a sample.

On the other hand, monocular estimation also comes with its own set of advantages. It is a cost-effective solution, requiring less computational resources, making it more accessible for various applications. Additionally, it is relatively simple design which makes it easier to implement and deploy in practical scenes.

The goal of computer vision is to enable an intelligent agent to perceive and comprehend its surroundings in a manner which is similar to humans. Instead of using human eyes, computers control cameras to project the 3D environment onto 2D images. Despite the loss of 3D information in this projection, the resulting images remain a valuable source of information. The objective of visual scene understanding is to extract this information, facilitating the creation of accurate representations of the surrounding world. Computer vision can even utilize sensors that provide information beyond what the human eye can perceive, such as RGB-D sensors that offer direct transmission of 3D information.

Deep learning-based ranging holds significant potential for various applications and can be seamlessly integrated with the existing approaches to yield superior results. By accurately calibrating the internal and external parameters of cameras, it becomes possible to determine the distance to the vehicle ahead. This information can then be utilized to provide timely alerts about potential accidents, simulates how objects are projected onto image plane based on camera parameters or the visual projection model that based on their appearance and geometric properties to estimate the distance of visual objects, also known as 3D vision. Both 2D vision and 3D vision are viable sensors for scene understanding. 2D vision is particularly well-suited for object and pixel-level detection and identification. On the other hand, 3D vision provides 2.5D images, including depth, enabling its use for geometric tasks, in addition to semantic ones, such as scene completion.

The application of deep learning for autonomous vehicles faces challenges due to variations in traffic laws and transportation infrastructure across nations, making it difficult to fully interpret complex traffic objects and scenes. Computer vision aims to empower intelligent systems to perceive and understand their surroundings, with cameras serving as the eyes of computers, projecting the 3D environment onto 2D images. Despite the loss of 3D information in this process, the extracted information remains valuable for visual scene understanding and accurate representation of the environment.

In addition to traditional cameras, computer vision can benefit from sensors that provide information beyond human perception, such as RGB-D sensors that offer 2.5D information (image+depth) or 3D vision. Both 2D vision and 3D vision are viable for scene understanding. While 2D vision is primarily suitable for visual object and pixel-level detection and identification, 3D vision provides 2.5D images with depth, making it more suitable for geometric applications as well as semantic understanding.

Two key areas of focus in scene understanding are vehicle tracking and semantic segmentation of 2D scene components. However, it's important to note that a 2D image is merely a projection of a stereoscopic scene, capturing only planar information. To address this limitation, depth and distance estimation techniques aim to map 2D images to depth maps containing 3D information. Despite their same goals, few research studies currently combine these two areas effectively. As technology continues to advance, bridging the gap between 2D and 3D scene understanding holds significant potential for enhancing the capabilities of autonomous vehicles and computer vision systems.

The focus of this thesis is on using deep learning to significantly reduce human workload in semantic segmentation, vehicle tracking, distance estimation, and depth estimation for vehicle scene understanding. From a 2D perspective of scene

understanding, we introduce a novel type of capsule networks that incorporate posture and location information with matrix for semantic segmentation. This integration enhances the ability of higher-level semantic information, setting it apart from typical capsule networks.

For vehicle tracking, we enhance YOLOv5 by adding attention modules and the Transformer. Through employing the Hungarian algorithm, we improve the performance of our proposed YOLOv5-CBAM-Transformer, extending its capabilities from single-target tracking to multi-target tracking. While shifting to 3D vision, we are use of the Transformer as an encoder to gain better detail on visual object for depth estimation. Furthermore, we employ the attention mechanism and Transformer on YOLOv7 as well as extend the prediction vector to estimate the vehicle distance. Our proposed vehicle ranging model, YOLOv7-CBAM-Transformer, effectively improves the model's understanding of local and global features, thereby enhancing the performance of the original YOLO series models.

1.2 Research Questions

The segmentation of scene images, vehicle tracking, distance estimation, and depth estimation of such traffic scenes all contribute to the understanding of scenes involving vehicles. In consideration of this research premise, in this thesis, we primarily pose the following research questions:

Question:

"How to achieve 3D performance with 2D images using deep learning? What deep learning methods could be best implemented to vehicle-related scene understanding in the way of 2D and 3D vision?"

We can detail this generalized question to:

"In the process of understanding traffic scenes in 2D and 3D vision, how to make the model better understand the high-level semantic information in the scene?"

and

"How to improve the accuracy of our proposed models to achieve a robust and reliable deep learning model."

The main focus of this thesis is on the segmentation, tracking, range, and depth estimation of vehicle-related scenes. Consequently, it is essential to thoroughly assess the effectiveness of the strategies employed to comprehend such scenes. By conducting comprehensive performance comparisons of various algorithms, we aim to identify and adopt the most suitable and effective strategy for scene understanding in vehicle-related contexts.

1.3 Contributions

Currently, most computational approaches for semantic segmentation rely on convolutional neural networks (CNNs) as the primary method. However, in this thesis, we employed a capsule network architecture to realize semantic segmentation and curated a customized Auckland traffic dataset, which we recorded and annotated by ourselves. The capsule network possesses the capability to capture pose features and better comprehend high-level semantic information, making it a promising choice for improving the semantic segmentation performance in our specific dataset. By utilizing Capsule Networks (CapsNets) with the Vector-Space (VS) routing method, our model achieves faster convergence compared to the dynamic routing mechanism. During the experimental phase, we observed that our model outperforms U-Net and SegNet in terms of Intersection over Union (IoU) and segmentation results. In our own dataset, IoU has reached an impressive 74.61%. By achieving superior IoU and segmentation outcomes, our model demonstrates its potentiality to contribute significantly to the

safety and efficiency of autonomous vehicles.

For vehicle tracking, we proposed a multi-target identification, tracking, and scene understanding system that combines a regression modified SiamRPN with a modified YOLOv5 (YOLOv5-CBAM-Transformer). Unlike the existing single target tracking methods, our approach successfully tracks multiple targets using SiamRPN. We created a new publicly accessible benchmark dataset from our traffic scene and achieved faster tracking with assured performance. By incorporating Convolutional Block Attention Module (CBAM) to YOLOv5, we enhanced the local feature extraction ability, while the Transformer improved the global feature extraction ability. This results in accurate localization of detection frames and detection of distant objects. The combination of the SiamRPN model and Hungarian algorithm enables multiple object tracking.

We have also made significant advancements in enhancing the performance of deep learning-based models from a 3D perspective. In our proposed depth estimation model, we integrated Transformer encoder and CNN decoder as the model architecture, leading to the improved accuracy. Notably, this is the first instance of using Swin Transformer as encoder for depth estimation with the KITTI dataset. Through comparisons of the advanced existing CNN model DensDepth with our proposed Transformer-based model, we discovered that Transformer as the encoder resulted in enhanced depth estimation performance, particularly in the objects like street lights, road signs, and pedestrians. The Transformer compensates for the limitation of CNN, which tends to overly focus on local features., so that comprehensive global and local feature understanding.

Furthermore, we propose a low-cost distance estimation approach to develop more accurate predictions from a 3D perspective for vehicle detection and ranging by using inexpensive monocular cameras. Our distance estimation model integrates YOLOv7 with an attention module (CBAM) and Transformer as the fundamental architecture,

leading to improved high-level semantic understanding and enhanced feature extraction ability. This integration significantly improved object detection and ranging performance, offering a more suitable and cost-effective solution for distance estimation.

In summery, the primary contributions of this thesis are as follows. From 2D perspective:

- In scene segmentation, we are use of CapsNet and VS routing algorithm to resolve the black box nature faced by all convolutional neural networks. Using the appearance and pose features improves the ability to understand high-level semantics, and then realizes the optimization of segmentation performance.
- Our contributions in vehicle tracking involve the development of two models: YOLOv5-CBMA-Transformer and SiamRPN+Hungarian algorithm. These models address the challenge of detailed object detection by enhancing adaptability to object size. Additionally, the integration of Hungarian algorithm with SiamRPN enables the tracking of multiple targets efficiently.

From 3D perspective:

- In depth estimation, we combined Swin Transformer as encoder and CNN as decoder for obtaining better feature extraction ability to solve the problem of poor handling of detailed objects in depth estimation tasks.
- In distance estimation, our proposed YOLOv7-CBAM-Transformer combined with the extended prediction vector effectively achieves accurate ranging by improving the ability of local features and global features extraction and the adaptability of different size vehicles, thus solving the problem of estimation difficulties of vehicle distances without expensive

rangefinders.

1.4 Objectives of This Thesis

This thesis encompasses scene understanding across multiple dimensions by using deep learning methods. It implements scene segmentation, vehicle tracking, distance estimation, and depth estimation, in both 2D and 3D perspectives.

The main goal of this thesis is to create exceptional deep neural networks with advanced semantic understanding and excellent feature extraction capabilities. In order to achieve this objective, the following adjustments will be made to the neural network design: (1) Experimenting with various hyperparameters while training the model to discover the optimal hyperparameter group. (2) Exploring the impact of different neural networks and different module combinations on network performance. (3) Contrasting our proposed models with existing state-of-the-art models to identify their performance.

By implementing these adjustments, we aim to build highly effective models that excel in semantic understanding and feature extraction for various tasks within the domain of deep learning.

1.5 The Structure of This Thesis

In Chapter 2, we will focus on the existing literature, including contemporary approaches and models that have implemented image segmentation, vehicle tracking, distance estimation, and depth estimation in traffic scenes. We will comprehensively analyze and address the previous works, taking into account three key perspectives: The neural network, its structure, and the evaluation techniques used.

In Chapter 3 of this thesis, we will present the research methods from a mathematical theory. It will delve into the specifics of the research approach, including

image preprocessing, dataset collection and processing, and concepts related to model assessment techniques. We will provide a comprehensive understanding of the methodology to conduct the study and lay the foundation for the subsequent analyses and results.

In Chapter 4 of this thesis, we will provide details about experimental setup, including equipment, materials, and methodology in conducting the experiments. Also, the information on how data was collected during the experiments, including the types of data recorded, the sensors or instruments used, and the data collection process. Moreover, the methods to analyse the collected data and implement vehicle-related scene understanding. The presentation of our experimental findings and outcomes includes tables, graphs and figures that illustrate the data and support the conclusions drawn from the analysis.

In Chapter 5, we will thoroughly provide interpretation and discussion of the results obtained from the experiments in Chapter 4. In this chapter, we will compare the findings to the existing literature, explain any unexpected results, and provide insights into the implications of the results for the research.

In Chapter 6, an overview of the entire thesis will be presented, summarizing the key findings and contributions made. Additionally, a strategy for future study in the field will be outlined.

Chapter 2 Literature Review

The focus of this thesis is on advancing vehicle-related scene understanding in images, encompassing class-based segmentation of scenes, multiple objects-based vehicle tracking, vehicle detection and depth estimation using deep learning. To achieve these objectives, we will discuss the strengths and weaknesses of various approaches and provide insights into different scenes. In this chapter, we will extensively examine and evaluate the achievements in scene understanding over the past several years. This examination will provide valuable insights into the progress made in the field and pave the way for further advancements in vehicle-related scene understanding.

.

2.1 Introduction

The comprehension of traffic scenes has emerged as a prominent research area in computer vision and a focal point in artificial intelligence, particularly in light of the progress made in autonomous driving (Zhu et al., 2022; Xing et al., 2022). The existing scholarly literature delves into traffic scene understanding from diverse perspectives, reflecting the significance and widespread interest in this subject. Despite the popularity of these approaches, deep learning revolution has transformed the associated areas. As a result, a multitude of computer vision challenges, such as semantic segmentation, vehicle tracking, distance estimation, and depth estimation, are now being addressed through the utilization of deep neural networks, with a particular emphasis on convolutional neural networks. These advanced techniques have demonstrated remarkable potential in effectively tackling these complex vision tasks.

In terms of accuracy and even efficiency, convolutional neural networks surpass conventional approaches in scene understanding and other disciplines (Wang et al., 2023). Moreover, deep neural networks such as Capsule network and Transformer in recent years have also shown extraordinary capabilities in computer vision. Therefore, the method of combining convolutional neural network with other neural networks to improve the accuracy of the model has become the general trend in the field of deep learning.

The popular approaches for visual object recognition involve utilizing candidate areas and employing techniques like R-CNN, SPP-NET, Fast R-CNN, and Faster R-CNN, as well as regression theory with YOLO and SSD as notable examples (Müller & Dietmayer, 2018; Nandi et al., 2018; Park et al., 2019). To enhance the network structure and utilize spatial and channel information in feature maps effectively, the attention module is often incorporated into deep learning models. This integration allows for the fusion of low-level and high-level features, resulting in more accurate

detection capabilities.

Moreover, in recent years, Transformer architecture, based on the encoder-decoder structure and combined with the self-attention mechanism, has shown remarkable improvements in both accuracy and speed for various computer vision tasks (Xiao et al., 2023). Transformer has emerged as a powerful approach for further advancing visual object recognition capabilities.

The evolution of visual object recognition and tracking, from R-CNN to Faster R-CNN, has traditionally relied on a two-stage training process. While this approach improves accuracy, it slows visual object detection due to the increased factors. On the other hand, YOLO models adopt a single-shot grid segmentation approach, where each grid is responsible for recognizing the center, bounding box, and class label of visual objects simultaneously. This end-to-end training significantly enhances real-time capabilities, making YOLOv5 and YOLOv7 more efficient compared to YOLOv4, saving over 90% of time in the YOLO series (Müller & Dietmayr, 2018; Alexey et al., 2020).

Additionally, Transformer architecture, particularly Swin Transformer, has demonstrated strong capabilities in visual object detection, tracking, ranging and depth estimation in traffic scenes by improving global feature extraction ability. Swin Transformer combines the powerful modeling ability of Transformers with important visual signal priors, including hierarchy, locality, and translation invariance. The design of shifted non-overlapping windows in Swin Transformer reduces computational complexity, leading to faster speeds compared to traditional sliding windows (Liu et al., 2021).

To further enhance target detection and tracking in traffic scene understanding, attention mechanisms are incorporated into deep learning networks. The methods such as Squeeze-and-Excitation (SE) enable the model to focus on essential channel

information by learning adaptive channel weights (Hu et al., 2018). The CBAM combines convolution and attention mechanisms from both spatial and channel perspectives (Woo et al., 2018). Coordinate Attention (CA) considers both channel and spatial dimensions, allowing the model to emphasize crucial channel information through learned adaptive weights (Hou et al., 2021). These attention mechanisms contribute to improved local feature extraction abilities, resulting in more accurate and efficient performance of the models in traffic scenes (Peng et al., 2020).

In recent years, there has also been an abundance of scene segmentation-related research that understand scene in 2D way. Similarly, the depth estimation for understanding traffic scene by using 3D vision has also made several significant advances. To far, however, little research work has integrated scene segmentation, vehicle tracking, distance and depth assessment based on deep learning methods to understand scenes from both 2D and 3D aspects. The primary objective of the semantic segmentation algorithm is to build a way to comprehend the composition of a 2D picture of a traffic scene, including prediction and object recognition for the complete 2D scene image. In addition to providing classes, semantic segmentation gives extra information on the physical placements of those classes. Accuracy and performance in real-time are crucial indications for traffic scene understanding. Much like semantic segmentation, vehicle tracking is another task rooted in a 2D viewpoint. This process involves identifying a vehicle's present position within a 2D image and forecasting its trajectory. Through assigning distinct identifiers to each vehicle present in the image, the objective of vehicle tracking extends to identifying suspicious vehicle actions and monitoring possible risks (Tak et al., 2021).

In contrast, semantic segmentation and vehicle tracking do not evaluate the relationships between objects longitudinally. This indicates that semantic segmentation and vehicle tracking do not incorporate the front-to-back position connection of objects in the 3D environment in its spatial information. For depth and distance estimation, the

conversion of 2D RGB scene graphs to RBG-D images is thus seen as essential. Using RGB pictures from one or more viewing angles, depth estimation and distance estimation estimates the relative distance and specific distance between each pixel in an image and the shooting source. Due to the availability of a considerable quantity of previous knowledge, the human eye can extract a substantial amount of depth information from the eye's image data. Then, estimate of depth must not only learn objective depth information from two-dimensional pictures, but also extract empirical information that is more sensitive to the camera and scene in the dataset (Zhang et al., 2023).

In this thesis, we comprehensively explore vehicle-related scene understanding from both 2D and 3D perspectives by implementing deep learning-based techniques for scene segmentation, vehicle tracking, distance estimation, and depth estimation. Compared to traditional machine learning methods, deep learning models eliminate the need for explicit feature extraction, as it takes data directly at the input layer, enabling the network to achieve exceptional performance. Therefore, deep learning methods could be employed in a wider variety of fields and applications. In addition, we also study other deep learning networks besides the most popular CNN, such as CapsNet and Transformer. We believe it is necessary to make up for some shortcomings of CNN, such as black box characteristics and preference for understanding local features, by combining other networks with CNN, so as to further optimize the performance and representation of the network.

2.2 Deep Learning

Scene understanding involves the interpretation and comprehension of the content, arrangement, and context within an image or scene. Both traditional machine learning and deep learning methods have been employed for tasks related to scene understanding, but they diverge in their methodologies, capabilities, and performance.

Regarding feature extraction, traditional machine learning necessitates significant human involvement through feature engineering to produce outcomes. For instance, in a complex traffic scene image aiming for tasks like object detection and classification, contours of intricate entities like vehicles, pedestrians, and traffic signals must be manually extracted and assigned weights (Jabri et al., 2018; Dimililer et al., 2019). In contrast, deep learning, functioning as an end-to-end model, accomplishes feature engineering with minimal human intervention (Ammar et al., 2021; Hassaballah et al., 2020).

In the context of depth estimation, traditional machine learning approaches rely on controlled lighting setups and images captured from diverse angles to compute depth information, accounting for variations in lighting and shadows. However, due to their dependency on features and models crafted by hand, these methods might encounter challenges when attempting to achieve generalization across various scenes and lighting conditions.

On the other hand, deep learning models take a different approach. They can leverage extensive datasets containing annotated ground truth depth information to train their networks. This enables them to enhance their accuracy even in scenes with occlusions and adverse weather conditions. Remarkably, these models can even predict depth from a single image, a capability that has been demonstrated in recent studies (Ming et al., 2021; Zhao et al., 2020).

In terms of model performance, for vehicle tracking tasks, traditional machine learning models often rely on methods such as Haar cascades, Histogram of Oriented Gradients (HOG), or classifiers built around hand-crafted features like Support Vector Machines (SVM) for vehicle detection. Subsequently, they employ techniques like Kalman filters, Particle filters, and Mean Shift to estimate the vehicle's position over time based on motion models and sensor data measurements (Kumar, 2021; Kong et al., 2019).

On the other hand, deep learning models in vehicle tracking leverage networks such as Convolutional Neural Networks (CNNs) for understanding the vehicle scenes. In stark contrast to traditional machine learning approaches, deep learning models possess the capability to learn intricate spatial and temporal relationships. This equips them to adeptly manage challenges like occlusions, interactions between vehicles, and complex scenes (Li et al., 2020; Huang et al., 2020).

Furthermore, recurrent neural networks (RNNs) and long short-term memory (LSTM) networks of deep learning play a pivotal role in capturing the temporal dependencies of vehicle motion. This makes them especially well-suited for tasks with real-time requirements, including vehicle tracking (Abdallah, Han, & Kim, 2022; Jeong, Kim & Yi, 2020).

In tasks like semantic segmentation and depth estimation, which demand abundant pixel-level training data and entail substantial time and hardware expenses during training, deep learning presents an advantageous solution. By leveraging a limited set of annotated data and the knowledge acquired from different tasks via transfer learning, deep learning can skillfully fine-tune pre-trained models (Alhashim et al., 2018; Swaraja et al., 2021). This strategy leads to the development of high-performance models without necessitating the complete retraining process.

Supervised learning is widely employed for training deep learning networks. Supervised learning involves adjusting the model's parameters to achieve the desired performance by utilizing a collection of samples from a known class. This method infers a function based on labelled training data, which includes input objects paired with desired output values. The supervised learning method analyzes this training data to generate an inference, enabling the model to map new instances. The ultimate goal is to have the model accurately predict the correct labels for unknown cases, thereby achieving the desired performance (Sarker, 2021).

Essentially, supervised learning in deep learning is an effort to learn a mapping relationship from x to y given a collection of training samples (x_i, y_i), such that given an x, the output y is as near as possible to the real y, even if x is not present in the training samples. The loss function is used to estimate the difference between the output y and the real value y, hence guiding the optimisation. Empirical and structural risk are included in the model's structural risk, with the loss function playing a significant role in the empirical risk function (Song et al., 2018):

$$\hat{\theta} = arg\,\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \theta) + \lambda \Phi(\theta)) \tag{2.1}$$

where the mean function is empirical risk, $L(y_i, f(x_i; \theta))$ is loss function, $\lambda\Phi(\theta)$ is structural risk, the $\Phi(\theta)$ measures the complexity of the model.

The L_1,L_2, and BerHu loss functions are frequent regression problem loss functions. The mean absolute error minimizes the sum (S) of the absolute differences between the goal value y_i and the predicted value $\hat{y}$:

$$S = \sum_{i=1}^{n} |y_i - \hat{y}| \tag{2.2}$$

L_2 also known as the mean square error

$$S = \sum_{i=1}^{n} (y_i - \hat{y})^2 \tag{2.3}$$

Compared the robustness and stability of the two loss functions, L_1 is comparatively robust, however L_2 is more stable. Because L_1 is more resilient than L_2, it has several uses. L_1 is robust because it can withstand data outliers. This may be beneficial for research in which outliers may be disregarded safely and efficiently. If some or all outliers should always be evaluated, L_1 is the superior option.

Since L_2 squares the error, it follows that if the error is greater than 1.0, the mistake is greatly magnified. The model's error will be greater than the L_1 norm. Consequently, the model will be more sensitive to this sample, necessitating

adjustments to reduce the error. If this sample is an outlier, the model must be modified to accommodate it, at the price of many other normal samples with less error than the outlier.

As a result of instability, a slight horizontal variation in the data set may cause the regression line to leap significantly. On certain data configurations, the approach has a large number of continuous answers; nevertheless, a little shift in the data set misses a large number of continuous solutions inside a particular section of a data structure. After excluding answers in this area, the slope of the L_1 line may be larger than the slope of the preceding line. In contrast, the solution of L_2 is stable due to the fact that for each tiny change in a data point, the regression line will always shift minimally; that is, the regression parameters are continuous functions of the data set.

L_2 differs significantly from the human visual system (HVS). HVS is very sensitive to changes in local information, light, and colour, and the human-made camera's photosensitivity is based on HVS (Parraga et al., 2002).

Similar to HVS, the structural similarity index (SSIM) may be generated and is sensitive to local structural changes. Even with white noise, contrast enhancement, and mean-shifted image processing, SSIM does not vary much. Since SSIM takes into consideration brightness, contrast, and structural indications, its findings will be more precise than those of L_1 and L_2 in general.

$$SSIM(y_i, \hat{y}) = \frac{(2\mu_{yi}\mu_{\hat{y}}+c_1)(2\sigma_{y_i\hat{y}}+c_2)}{(\mu_{y_i}^2+\mu_{\hat{y}}^2+c_1)(\sigma_{y_i}^2+\sigma_{\hat{y}}^2+c_2)} \tag{2.4}$$

where the μ_{yi} is the average of y_i, $\mu_{\hat{y}}$ is the average of $\hat{y}$, the $\sigma_{y_i}^2$ is the variance of y_i, $\sigma_{\hat{y}}^2$ is the variance of $\hat{y}$, $\sigma_{y_i\hat{y}}$ is the covariance of y_{i} and $\hat{y}$. $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ are the two variables to stabilize the division with weak denominator; L is the dynamic range of the values, k_1=0.01 and k_2=0.03 by default (Peng et al., 2020).

Backpropagation neural network is one of the common models used in supervised

learning. Backpropagation Neuron Networks discover parameters by reducing the loss function's value. Deep learning models commonly utilize Stochastic Gradient Descent (SGD) and backpropagation to efficiently optimizes the model's parameters to minimize the loss function and computes the gradients that guide the parameter updates during training, allowing the network to learn and make accurate predictions on new data (Newton, Pasupathy & Yousefian, 2018). For classification tasks, Cross Entropy Loss is the most frequently used loss function. In binary classifications, the model is required to predict between just two possibilities in the conclusion. Our expected probabilities for each category are *p* and 1-*p*. This phrase means:

$$L = \frac{1}{N}\sum_i - [y_i \cdot log\,(p_i) + (1 - y_i) \cdot log\,(1 - p_i)] \tag{2.5}$$

where y_i represents the label of sample *i*, where the positive class is denoted as 1.0, and the negative class is denoted as 0. p_i represents the probability that the sample *i* is predicted to belong to the positive class (Murphy, 2012). The multi-class case is an extension to the binary classification:

$$L = \frac{1}{N}\sum_i L_i = -\frac{1}{N}\sum_i \sum_{c=1}^{M} y_{ic}\, log\,(p_{ic}) \tag{2.6}$$

where *M* is the number of categories; If the true class of sample *i* is equal to *c*, y_{ic} takes 1.0, otherwise takes 0. p_{ic} is predicted probability that the observed sample *i* belongs to class *c*.

During the forward pass of backpropagation, the input data is fed through the neural network, and predictions are made. The predicted outputs are then compared to the actual targets using the loss function to calculate the error. The backpropagation algorithm then starts at the output layer and works backward through the network to calculate the gradients of the loss function with respect to each parameter in the model. These gradients represent the sensitivity of the loss function to changes in the model's parameters.

Once the gradients are computed, SGD (or another optimization algorithm) uses them to update the parameters in the direction that reduces the loss, as described earlier. This process is repeated for multiple iterations (epochs) until the model's performance converges to an acceptable level.

In summary, Stochastic Gradient Descent (SGD) and backpropagation are integral components of training neural networks. SGD efficiently optimizes the model's parameters to minimize the loss function, while backpropagation computes the gradients that guide the parameter updates during training, allowing the network to learn and make accurate predictions on new data.

Because of the remarkable capabilities demonstrated by Convolutional Neural Networks (CNNs), they have become a focal point in numerous studies targeting vehicle-related scene comprehension. However, an issue arises when the network's depth becomes excessive. Modifying parameters through backpropagation in such deep layers can result in slow parameter adjustments closer to the input layer. Additionally, the application of gradient descent algorithms can lead to the training outcomes converging towards local minima instead of the global minimum. Furthermore, pooling and transformation layers might inadvertently cause the loss of valuable information, neglecting the interconnectedness between parts and the entirety of an image. The most important, given that CNNs operate within a high-dimensional realm, comprehending and visualizing how input data evolves through different layers becomes challenging for humans. Moreover, CNNs arrive at decisions by amalgamating features learned from data, yet these decisions might lack a clear, transparent connection to particular input features or attributes. From the perspective of feature understanding, CNNs acquire layered depictions of features, typically commencing with basic attributes like edges and advancing to more advanced traits. The amalgamation of these attributes across multiple tiers can be intricate and pose challenges in interpretation.

In order to make up for the shortcomings of CNN, many current research attempts

to achieve scene understanding through the combination of Capsule Network (CapsNet) or CapsNet and CNN (Chen et al., 2020; Liu et al., 2020; Liu et al., 2021).

To begin with, CapsNets tackle the issue of vanishing gradients, common in deep networks, by employing routing-by-agreement mechanisms. These mechanisms leverage the consensus between lower-level and higher-level capsules to effectively guide gradients during training.

Under challenging conditions such as extreme weather and intricate traffic scenarios, CapsNets exhibit potential in efficiently managing occlusions. This ability stems from their consideration of object part presence and orientation, making them potentially more resilient in crowded scenes.

Furthermore, Capsule Networks prioritize minimizing the usage of max pooling layers—a common feature of CNNs. This strategic approach aids in the preservation of spatial information, mitigating the loss of spatial relationships.

In comparison to CNNs, which often demand substantial training data, CapsNets display a tendency to require fewer labeled examples. This characteristic, coupled with their potential to learn generalizations across diverse object poses and variations, can lead to better generalization on unseen data (Hao, 2020).

Moreover, Capsule Networks introduce a hierarchical architecture tailored to capturing spatial relationships among object parts. This hierarchical representation proves advantageous in comprehending intricate patterns and rotations more effectively than CNNs (Qu & Shao, 2020).

Lastly, CapsNets leverage dynamic routing mechanisms to facilitate communication between lower-level capsules and their higher-level counterparts that exhibit congruence in orientation. This dynamic routing mechanism bolsters their capacity to manage spatial hierarchies adeptly (Liu, Yan & Kasabov, 2020).

Beyond Capsule Networks, Transformers have emerged as an additional approach to compensate for the limited semantic comprehension. The integration of positional embeddings furnishes spatial information to the model. These embeddings play a pivotal role in facilitating the model's comprehension of the relative positions among distinct tokens. This understanding is imperative for the model to effectively capture spatial relationships and high-level semantics (Zhang et al., 2022). Concurrently, the self-attention mechanism intrinsic to Transformers empowers them to encapsulate global context and extended relationships. This capacity to assimilate information from distant regions of the image augments the model's aptitude for apprehending high-level semantics (Lu et al., 2023). In the case of the Swin Transformer, a shifted window mechanism is employed. This mechanism involves displacing local windows in consecutive stages, creating an offset between them. This strategic displacement facilitates the capturing of increased spatial context and the establishment of relationships among neighboring local windows. As a result, the model's grasp of high-level semantics is further enriched (Lu, Zhao, Xu & Zhang, 2023).

To conclude, while CNNs outperform traditional machine learning methods, they still face limitations in comprehending high-level semantics within intricate traffic scenarios. Hence, there is a promising avenue to enhance CNNs' semantic understanding prowess through models like CapsNet and Transformer. Embracing CapsNet and Transformer holds great potential for refining network performance in the context of understanding complex traffic scenes.

2.3 Typical Deep Learning Model

2.3.1 Capsule Network

The introduction of CapsNet in deep learning accelerates the development of computer vision. CapsNet observes the scene and represents objects as possibility vectors,

capturing various aspects such as posture (i.e., location, dimension, orientation), deformation, velocity, and more. This allows models to incorporate the relative relationships between objects in the scene. The convolutional neural network mainly focuses on the basic element representation inside the object, and it is unable to consider the high-level semantic association between basic elements and complex objects like CapsNet. CapsNet activates several characteristics of the same class of objects using capsules containing numerous neurones (Shi, et al., 2022). An increase in the number of capsules with constant output leads to a rise in the segmentation accuracy. CapsNets are able to recognise objects based on their postures rather than the objects themselves since they have also acquired the object's spatial connections. In several instances, compare to CNN, the output of CapsNet seems to be more relevant (Patrick, 2022).

The architecture of CapsNet is shown in Figure 2.1. The CapsNet starts with an input layer that receives the raw data, such as an image. The first layer of the CapsNet is a traditional convolutional layer that applies filters to the input to extract low-level features. Next, the output of the convolutional layer is transformed into primary capsules. Each primary capsule group represents a set of detected features in the input image. The primary capsules are then fed into a capsule layer, which consists of capsules (vectors of neurons). Each capsule in this layer represents a higher-level entity or part of an object. The key innovation of CapsNet is dynamic routing. Capsules in the lower layer communicate with capsules in the higher layer through dynamic routing algorithms. Dynamic routing allows capsules to vote on the presence and properties of higher-level entities, creating more robust and interpretable representations. Capsules not only detect the presence of features but also estimate their pose, such as the position, orientation, and size of the entity they represent.

This pose estimation enables CapsNet to be more robust to spatial transformations and view variations. For tasks like image reconstruction, a decoder network can be added to the CapsNet architecture. The decoder network takes the output of the capsules

and reconstructs the original input, providing a reconstruction loss that helps the network learn better representations.

Figure 2.2 illustrates the operational process of CapsNet. It delineates the steps as follows: firstly, computing a matrix multiplication of the input vectors; secondly, adding scalar weights to the input vectors to obtain weighted input vectors; and finally, executing a vector-to-vector nonlinear transformation. The input vector of the capsules is obtained from the output of three low-level capsules. The probability of a related object is estimated using these lower layer capsules. The vector represents the internal states of the object being estimated. These vectors are multiplied by their corresponding weight matrix, which encode the spatial relationships between high-semantic objects and their basic constituent elements. The resulting multiplication yields the anticipated position information of the object. Through dynamic routing, capsules in one layer send "votes" to capsules in the next layer based on their agreement about the presence of certain features, in contrast to the backpropagation algorithms used by convolutional neural networks (Steuer et al., 2021; Liu et al, 2020).

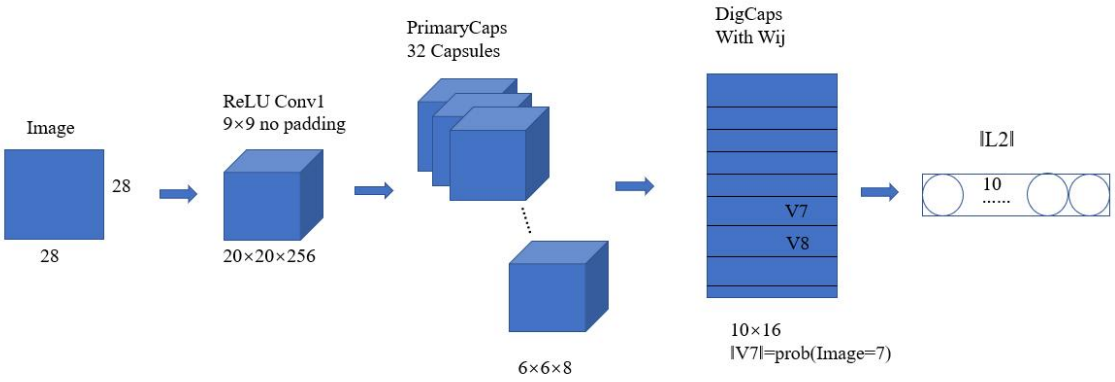


Figure 2.1 The architecture of Capsule network

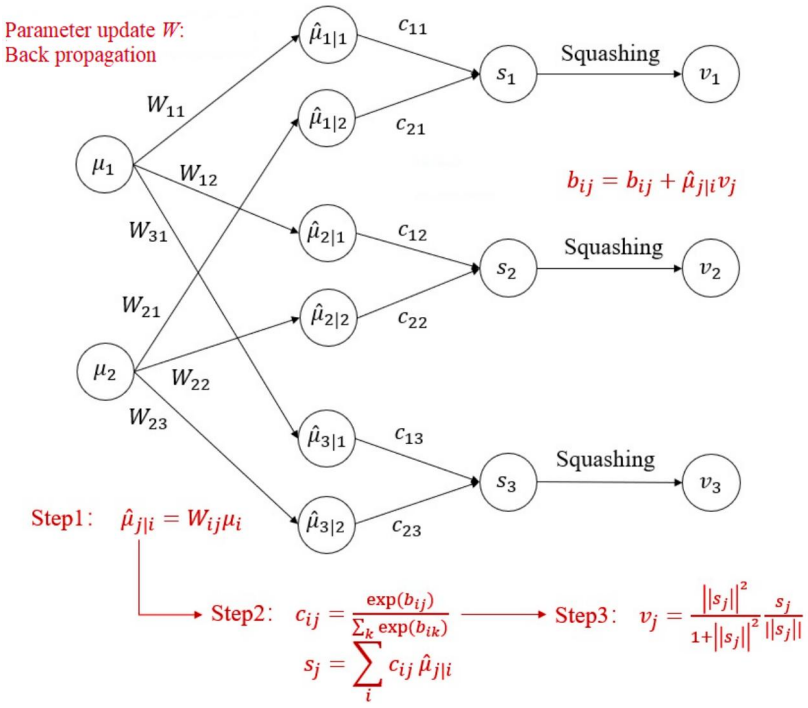


Figure 2.2 The operational process of CapsNet

When comparing the linear weighted summation of fully connected neural networks to the weighted summation of CapsNet S_j, the latter introduces a coupling coefficient c_{ij} which is expressed as (Ghafari, et al., 2021),

$$c_{ij} = soft\,max\,(b_i) = \frac{exp\,(b_{ij})}{\sum_k exp\,(b_{ik})}. \tag{2.7}$$

Meanwhile,

$$b_{ij} = b_{ij} + \hat{\mu}_{j|i} \cdot v_j \tag{2.8}$$

where the coupling coefficient c_{ij} represents the strength of coupling between capsules, while b_{ij} denotes the logarithmic prior probabilities of capsules i and j being coupled. Another notable advancement in CapsNet is its utilization of a nonlinear activation function that generates a vector and normalizes the input vector to unit length (Jia & Huang, 2020).

$$v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2}\frac{s_j}{\|s_j\|} \tag{2.9}$$

where v_j is the vector output of capsule j and s_j is its total input.

2.3.2 Loss Function

In CapsNet, the probability of a pattern is characterized by the length of the capsule vector. Furthermore, a margin loss function is employed for each capsule to represent the characteristics of the object class.

$$L_c = T_c \max\left(0, m^+ - \|v_c\|\right)^2 + \lambda(1 - T_c) \max\left(0, \|v_c\| - m^-\right)^2 \qquad (2.10)$$

The loss of class c, denoted as L_c, is determined based on whether there are objects belonging to class c in the images, where $T_c = 1$ indicates the presence of such objects. To mitigate the loss when certain object classes are absent and prevent excessive compression of the activation vector modulus of all object capsules during the initial learning stage, m^+ is typically set to 0.9, m^- to 0.1, and λ to 0.5. The total loss is then obtained by summing up the individual losses of each digital capsule.

2.3.3 Comparison of CNNs and CapsNet

CNNs usually require a raft of data sets for training, CapsNet can generalize using much less training data. CapsNet still needs to solve the defects about the background image notwithstanding, in general, CapsNet can better deal with ambiguity than CNNs, it also performs well in very crowded scenes.

The most important thing is that CNN loses a lot of information in the pooling layer, which reduces the spatial resolution, and leads to almost unchanged output for inputs with small adjustments. This has a serious impact on semantic segmentation. In semantic segmentation, it is essential to preserve detailed information within the network. CNN currently solves this conundrum by constructing an intricate architecture to reclaim this lost information. CapsNet stores a sequence of pose information encompassing the precise object location, orientation, dimensions, tilt, and size, instead

of discarding it and later attempting to recover it. In other words, a minor variation in the input of CapsNet will bring in a small modification in the output. This enables CapsNet to ultilize a simple and unified architecture to deal with different visual tasks. Finally, CNN needs additional components to automatically identify the attribution of each small part, and CapsNet can build a hierarchical structure of these parts (Liu et al., 2021).

2.4 SiamRPN

The area of tracking is currently separated into two major branches: correlation filter-based tracking algorithms and Siamese network-based tracking algorithms (Wu et al., 2023). Due to the difficulty of extracting and updating depth data in real time, the online fine-tuned network-based depth target tracking approach makes the tracker much less effective. SiamFC provides an offline end-to-end trained full convolutional pleiomorphic network based tracking approach, which has a quicker tracking speed while keeping a high tracking accuracy, as a solution to this challenge. SiamRPN utilises the RPN structure of target detection to make the tracking frame more precise and to reduce the time-consuming multi-scale testing, hence attracting significant interest.

The Siamese-RPN permits offline end-to-end training using extensive image datasets. The Siamese subnetwork for feature extraction employs AlexNet and consists of five layers of convolution. Two network branches, the template frame and the detection frame, share CNN parameters. Two branches of the RPN network classify the foreground background and conduct proposal regression, respectively (Xiao et al., 2023).

Most tracking algorithms see tracking as a localisation issue, but SiamRPN differs in that the localisation of the target is just as crucial as the regression prediction of the

target's bounding box. For this reason, SiamRPN encapsulates the tracking issue into a single-sample detection problem, i.e. a local detector must be setup with information from the first frame. SiamRPN combines consequently the Siamese network for tracking with area recommendation networks for detection: Siamese network facilitates adaptation to the tracked target, letting the algorithm to utilise information about the tracked target to initialise the detector; area recommendation networks enable the system to generate more accurate predictions about the location of the target (Huang et al., 2022). SiamRPN may be taught end-to-end using the combination of these two.

Moreover, earlier filtering-based techniques are incapable of enhancing tracking performance in a data-driven manner. SiamRPN is capable of end-to-end training, which allows for leveraging larger datasets to enhance its performance in a data-driven manner.

By using Siam-FC as a benchmark, SiamRPN achieves not only an increase in accuracy, but also a quicker tracking speed and a more optimal balance between accuracy and speed (Yang et al., 2020).

2.5 Transformer

Compared with RNNs, Transformers are able to model long-term dependencies between elements of input sequences and support parallel processing of sequences. The straightforward design of Transformers allows multiple modalities to be handled using similar processing blocks. Some works try to transfer Transformer to the computer vision field to process image data. However, there are a series of challenges in applying Transformer to the image field. First of all, images are extremely sparse data sets with high resolution and many pixels. The usage of global self-attention in Transformer leads to a substantial computational burden on the model. This feature promotes the wide application of CNN with sliding window design on image data. Secondly, the images

are not naturally represented as a sequence of data, which presents a challenge in converting the image into a suitable sequence and feeding it as input to the Transformer. In addition, the scale of image data varies greatly, and it is often not effective to directly process it as a sequence (Shi et al., 2023).

The introduction of Swin-Transformer has addressed the aforementioned challenges to a considerable extent, leading to remarkable achievements in various computer vision tasks, including image classification, object detection, and semantic segmentation. At present, some work has explored the application of Transformers in image tasks, such as ViT, which divides the image into image blocks and inputs them into Transformer as a sequence, and uses Position Eemdedding to preserve the relative positional relationship (Dosovitskiy et al., 2020; Parvaiz et al., 2023).

The groundbreaking contribution lies in its direct utilization of a Transformer architecture for image classification by applying it to non-overlapping medium-sized image patches. When contrasted with convolutional networks, this approach attains a remarkable equilibrium between speed and accuracy in image classification. Nonetheless, ViT demands a substantial number of images to effectively train the network, whereas DeiT enhances the training strategy to reduce the required image dataset size (Shi et al., 2023). Although ViT improves image classification, it is not suitable for high-resolution images because its complexity is quadratic in image size. When employing the ViT model directly with upsampling or deconvolution algorithms in dense vision tasks like vehicle detection and scene segmentation, the outcomes are relatively unsatisfactory. The Swin Transformer, on the other hand, refines the ViT architecture, leading to significant improvements in the image classification task. Empirically, the Swin-Transformer architecture achieves the most favorable speed-accuracy trade-off among these image classification methods (Liu, et al., 2021). Nevertheless, this research emphasizes overall performance, not limited to classification alone. Consequently, the Swin Transformer demonstrates outstanding capabilities in

various tasks, including target detection and segmentation, resulting in excellent performance. By carefully balancing model performance and speed, the Swin Transformer sets a new state-of-the-art (SOTA) benchmark in COCO target detection and ADE20K semantic segmentation.

CNN is an inductive bias formed based on the assumption that adjacent pixels have greater similarity. Locality is its typical feature, while Transformer performs global interaction on features. Therefore, the way of feature learning and the content of feature encoding are quite different. CNN is good at fully retaining local information in the process of processing features, while Transformer focuses more on understanding global information. The fusion of Transformer and CNN proves to be an effective approach for enhancing the model's feature extraction capabilities. To enable a more comprehensive understanding of image content, researchers have explored integrating Transformer and CNN through structural fusion, feature fusion, and mechanism fusion. These methods aim to leverage the strengths of both architectures, leading to improved performance and better comprehension of visual information.

Guo et al. (2022) formed a new network structure by effectively combining multiple modules by means of structural fusion. CMT extracts the local features of the image through deep convolution. Building upon this foundation, the attention module captures global dependencies among the features while incorporating residual links in various parts of the model to preserve local features and achieve a fusion of both local and global characteristics. MobileViT regards Transformer as a module and integrates it into the convolutional neural network to make the model both local and global (Mehta & Mohammad, 2021). MPViT uses multi-channel parallel Encoder and convolution to realize the sharing of global features and local features, achieving SOTA performance (Lee et al, 2022).

The feature fusion starts from the feature level, and generally adopts a parallel branch structure to fuse the features extracted by CNN and Transformer to enhance the

feature expression ability. The Conformer model proposed by Peng et al. (2021) designs parallel CNN and Transformer branches, and uses a bridge module to achieve feature fusion. Conformer uses Conformer as a Backbone, and the mAP on COCO reaches 44.9%. The Mobile-Former proposed by Chen et al. (2022) also uses parallel branching and bridging modules, which design a feature bridging method based on a lightweight cross-attention mechanism, making the fusion of features more effective. MobileFormer has a faster inference speed when processing high-resolution images, but when the input resolution drops, the inference speed is surpassed by MobileNetV3. This is because the code implementation of the bridge module is not efficient, and the calculation of the module does not change with the resolution.

Structure fusion and feature fusion realize the combination of Transformer and CNN in a serial or parallel manner. However, the attention mechanism and convolution are still two different parts, and the correlation between them is not fully utilized. The mechanism fusion reasonably integrates attention and convolution by digging deep into the inner connection between the two. ACmix deeply analyzes the similarity between self-attention and convolution feature extraction mechanisms, and realizes the mechanism fusion of self-attention and convolution by sharing feature mapping parameters. ACmix is both local and global, achieving 51.1% mAP on COCO when transferred to the object detection task (Pan et al., 2022).

2.6 Scene Understanding

The comprehension of traffic scenes has emerged as a pivotal research area in computer vision and a prominent focus in artificial intelligence, primarily driven by advancements in autonomous driving technologies.

A scene comprises a wide array of visual elements within a specific environment. the emergence of scene-based feature representation as a more efficient approach for interpreting visual scenes is rooted in its ability to capture context, semantic

relationships, and hierarchical information. By considering the entirety of a scene, algorithms can achieve a deeper and more robust understanding, aligning with the complexities of real-world scenes and improving performance across various computer vision tasks (Xie et al., 2020).

Classification of scenes is the process of detecting and learning semantic labels in photographs and situations. In general, the approaches include scene classification with local semantics, scene classification with intermediate semantics, and scene classification with a semantic subject model (Prykhodchenko et al., 2022; Wang, Peng & De Baets, 2021). Nevertheless, scene classification still exhibits a limited level of comprehension within the model, possibly due to the classification task's challenge in establishing relationships between objects across different classes.

Scene retrieval is a comprehensive understanding of the scenes that have been saved and retrieved from a database. In addition to the modules that exist in scene understanding, there is a separate module for similarity matching in scene retrieval (Li, Zhou & Shen, 2020).

In contrast, object detection must not only acquire class labels and learn characteristics, but also determine the object's orientation and location in the scene. First, characteristics that may describe the object are retrieved and combined with a specified classifier to achieve categorisation.

While object detection and identification can help outline the boundaries of entities, true recognition of each object at the pixel level and defining its precise boundary requires human comprehension of the situations. Simple detection and identification do not always assess the scene's relative posture information. Therefore, excellent segmentation significantly aids autonomous cars in perceiving the driving environment. Semantic segmentation is formulated as a classification issue for pixels with semantic labels. At the pixel level, the model leverages a set of object classes to label all image

pixels (Bucher et al., 2021).

In contrast to indoor and static scene understanding, comprehending vehicle-related traffic scenes presents greater challenges due to the interference of additional factors on prediction accuracy. Firstly, real traffic scenes are exceptionally complex, containing numerous objects and pervasive occlusion issues. Furthermore, in dynamic environments with vehicle-related scenes, the same objects often appear from multiple angles, making it challenging for CNNs to learn various views from limited input data. Additionally, outdoor weather conditions further diminish the discriminating accuracy of objects under varying illumination situations.

Therefore, various research have been conducted to resolve the aforementioned challenges. Object occlusion in the scene is remedied by a detection technique comprised of individual components. This method divides the human body into numerous sections for object identification, integrates the data, and then reassembles the components (Yan et al, 2015). However, this strategy significantly exhausts training time.

By employing tracking techniques, T-CNN enhances the performance of video target recognition, aiming to overcome comprehension challenges related to moving objects in traffic scenarios. This approach depends on a multitude of algorithms for video target tracking; as a consequence, the process is more complex and real-time performance is limited. In addition, there is no direct use of video information, such as fuzzy and motion information, to address comparable issues in a video. Despite these constraints, STMM makes use of the video's motion information to comprehend the traffic scenario. It is a two-way cyclic convolutional network, thus the module may utilise motion information between a few consecutive frames or learn movement and visual information about the object over an extended period of time (Zheng et al., 2022). In addition, a brain-like spiking neural network (SNN), NeuCube, is also considered to be one of the outstanding models capable of accurately identifying fast-moving objects.

NeuCube is organised in three dimensions, representing the organisation of primary visual cortex. Kasabov et al. (2018) capture spike column data from a dynamic visual sensor (DVS), which mimics the human retina's functionality by generating spike column data based on object motion through address event output. Subsequently, the spike sequences undergo convolution and are introduced into the NeuCube system. During the initial phase of deep unsupervised learning, the system acquires the spatio-temporal patterns of the data by leveraging plasticity that is sensitive to spike timing. Following this, in the second stage, supervised learning is employed to train the network for the classification task. Convolutional algorithms and network mapping are employed to emulate the functionality of retinal ganglion cells and the retinal organization of the visual cortex (Kasabov, 2019).

Efforts have also been made to address the processing demands posed by the online learning scenario through fine-tuning the impulse neural network. A novel eSNN model incorporates a range of innovative components designed for efficient handling of online learning applications, including constraints on neuron repository sizes and the implementation of sliding windows. The model that was developed exhibits the ability to classify the input after a single training sample presentation, eliminating the requirement to present the entire training set beforehand. Through the integration of the proposed eSNN approach with a drift detector, a dependable drift object learning model was effectively created (Lobo et al., 2018). Nonetheless, for the short-term forecasting models, their performance gradually diminishes as the forecasting horizon increases. To overcome this limitation, Ibai et al. (2018) propose a long-term estimation scheme that incorporates automatic mode discovery and couples it with an online change detection and adaptation mechanism. This enables similarity-based clustering of daily traffic volume data to generate long-term forecasts and monitor them in real-time, allowing adjustments in case of mismatches or surprises. The framework employs the evolving architecture of impulsive neural networks (eSNN) to achieve adaptation without the

need for retraining the model, enabling the entire system to operate autonomously in an online fashion (Laña, 2019).

2.7 Semantic Segmentation

In this section, our focus will be on the structural characteristics of fundamental segmentation models.

Previous research has indicated that image segmentation near the border between two objects is much less accurate compared to areas away from the boundary. Existing top-performing systems rely on pixel-level labeling, which incurs high costs and demands substantial labor. To address this issue, one approach involves employing precise contour detection to achieve more accurate segmentation results at the object's boundary. The primary objective is to classify area proposals using object detection techniques and then further refine these proposals using saliency detection methods (Yao & Wang, 2023). Although this model has undergone extensive improvements to increase accuracy, there is room for further enhancement.

D-RefineNet is another depth-assisted model that integrates a depth-assisted loss function from RefineNet. This function leverages depth information to capture sharp changes at the object's boundary, constraining edge segmentation and thereby enhancing its effectiveness. Notably, the proposed network can predict object classes using RGB images alone, with depth photos solely utilized for loss functions, avoiding any increase in model size (Chang, Guo & Ji, 2018).

Many object suggesting algorithms encounter challenges such as occlusions, shape changes, and lack of class information. General techniques using bottom-up regional grouping or segmentation attempt to address these issues. They measure similarity to determine if two neighboring regions should be combined, employing an inference procedure to aggregate regions into super regions for generating object proposals.

However, such solutions often require manual adjustments or configurations, limiting performance in complex scenarios.

To address this, a recurrent neural network (RNN) is employed to learn a segmentation proposal through a hierarchical strategy for area grouping. The learning process involves cross-region and similarity-based measures during the bottom-up region merging. Additionally, for enhancing cross-region similiarity and object prediction, a structural loss function is introduced to account for candidate mixture by evaluating the similarity and objectivity of neighboring areas (Pinheiro, Collobert, & Dollár, 2015).

Due to uncontrollable variables like weather and lighting conditions, images can suffer from blurriness or obstructions, leading to challenges in accurately detecting objects. While deep visual models excel at segmenting traffic scenes, they struggle with changes between source and target datasets brought about by varying weather and lighting. To tackle this issue and create features that remain consistent across different scenarios, an unsupervised domain adaptation model is introduced (Saffari & Khodayar, 2023). This innovative approach involves capturing sparse representations of source traffic scenes using spectral low-rank dictionary learning. Additionally, a generative adversarial framework is utilized to grasp the distribution of sparse features from the source. Simultaneously, alignment of source and target scene representations is achieved through a sparse domain-invariant feature extractor, which is honed through a min-max optimization process. To validate its effectiveness, the proposed model is pitted against cutting-edge methodologies and is proven superior through experiments conducted on a real-world dataset.

While the aforementioned models demonstrate commendable performance, they still encounter challenges such as instances of false positives and missed detections. Present convolutional neural network CNN architectures emphasize inter-layer

information flow while disregarding the significance of spatial information within individual layers. In response to these issues, An innovative solution is proposed known as the Attention-Based Spatial Segmentation Network (ABSSNet) (Li, Zhao & Wang, 2021). This novel approach comprises two key components: an encoder featuring convolutional attention modules and a multitask decoder leveraging Spatial CNN. The encoder is constructed upon a dilated ResNet backbone, where convolutional attention modules are seamlessly integrated into Res-Blocks, thereby bolstering spatial localization comprehension. The decoder, employing SCNN, effectively translates encoder features into segmentation outcomes, with multiple distinct decoders dedicated to lane line and road detection. By synergizing attention mechanisms with SCNN, our network achieves heightened spatial information perception and subsequently elevates the precision of detection tasks.

Nonetheless, during the convolution and pooling process, CNN tends to lose image details, resulting in a shrinkage of the feature map's size, making it challenging to precisely identify the object's contour and accurately segment pixels corresponding to it. Hence, objects cannot be precisely segmented. Currently, many studies deal with this problem by transfer learning. In transfer learning, Fine-tuning is able to adapt a pre-trained model to a specific target task, while split-task enables the model to address multiple tasks simultaneously using the same base model. By leveraging transfer learning and carefully designing your new task-specific layers, the knowledge learned from the source dataset can be efficiently utilize and achieve good performance on semantic segmentation (Donahue et al., 2020).

2.8 Depth Estimation

In recent years, numerous studies have focused on depth estimation using deep learning in both indoor and outdoor scenes (Li et al., 2021; Ji et al., 2021; Zhou, Wang & Yang, 2020). Depth estimation can be approached through two distinct methods: binocular

depth estimation and monocular depth estimation. Binocular depth estimation leverages the disparity present in two images taken from slightly different perspectives to achieve precise depth estimation (Poggi et al., 2021; Zheng et al., 2022). This approach provides heightened accuracy and dependable depth perception when compared to monocular techniques. Moreover, binocular systems exhibit greater resilience to challenges such as alterations in lighting, texture-deficient areas, and occlusions—difficulties that monocular methods may encounter in accurately gauging depth under similar conditions (Shu et al., 2022; Wang et al., 2022). Contrasting binocular depth estimation, monocular depth estimation centers around a single camera, making it a more economical and straightforward implementation when compared to binocular configurations that necessitate two cameras. Furthermore, the applicability of monocular depth estimation extends to a broader array of devices, including smartphones, drones with single cameras, and robots, where spatial and budgetary constraints could hinder the use of multiple cameras. Consequently, for researchers operating within limited budget and equipment parameters, monocular depth estimation emerges as a viable cost-saving alternative while still upholding the prerequisite accuracy.

Presently, there exists a plethora of successful monocular depth estimation models based on deep learning. For instance, an innovative approach employs a deep neural network model grounded in a semantic divide-and-conquer strategy (Wang et al., 2020). This strategy entails breaking down an image into semantic segments encompassing object instances and background stuff categories. Subsequently, depth maps are predicted for each segment within a standardized space. The primary objective is to simplify the complexity of depth prediction while harnessing the consistent depth patterns within semantic groups. The proposed framework, dubbed SDC-Depth Net, encompasses four essential components: a backbone network, a segmentation module, a depth prediction module, and a depth aggregation module. The backbone network extracts feature representations from the input image. The segmentation module engages in semantic and instance segmentation to partition the image into semantic

segments. The depth prediction module computes depth maps tailored to semantic segments and individual object classes. The depth aggregation module assembles and combines the segment-specific depth maps to generate a coherent global depth map. This approach results in enhanced depth estimation performance, culminating in achieving state-of-the-art outcomes across diverse benchmarks.

However, due to the characteristics of CNN, some global features are often ignored by the model compared to local features. In order to make up for this shortcoming, some studies combine CNN with the Transformer of the self-attention module to achieve depth estimation.

The BinsFormer is constructed from three core components: the pixel-level module, the Transformer module, and the depth estimation module (Li, Wang, Liu & Jiang, 2022). The pixel-level module incorporates a backbone and decoder to capture and magnify image features. Within the Transformer module, a Transformer decoder is utilized to anticipate bins and embeddings, treating the generation of bins as a problem of predicting sets. This novel approach enables the model to explore global insights and forecast bins that facilitate integral depth estimation. The depth estimation module harmonizes the outputs from the previous modules to yield the ultimate depth prediction. Furthermore, an auxiliary scene comprehension query is integrated into the Transformer decoder to foretell the classification of the input surroundings. This supplementary query aids in generating suitable bins through implicit guidance.

Additionally, a multi-scale decoder architecture is introduced, enabling a comprehensive grasp of spatial geometric details and gradual refinement of depth estimation. This is achieved through the interplay between Transformer queries and multi-scale features, fostering the aggregation of features at varying levels of granularity.

Simultaneously, for CNNs, the inherent focus on local interactions through convolutions imposes constraints on their ability to effectively capture distant relationships, especially within shallow feature layers. This becomes particularly

problematic when discerning intricate foreground-background structures becomes challenging, especially in scenarios where object differentiation is complex. The MonoViT framework ingeniously combines Convolutional layers with Transformer blocks, amalgamating localized object-specific insights with broader scene-wide connections (Zhao et al., 2022). This innovative approach addresses the shortcomings of prior methods, which are hindered by the relatively confined scope of CNNs, resulting in struggles to adequately model long-range dependencies. The Depth Network leverages the power of both Transformers and convolutional layers to refine feature extraction and facilitate accurate depth prediction. Concurrently, a streamlined Pose Network is employed for the estimation of relative poses between consecutive frames, effectively enhancing the framework's overall performance.

2.9 Vehicle Tracking

Over the past decade, deep learning methods have demonstrated strong capabilities in visual object tracking. Conventional object tracking algorithms rely on particle filtering, which necessitates a large number of particles for classifier training, resulting in complex convolutional layers for feature extraction (Elhani et al., 2023). To enhance accuracy and speed, many algorithms have combined deep learning methods with relevant filtering techniques. HCFT utilizes VGG-16 to extract features from Conv3-4, Conv4-4, and Conv5-4, training corresponding correlation filters to locate the target accurately. Similarly, HDT utilizes a combination of multi-layer depth features and correlation conversion, enhancing the depth from three to six layers and adopting adaptive weight addition (Ma et al., 2020; Qi et al., 2021).

The another type of network to implement tracking task is Siamese networks, a typical end-to-end deep learning correlation filtering method, have been used to simulate the entire process of related filtering. One branch saves target template information, while the other extracts features in the search region. The merged parts generate the

response map, reflecting the target's state (Bertinetto et al., 2016). Siamese networks excel in one-shot and few-shot learning scenarios that can effectively learn to distinguish between different object instances using only a few examples, making them suitable for tracking tasks where acquiring extensive labeled data is challenging. At the same time, Siamese networks directly learn a similarity metric in the feature space. This characteristic is well-suited for tracking tasks where the focus is on finding the similarity between the target object's features and those of the candidate regions.

One of the primary hurdles in tracking tasks involves the intricate interplay between background elements and objects within complex environments. To tackle this issue, the joint Siamese attention-aware network (JSANet) integrates self-attention and cross-attention modules, strategically designed to surmount the challenges arising from subtle features and background noise (Song et al., 2022). The self-attention modules introduced in this framework synergize channel and spatial attention mechanisms. The channel attention component accentuates pertinent channel coefficients to spotlight high-scoring channels, whereas the spatial attention module transforms spatial domain data, ensuring precise identification of crucial regions. Moreover, the cross-attention element orchestrates the fusion of contextual dependencies between the target template and the search image through cross-channel attention. This sophisticated approach enables the unveiling of correlations between objects with temporal associations. The utilization of Siamese region proposal networks (SiamRPNs) further amplifies this methodology, enabling the prediction of a singular tracking region based on the feature streams that have been modulated by attention mechanisms.

However, most models focus on single-target tracking, while multitarget tracking research progress is relatively slower due to limited datasets and references. Single-target tracking is often used for short-term image sequences, while multitarget tracking deals with longer videos, involving various targets' appearance, occlusion, and departure. The implementation methods also differ, with single-target tracking prioritizing target relocation, while multitarget tracking focuses on matching detected

targets. Multitarget tracking algorithms can be detection-based or non-detection-based, and they are further classified into online tracking and offline tracking based on frame processing and utilization of subsequent frames (Luo et al., 2021).

A multiple target tracking method proposed by Sun et al (2021) involves using a detector to identify targets in video image space, predicting the position and motion of the targets in the next frame using a Kalman filter, calculating the overlap between detection and prediction boxes using Complete Intersection over Union (CIoU) as a distance measure, and using the Hungarian algorithm to perform data association between multiple targets. The Hungarian algorithm can effectively handle situations where there are multiple detections and tracks, and the number of detections might not match the number of tracks. It assigns each detection to a track and vice versa, making it robust against cases where the number of objects being tracked changes dynamically. Furthermore, the Hungarian algorithm guarantees finding the globally optimal solution to the assignment problem. This means that it provides the best possible assignment that minimizes the total cost among all possible combinations, ensuring high-quality associations between detections and tracks (Wang & Liu, 2022; Gai, He & Zhou, 2021).

2.10 Distance Estimation

Vehicle detection forms the basis for vehicle ranging, and estimating the distance from the vehicle in front is essential for vehicle collision avoidance systems. As a result, an increasing number of articles in the field of computer vision are focusing on the challenges related to vehicle detection and range estimation.

We delve into two areas of literature. In the first area, we explore vehicle detection and range estimation using a binocular camera. Binocular stereo vision involves using two cameras with a known baseline to capture images of the same scene from slightly different viewpoints. Generally, the first step of binocular stereo distance estimation is to make key points or features in the left and right images are identified, and

corresponding points are matched between the two images. And then calculate the disparity between matched points to represent the relative distance of objects in the scene. Finally, using the disparity and known camera parameters to compute the depth information for each pixel.

A multi-resolution stereovision system is proposed by Chui et al (2020). The system creates image pyramids by down-sampling the captured images to different resolutions. Each level in the pyramid represents a different scale of the scene. The process includes feature extraction, feature matching, disparity calculation, depth estimation, and fusion/refinement steps for each level of the image pyramid. By performing these steps at multiple resolutions, the system can handle objects at different distances effectively.

Also, there are some studies estimating distance by identifying corners with high eigenvalues in segmented regions of both images (Alvarado et al., 2022). The model segments the left and right images to identify regions of interest. Image segmentation techniques such as thresholding, edge detection, or clustering can be used to group pixels with similar characteristics into distinct regions. And then, within each segmented region, apply a corner detection algorithm to identify key points or corners. Corner detection algorithms like Harris corner detector or Shi-Tomasi corner detector are commonly used. These algorithms identify points where there are significant variations in intensity in multiple directions, making them suitable for detecting distinctive features. Once the corners are identified, calculate the eigenvalues of the gradient matrix at each corner point. The eigenvalues represent the rate of change of intensity in the x and y directions around the corner point. High eigenvalues indicate strong intensity changes, which correspond to well-defined corners. After matching and disparity estimation, apply triangulation to compute the 3D coordinates of the scene points corresponding to the matched corners in both images. With the 3D coordinates of the matched points, estimate the distance of the objects in the scene from the cameras. The distance can be calculated using simple geometric principles or calibrated camera

parameters.

Binocular distance estimation, which relies on using two cameras to capture images of the same scene from slightly different viewpoints, has certain disadvantages when compared to monocular distance estimation, which uses a single camera. Firstly, binocular distance estimation requires the use of two cameras, increasing the hardware complexity and cost compared to monocular systems that only need a single camera. Furthermore, accurate calibration of the stereo camera setup is crucial for precise distance estimation. Calibration involves determining the intrinsic and extrinsic parameters of both cameras, and any errors in calibration can lead to inaccurate distance measurements. Also, the baseline between the two cameras limits the effective field of view for distance estimation. Objects outside this field of view may not be accurately measured using the stereo vision system. At the same time, in complex scenes, occlusions and disparities between the left and right images can make feature matching and distance estimation challenging. These situations can lead to errors in distance estimation, especially for regions with insufficient texture or ambiguous features. Finally, changes in lighting conditions, reflections, and other environmental factors can affect the performance of binocular distance estimation, making it less robust in certain scenarios.

In contrast, monocular distance estimation, though having its own set of challenges, offers some advantages. Monocular distance estimation requires only one camera, making it a simpler and more cost-effective solution compared to binocular setups. And since monocular cameras are ubiquitous in many devices, it's easier to integrate monocular distance estimation into various applications and platforms. Moreover, monocular cameras can be placed in a variety of positions and orientations, providing more flexibility in system design. At the same time, recent advances in deep learning, especially with monocular distance estimation networks, have improved the accuracy and robustness of monocular distance estimation methods.

Monocular camera-based distance estimation relies on various cues or assumptions. Some research leverages cues such as perspective, size, and occlusion to estimate distance in the scene (Parker et al., 2022). And there are also same studies estimate distance by analyzing the movement of objects in consecutive frames or using visual odometry techniques (He et al., 2020). Normally, monocular camera-based distance estimation need to establish correspondences between the extracted features in the image and the corresponding points in the real world (Vijayanarasimhan et al., 2017). This can be achieved by manually identifying the matching points or using known 3D reference points. And then using the camera calibration parameters and the matched feature correspondences, apply triangulation to obtain 3D coordinates (X, Y, Z) of the real-world points. Triangulation calculates the intersection of rays originating from the camera center and passing through the matched feature points. Since the initial 3D coordinates are only up to an unknown scale factor, the known dimensions are needed to estimate the scale and convert the 3D coordinates to actual world coordinates. Once the actual world coordinates of the structures, objects, or road segments are obtained, the distances between points of interest in the scene can be computed by measuring the Euclidean distance between their corresponding 3D coordinates.

There has been some research implements monocular distance estimation using inverse perspective mapping (IPM) from a bird's-eye view. This transformation allows to estimate distances directly in the transformed view, which simplifies distance estimation. Perform a perspective transformation (IPM) is to map the image from the camera's perspective to a bird's-eye view. In the bird's-eye view, parallel lines become parallel and perpendicular to the ground, simplifying distance estimation. And then a mapping between the bird's-eye view and the real-world coordinates is created. This mapping relates the pixel coordinates in the transformed view to the corresponding real-world 3D points (Vakili et al., 2020).

Several studies have explored the integration of attention mechanisms to enhance

the accuracy of distance estimation models. An innovative strategy has emerged that integrates global relative constraints to promote consistent vehicle state estimations. This methodology emphasizes the significance of capturing both contextual and spatial details during the estimation process. The architecture of MSANet, the proposed framework, is elaborated, encompassing distinct streams for motion, context-awareness, and spatial information extraction from input data. To achieve enhanced estimation accuracy, the multi-stream attention fusion (MSAF) block is introduced as a means to effectively amalgamate these distinct features (Huang, Huang & Hsu, 2021).

Furthermore, certain research endeavors have integrated Transformer and attention modules into their models. The present paper proposes an advanced system rooted in deep learning, designed to autonomously identify physical distancing through the analysis of surveillance footage from security cameras. In this approach, TH-YOLOv5 is adopted for the purpose of object detection and classification, while Deepsort is employed to track individuals detected within bounding boxes outlined in the video material.

An innovative facet of this methodology involves the incorporation of Transformer Heads (TH) into the TH-YOLOv5 framework. This addition capitalizes on the self-attention mechanism, thus augmenting the model's predictive capabilities. Furthermore, the Convolutional Block Attention Model (CBAM) is introduced to pinpoint regions of interest within densely populated scenes. To achieve this, specific convolutional and bottleneck blocks are replaced with transformer encoder blocks, drawing inspiration from the architecture of vision transformers. This adaptation enables more comprehensive acquisition of global and contextual information, proving especially advantageous in intricate scenarios involving occlusions. These transformer encoder blocks are seamlessly integrated into the head segment of the backbone, enhancing feature representation and ultimately contributing to heightened object detection performance.

Moreover, the incorporation of the CBAM module aids the TH-YOLOv5 model in directing its attention towards pertinent target elements present in images captured by CCTV cameras.

The system estimates the depth between the camera and objects by utilizing coordinate transformations and camera intrinsics. The pairwise L2 normalization is used to calculate the distance between tracked individuals, and a violation index is computed to identify breaches of physical distance rules (Junayed & Islam, 2022).

Chapter 3 Methodology

The main focus of this chapter is on the research methods used for vehicle scene understanding. It will delve into the specifics of scene segmentation, vehicle tracking, distance estimation, and depth estimation. Furthermore, the chapter will outline the evaluation methods utilized throughout this thesis.

3.1 Research Design

Our research introduces a comprehensive experimental approach utilizing deep learning for various tasks, including semantic segmentation, vehicle tracking, distance estimation, and depth estimation. The main goal is to assess the impact of deep learning on enhancing traffic scene understanding. Simultaneously, our investigation aims to enhance the model's cognitive capabilities for high-level semantics by exploring both 2D and 3D aspects. Additionally, we seek to address the black box characteristics of CNN and the tendency to overly focus on local features, thus improving the model's comprehension of both local and global features.

3.1.1 Research Design for Scene Segmentation

In order to enhance the understanding of traffic scenes for autonomous vehicles, high-performance models need to not only segment various elements from the scene but also comprehend the high-level semantics of both the scene and objects. Deep learning is employed in this research project to mimic the characteristics and functioning principles of the human brain and neurons, enabling the comprehension of complex traffic scenes' high-level semantics.

In the context of human cognitive processing of scenes, the bottom layer of network extracts the most fundamental features, such as visual and auditory information. The higher layers of the network combine these fundamental operations and make decisions based on them. Ultimately, the top layer of the network infers the implicit semantics of the information, confirming semantic expression and understanding (Taye, 2023).

The high-level semantics, especially the positional relationships between visual objects in the scene, are of significant importance and the ultimate goal of deep learning. To further investigate vehicle-related scenes, the spatial connections of visual objects

for semantic segmentation are investigated.

The hierarchical relationships in capsule networks are central to their ability to capture complex features, spatial hierarchies, and semantic relationships within data. This architecture facilitates more nuanced and contextually rich representations, making capsule networks effective for tasks in deep learning. Remarkably, CapsNet achieves sophisticated results with only a small quantity of data compared to what traditional CNNs require.

Figure 3.1 presents a detailed comparison between the working principles of CNN and the human brain. Several key points can be highlighted:

Human brain cognition involves not just a single neuron focusing on a single feature, but rather a group of neurons focusing on a single characteristic. Unlike a single neuron that produces only one value, a neuron group can output a vector.

In CNN, the same element appearing in multiple places activates the same neuron, whereas different entities in the same position seem to activate different neurons in the human brain. Even entities with minor differences at the same position activate the same set of neurons, while the same object at a different position triggers distinct groups of neurons. This indicates that the human brain's recognition possesses a generalizability of position and adaptability to entity changes.

Human brain, when repeatedly stimulated by similar objects, creates specific connections to accelerate and enhance recognition. However, CNN lacks this capacity for forming specific connections based on repeated exposure to comparable elements.

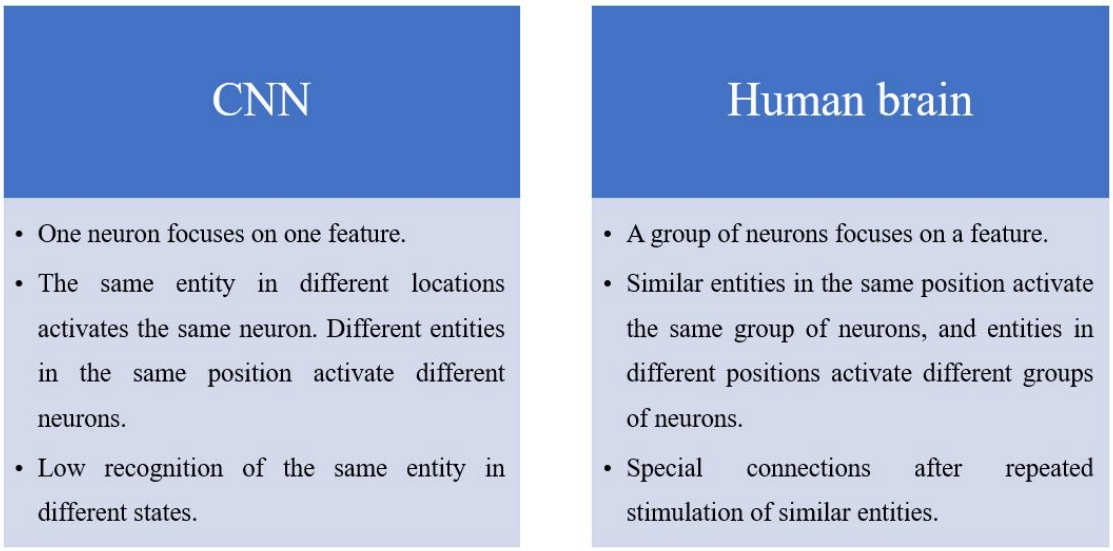


Figure 3.1 Comparation of the working principle of CNN with that of human brain

The primary difference between a capsule network and a CNN lies in the nature of their neurons. In a capsule network, neurons are comprehensive and contain all relevant information about the feature they represent, including its length, angle, orientation, and more. On the other hand, in a traditional CNN, each neuron is a standalone unit and lacks the capability to represent information such as location and angle. This is why CNNs often rely on data augmentation, where images from different angles and positions of the same object are added to the training set, to improve the model's overall performance.

In this context, capsules exhibit similarity to the human brain, as they can replicate the human brain to understand objects from different perspectives and positions. The concept of inverse rendering in the capsule network reflects this operating mechanism of the human brain.

Inside a capsule, several scalar operations are typically performed, including:

Multiplication of the input vector by the weight matrix using a matrix: This operation captures essential spatial correlations between low-level and high-level image features.

Weighting of the vector inputs: These weights determine which capsule of a higher level will receive the output of the current capsule. This is achieved through dynamic routing.

Summing the vectors with weights: The vectors are combined using the assigned weights.

Nonlinearization using the squash function: This function compresses the vector, ensuring its length remains between 1 and 0 while preserving its direction.

Dynamic routing establishes connections between capsules, and during the routing process, the lower-level capsule serves as the input vector for the upper-level capsule. The lower-level capsule generates a prediction vector for each higher-level capsule it may be sent to by multiplying its own output with a weight matrix. If the scalar product of the prediction vector is significant, it indicates a reliable forecast. Top-down feedback occurs from the higher capsule's output, which enhances the coupling coefficient of the upper capsule while reducing the coupling coefficient of other capsules.

In contrast to CNNs and the backpropagation algorithm, CapsNets and the dynamic routing algorithm exhibit a simpler network structure and parameter configuration. Capsule networks have fewer layers and connections, and the routing mechanism provides a more direct path for information flow compared to backpropagation, which involves computing gradients layer by layer.

However, despite their relative simplicity in terms of architecture and training set, capsule networks still necessitate a substantial allocation of memory and computational time due to certain factors. Firstly, the dynamic routing algorithm used in capsule networks involves iterative computations between capsules in different layers, which can be computationally expensive, especially as the number of capsules and their dimensions increase. Furthermore, capsule networks need to store information about the pose and instantiation of entities, which increases the memory requirements compared

to traditional CNNs. To address this memory issue, we aim to improve the network's architecture and routing algorithm.

The proposed approach introduces local kernel routing, where children are routed only within the local region of the given space. Furthermore, the transformation matrix for a member of the grid is shared within the same capsule type, but not across different capsule types. To address the global connectivity loss caused by routing, we enhance the capsule network with a “deconvolution" capsule that performs a transposed convolution operation using a routing protocol. By adopting the suggested deep convolution deconvolution architecture, we can retain global context information, significantly reduce the number of network parameters, alleviate memory usage, and achieve state-of-the-art results. Our capsule network, as illustrated in Figure 3.2, incorporates both posture and appearance data (Liu, Yan & Kasabov, 2022).

Unlike Convolutional Neural Networks where scalars represent object features, CapsNet is a renowned neural network for vector representation. If a subcapsule agrees with its parent capsule, it generates a vector output. The connection between the lower capsule and the parent capsule is influenced by the parent capsule's prediction, aligning with the actual output. The conventional CapsNet employs dynamic routing to compute an $N \times N$ matrix that transforms the child vector v_i of size N into the parent vector v_n of size N. Additionally, a novel technique is introduced that utilizes the matrix describing the object's pose and appearance for feature representation (Dhamidi & El-Sharkawy, 2020).

To optimize the model's performance, feature representation is achieved by combining the posture matrix and the appearance matrix using a transformation matrix. By combining the appearance feature vector A of size N with the transformation parameter matrix, a resulting matrix of size $N \times N$ is obtained. The position vector P_i of child i is then transformed into the posture matrix P_{i-n} of parent n, defined as:

$$P_{i-n} = P_i T_{i-n}^P. \tag{3.1}$$

The vector A_i of the child I is translated into the matrix A_{i-n} of the parent *n*, which is defined as follows:

$$A_{i-n} = (A_i + b_{i-n}) T_{i-n}^A \tag{3.2}$$

where T_{i-n}^P is the transformation matrix of pose, T_{i-n}^A is the transformation matrix of appearance, b_{i-n} is learn bias, image coordinates (*x, y*) are combined with each T^P. The final parental pose and appearance matrix P_n and A_n are obtained by combining all their transformation matrix, which are defined as follows:

$$P_n = Psquash\left(\sum_i \alpha_{i-n} P_{i-n}\right) \tag{3.3}$$

and

$$A_n = squash\left(\sum_i \alpha_{i-n} A_{i-n}\right) \tag{3.4}$$

where α_{i-n} is the weighting factor defined in the routing algorithm. The squash(·) is a nonlinear function which is proffered by Hinton (2017). Another nonlinear function Psquash(·) for pose matrix is defined as

$$Psquash(P) = \frac{P}{max\, abs(P)} \tag{3.5}$$

The dynamic routing mechanism defines $c_i \rightarrow n$ through the cross-correlation iterative optimization strategy between the child and parent vectors. Based on this concept, we multiply the pose features and appearance features that have been processed

$$c_{i-n} = \langle P_{i-n}, P_n \rangle_F \cdot \langle A_{i-n}, A_n \rangle_F \tag{3.6}$$

where $\langle , \rangle_F$ denotes the Frobenius inner product. Finally, the weighting factors α_{i-n}

for each child are calculated by applying the sigmoid function to $c_i \rightarrow n$.

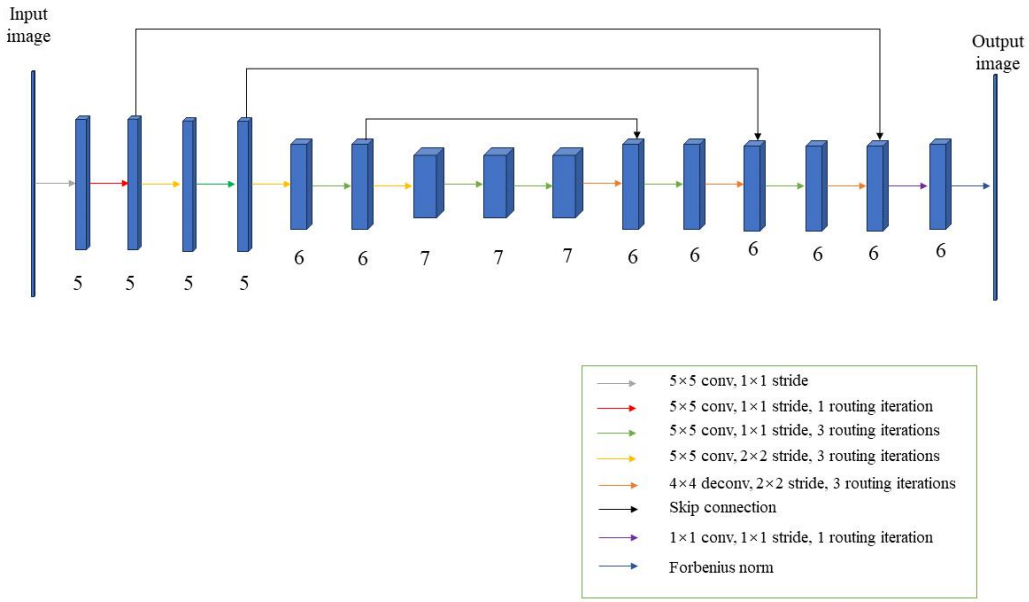


Figure 3.2 The architecture of segmentation capsule network

We generate a group of capsules P_i (x, y) and A_i (x, y) for each pixel (x, y) or intermediate layer in the input image (x, y), as illustrated in Figure 3.3. Moreover, a convolution kernel of size $n \times n$ is applied to the posture matrix P_i and the matrix A_i to incorporate neighborhood information into the network. The projected multilabel subdivision L represents the index of the last activated capsule at each position (x, y):

$$L(\mathrm{x},\mathrm{y}) = arg_i \max \left(\|P_i(x,y)\|_F, \|A_i(x,y)\|_F\right) \tag{3.7}$$

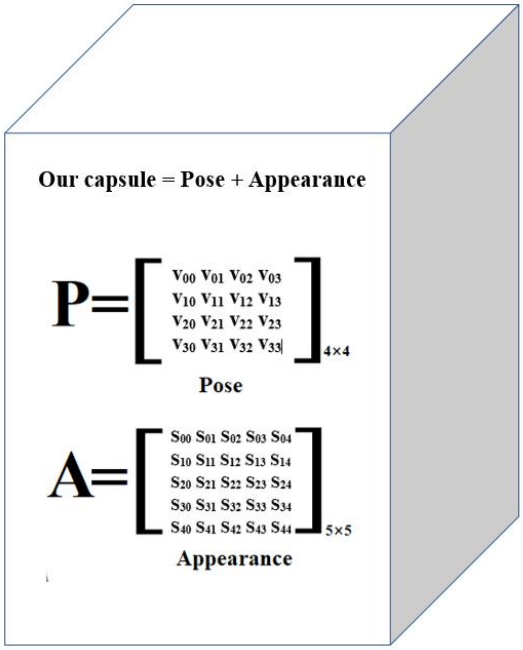


Figure 3.3 Composition of capsules

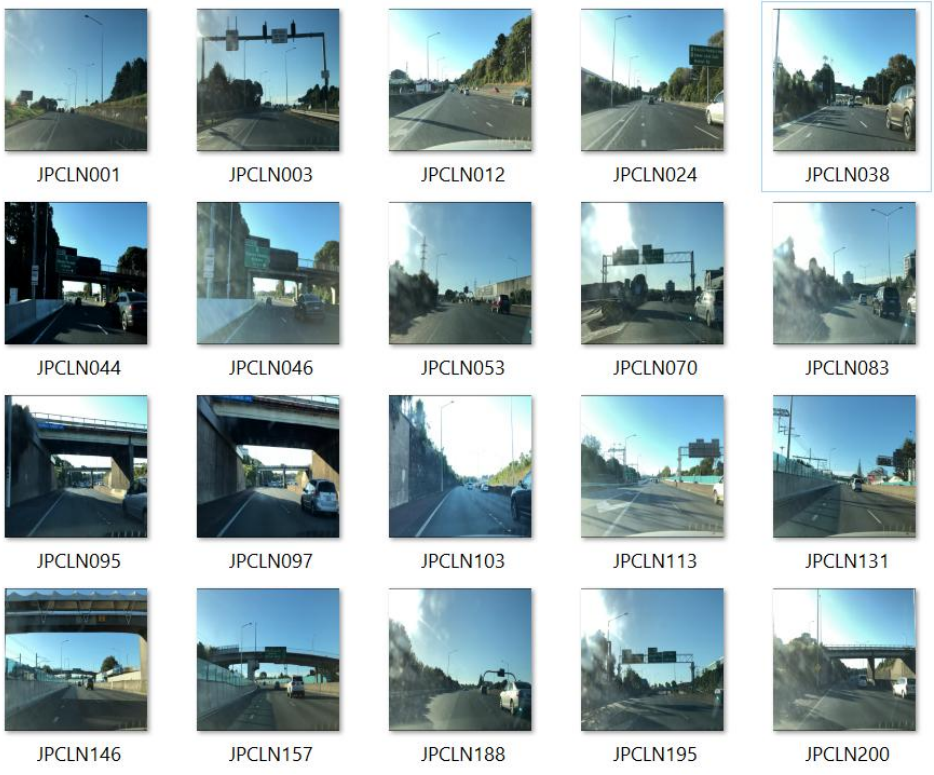


Figure 3.4 The raw images of the training data

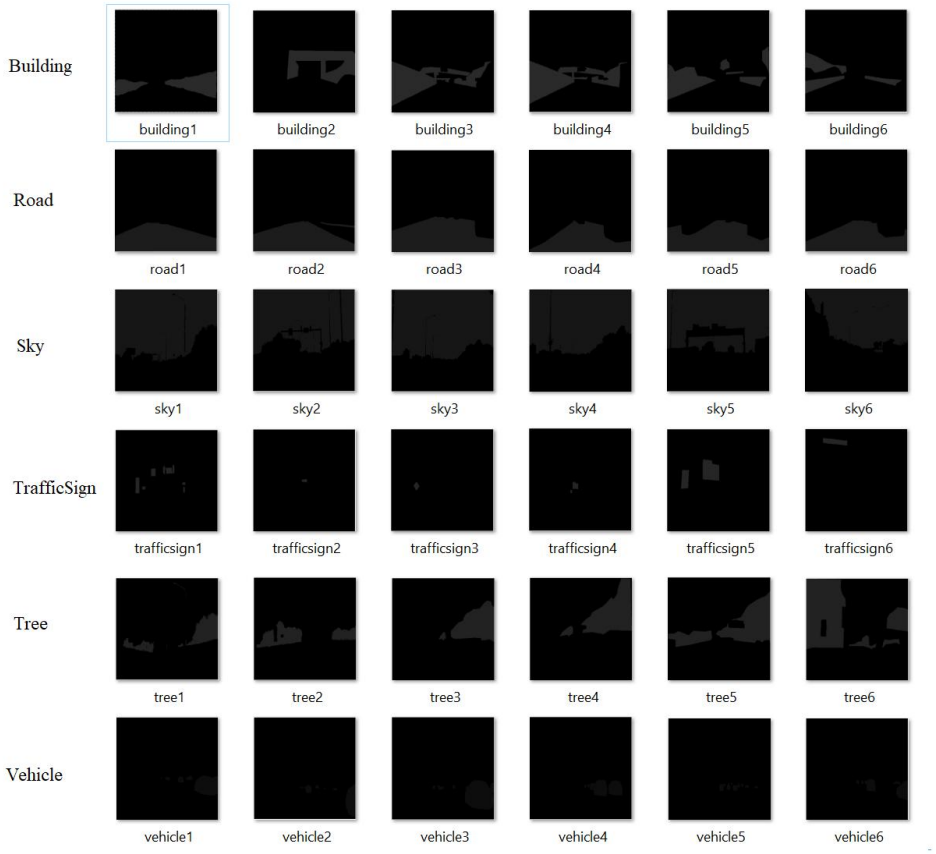


Figure 3.5 The ground truth of the training dataset

The evaluations of the proposed network are conducted using a custom dataset acquired from a vehicle camera on Auckland highways. As shown in Figure 3.4, our dataset comprises of 1,200 images of motorways in Auckland at a resolution of 1024 ×1024.

The ground truth segmentation labels were manually annotated, resulting in six classes: vehicle, sky, tree, building, road, and traffic sign, as depicted in Figure 3.5. Thus, our segmentation model is trained on a dataset containing 1,200 images, which is then divided into a training set and a test set in a ratio of 3:1. For memory and time efficiency, we resize all data to 128 × 128 pixels and apply intensity preprocessing to the dataset. We rescaled the input images to ensure that the pixel values of each image fall within the range of [-1.0, 1.0]. To avoid overfitting, we applied data augmentation

techniques to the dataset, introducing spatial alterations such as translation, scaling, rotation, and elastic deformation, as well as intensity-wise adjustments like translation and scaling of the input images.

3.1.2 Research Design for Depth Estimation

In this thesis, we will use the combination of Transformer and CNN to achieve depth estimation. Transformers are designed to handle long-range dependencies in data, which is essential for depth estimation. In scenes with complex structures and occlusions, information from distant regions can influence the depth prediction. The attention mechanism in transformers allows the model to focus on relevant regions while ignoring irrelevant or noisy parts of the input. This helps improve the accuracy of depth estimation, especially in challenging scenes. Moreover, transformers can aggregate information from the entire input sequence simultaneously. This global context is valuable for depth estimation, as it helps the model understand the overall scene layout and make more informed depth predictions. Transformers use positional encoding to inject spatial information into the input, which helps the model understand the spatial relationships between pixels.

We are use of Swin transformer encoder as the encoder of our proposed depth estimation network. The key idea is to divide the input image into non-overlapping patches, which are then processed using a hierarchical architecture. Instead of processing the patches sequentially, the Swin Transformer employs a two-dimensional shifted window mechanism. It shifts the window over the patches to capture interactions between different spatial locations efficiently. In this way, each patch can attend to a larger contextual area, allowing the model to capture long-range dependencies. The Swin Transformer takes use of a hierarchical design, incorporating multiple stages of transformer blocks. Each stage processes the output of the previous stage, with increasing resolution. The lower stages capture local information, while the higher

stages focus on modeling global dependencies. This hierarchical approach enables the model to process high-resolution images more efficiently than a single-stage transformer (Liu et al., 2021).

Our proposed depth estimation decoder utilizes a CNN decoder, which is composed of a sequence of deconvolutional layers followed by convolutional layers with Rectified Linear Unit (ReLU) activations. The deconvolutional layers are designed with 2×2 kernels and a stride of 2, effectively doubling the input size, while the convolutional layers employ 3×3 kernels and a stride of 1. The network's feature depth progresses through $512 \rightarrow 256 \rightarrow 128 \rightarrow 64$. Lastly, a 1×1 convolutional layer is applied as the final layer to generate the depth map prediction. The network architecture of our proposed depth estimation model is shown in Figure 3.6.

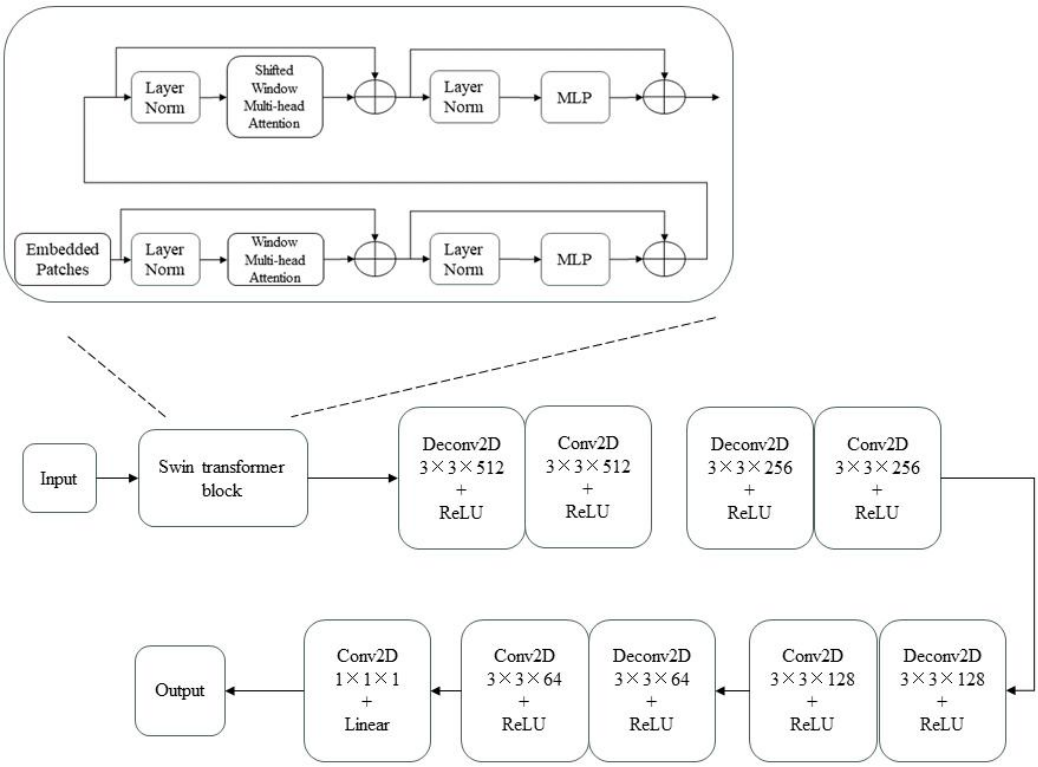


Figure 3.6 The network architecture of depth estimation

3.1.3 Research Design for Vehicle Tracking

Figure 3.7 depicts the functional diagram of the proposed vehicle tracking model. The implementation of the proposed tracking network involves utilizing a modified SiamRPN subnetwork. In contrast to the original SiamRPN, the modified SiamRPN sub-network has been integrated with the Hungarian algorithm (Kuhn, 2012). This integration allows for the advancement of single-target tracking to multi-target tracking.

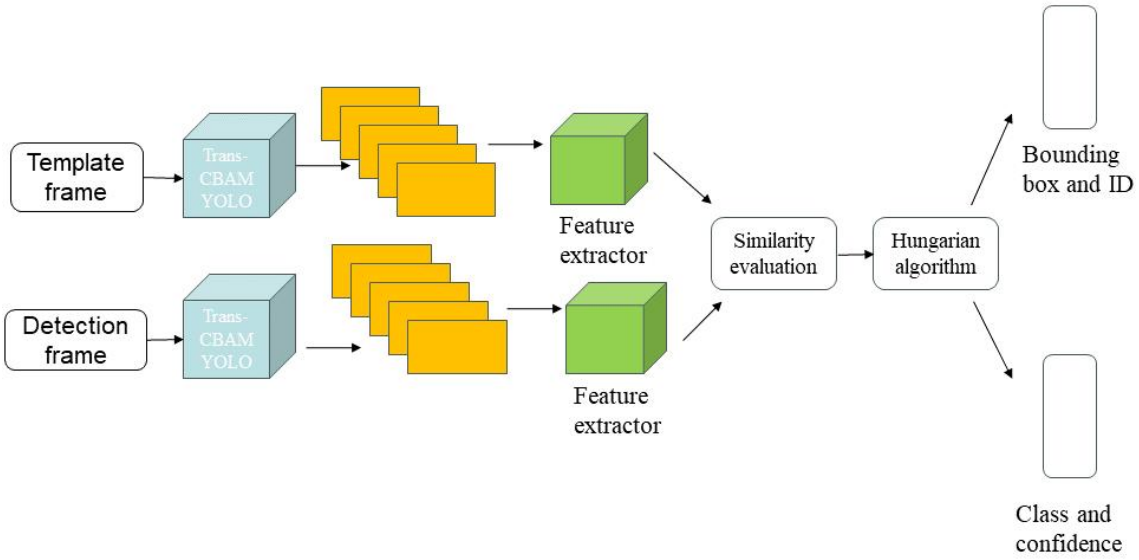


Figure 3.7 A diagram of the vehicle tracking model

The Multiple-object tracking is different from Single-object tracking. The Single-object tracking focuses on tracking a single specific vehicle throughout the video sequence. The goal is to follow the movement of a particular vehicle, keeping it in focus and providing its trajectory over time. Our proposed multiple-object tracking is to detect and track all vehicles present in the scene, assigning unique identities to each vehicle and keeping track of their individual trajectories. The way in this thesis to implement multi-target tracking by using YOLOv5 combined with attention modules and Transformer for object detection to detect all vehicles in each frame and then using Hungarian tracking algorithms to associate the detected objects across frames.

For each detected vehicle in the current frame, calculate the distance between its bounding box and the bounding boxes of all previously tracked vehicles in the previous frame. Use the Hungarian algorithm to find the best assignment of detected vehicles to previously tracked vehicles based on the distance matrix. The Hungarian algorithm efficiently solves the assignment problem, maximizing the total similarity between detected vehicles and tracked vehicles. If a detected vehicle is assigned to an existing track, update the track with the vehicle's new position and other information. If a detected vehicle is not assigned to any existing track, create a new track for that vehicle. To handle occlusions or vehicles leaving the scene, remove any tracks that have not been assigned a detected vehicle for a certain number of frames (An et al., 2021).

To enhance the performance of vehicle detection in traffic scenes, the target detection component of this model combines Transformer, CBAM and YOLOv5 that shown in Figure 3.8. YOLOv5 inherits its structure from the four-part networks YOLOv3 and YOLOv4, consisting of Input, backbone, neck, and prediction stages (Bochkovskiy et al., 2020; Redmon & Farhadi, 2018).

However, YOLOv5 introduces further improvements, including data augmentation at the input side, as well as adaptive anchor frames and adaptive image scaling functions. These enhancements contribute to better accuracy in anchor location and faster inferencing speed (Chuyi et al., 2022; Chienyao et al., 2022).

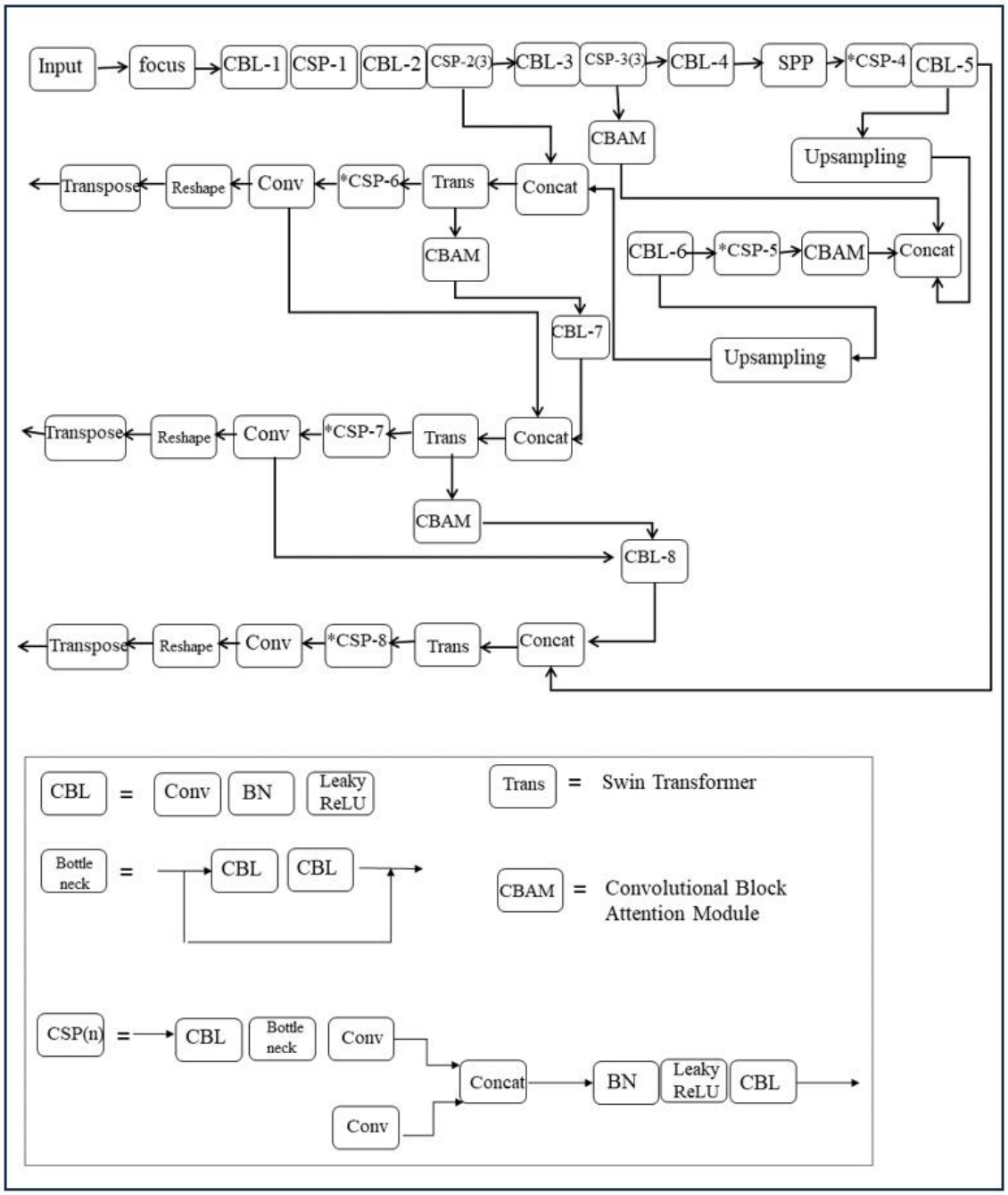


Figure 3.8 The diagram of improved YOLOv5 as used in the proposed method

Similar to YOLOv4, YOLOv5 also employs Darknet as the backbone to extract features from input images (Redmon & Farhadi, 2017). Additionally, YOLOv5 benefits from the Cross Stage Partial Network (CSPNet), which addresses the gradient problem in network optimization for other large-scale CNN frameworks. CSPNet integrates gradient changes into the feature map from start to finish, leading to a reduction in

model parameters and floating-point operations (FLOPs) value. This approach allows for a reduction in model size while maintaining both inference speed and accuracy.

In the proposed YOLOv5-CBAM-Transformer, by learning the four offsets of t_x, t_y, t_w and t_h, the bounding box coordinates obtained by regression are b_x, b_y, b_w, b_h, that is, the positioning and size of the bounding boxes are interconnected with the feature map. Among them, t_x and t_y represent the predicted coordinate offset values, while t_w and t_h denote the scaling factors.:

$$b_x = 2\sigma(t_x) - 0.5 + c_x \tag{3.8}$$

$$b_y = 2\sigma(t_y) - 0.5 + c_y \tag{3.9}$$

$$b_w = p_w(2\sigma(t_w))^2 \tag{3.10}$$

$$b_h = p_h(2\sigma(t_h))^2 \tag{3.11}$$

where c_x and c_y correspond to the coordinates of the upper left corner of the grid cell in the feature map, while P_w and P_h represent the width and height of the predefined anchor box mapped to the feature map.

The major difference in the loss function between YOLOv5 and the previous YOLO series lies in the computation of the positive sample anchor area. The classification and confidence branches utilize Binary Cross Entropy (BCE) loss, while the bbox (Bounding box) branch employs the GIoU loss. BCE loss is well-suited for binary classification tasks, such as determining whether an object is a vehicle or not, and estimating confidence scores associated with the predictions. And GIoU loss takes into account the spatial overlap between predicted bounding boxes and ground truth boxes, leading to better localization accuracy. Moreover, GIoU loss penalizes predicted boxes based on the extent of overlap with ground truth boxes, making it robust to cases where multiple objects overlap. At the same time, GIoU loss considers the difference in sizes and shapes between predicted and ground truth bounding boxes, which is particularly

beneficial for handling variations in vehicle sizes and aspect ratios.

$$\mathrm{BCE}(\hat{c}_i, c_i) = -\hat{c}_i \times \log(c_i) - (1 - \hat{c}_i) \times \log(1 - c_i) \tag{3.12}$$

$$\mathrm{GIoU} = \frac{|A \cap B|}{|A \cup B|} - \frac{|A_c - U|}{|A_c|} \tag{3.13}$$

where A_c represents the minimum overlap area of the two boxes. To consider the aspect ratio of the bounding boxes in the loss function, the CIoU loss is utilized as the boundary regression loss function:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{3.14}$$

$$v = \frac{4}{\pi^2}(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h})^2 \tag{3.15}$$

where b and b^{gt} represent the centre points of the predicted bounding box and the ground truth bounding box, respectively; ρ denotes the Euclidean distance between the two center points, while c represents the diagonal distance of the smallest enclosing area that contains both the predicted box and the ground truth box. Additionally, α is the weight applied in the calculation.

The matching strategy in YOLOv3 ensures that each ground truth bounding box is assigned a unique anchor. The rule dictates that, while guaranteeing the maximum Intersection over Union (IOU), a ground truth box cannot be matched to predictions across all three prediction layers simultaneously. However, this matching strategy does not account for cases where one ground truth bounding box corresponds to multiple anchors, nor does it consider the appropriateness of anchor settings. Consequently, if a ground truth bounding box is associated with multiple anchors, it may slow down the overall model convergence. In this thesis, the approach of augmenting the number of positive sample anchors is used to expedite convergence. This is the key reason why YOLOv5 achieves rapid convergence in practical applications. In contrast to the max IOU matching rule used in previous versions, YOLOv5 abandons this approach for any

output layer. Instead, it directly utilizes the shape rule for matching, wherein the bounding box and the anchor of the current layer are employed to calculate the aspect ratio.When the aspect ratio exceeds the predefined threshold, the object feature is revealed. For the remaining bounding boxes, YOLOv5 identifies the nearest two grids that encompass the box based on which grid it falls into. By applying the rounding rule, these three grids are collectively deemed responsible for predicting the box. By employing this approach, the number of positive samples is roughly estimated to have increased by at least three times compared to the previous YOLO series.

To further enhance the performance of the model, we have incorporated the attention mechanism module and Transformer with YOLOv5. The attention mechanism simulates the internal process of biological observation behavior, where it aligns internal experience with external senses, thereby enhancing the precision of observation in specific areas. For instance, during the processing of an image, human vision promptly scans the global image to identify a target area for concentrated focus, referred to as the focus of attention. Subsequently, a greater allocation of attentional resources is directed towards this region to acquire more comprehensive information regarding the attended target and, simultaneously, to suppress irrelevant information from other regions.

In summary, the attention mechanism assigns distinct weighting parameters to individual elements of the input, thereby intensifying focus on elements that bear similarity to the input and concurrently suppressing superfluous information. Its principal advantage lies in its capacity to simultaneously account for global and local connections in a single step, facilitating parallel computation, a crucial attribute especially pertinent to big data scenarios.

In this experiment, our primary research objective is to enhance the performance of the original YOLOv5 network by incorporating the CBAM in conjunction with Swin Transformer (Woo et al., 2019; Liu et al., 2021). The integration of CBAM facilitates

the network in determining what and where to focus when analyzing intricate traffic environments by leveraging both spatial and channel feature connections. The network's neck is responsible for reprocessing crucial environmental features extracted from the backbone, transmitting them to the head, and subsequently producing prediction outcomes. To achieve this, we strategically insert the CBAM module after each Concatenation operation, refining the information of the channel and spatial feature fusion layer. This refinement assists the model in allocating greater attention to key information within the complex traffic environment. The experiment aims to ascertain whether the inclusion of CBAM can lead to improved performance compared to the original YOLOv5.

The architecture of the CBAM attention mechanism module consists of two main components: spatial attention and channel attention. Upon receiving the feature map as input, it undergoes the channel attention process. Global Average Pooling and Global Max Pooling operations are performed based on the width and height of the feature map. Subsequently, the channel attention weight is obtained through the Multilayer Perceptron (MLP), and further normalized using the Sigmoid function. Finally, the original input feature map is recalibrated channel by channel through element-wise multiplication, completing the channel attention-based feature recalibration

In pursuit of attention features in the spatial dimension, the feature map derived from channel attention undergoes both global maximum pooling and global average pooling operations, resulting in a transformation of the feature dimension from H×W to 1×1. Afterward, the dimension of the feature map is reduced via convolution with a 7×7 kernel, followed by the application of the ReLU activation function. Subsequently, the feature map is restored to its original dimension through another convolutional operation, culminating in the completion of the feature map's recalibration process.

Within the spatial attention module, spatial attention features are obtained using

global average pooling and maximum pooling techniques. The establishment of spatial feature correlations is achieved through two convolutional operations, ensuring that the input and output dimensions remain unchanged. The utilization of a 7×7 convolutional kernel significantly reduces the parameters and computational complexity, facilitating the establishment of high-dimensional spatial feature correlations. Following the application of CBAM, the new feature map acquires attention weights in both the channel and spatial dimensions. This improvement significantly enhances the interconnection between each feature in the channel and space, thereby promoting the extraction of effective target features.

For comparison with CBAM, we also explored the Squeeze-and-Excitation (SE) attention mechanism (Hu, Shen & Sun, 2018). The SE attention mechanism was introduced to address the issue arising from the varying significance of different channels within a feature map during the convolutional pooling process. In conventional convolutional pooling, each channel of a feature map is inherently considered equally important. However, in practical scenarios, the significance of different channels varies.

The SE attention enhancement model focuses on the object in two steps: squeeze and excitation. Global average pooling of channel information for the input feature map (squeeze). Conditional on input *x*, the squeeze step for the *c*-th channel can be expressed as

$$Z_c = \frac{1}{H*W}\sum_{i=1}^{H}\sum_{j=1}^{W} x_c(i,j) \tag{3.16}$$

in the equation, Z_c represents the output of the *c*-th channel. The input *x* originates from a convolutional layer with a predetermined kernel size, and the squeeze operation enables the model to gather global information.

The information after squeeze is multiplied onto the input feature map by two fully connection layers, the activation function and then normalized. The aim of the excitation

is to completely capture the dependencies between channels:

$$Y = X * \sigma(\overline{Z}) \tag{3.17}$$

$$\overline{Z} = T_2(ReLU)(T_1(Z)) \tag{3.18}$$

where T_1 and T_2 are two linear transforms that capture the importance of each channel through learning.

SE is incorporated into YOLOv5 using two approaches: firstly, attention is added to the final layer of the backbone, and secondly, all occurrences of C3 in the backbone are replaced. In the discussion section, we will conduct a detailed analysis of the impact of both approaches on the results.

The Coordinate Attention (CA) is the second attention mechanism we tested. CA disassembles channel attention into two one-dimensional feature encoding processes, each designed to gather features along two distinct spatial orientations. As a result, this configuration enables the capturing of long-range dependencies along one spatial direction while preserving precise positioning information along the other spatial direction. The generated feature maps are encoded as a pair of direction-aware and location-sensitive attention maps, which can be combined with the input feature maps to enhance the representation of the target object (Hou et al., 2021).

The incorporation of the CA attention mechanism involves two steps: the embedding of the coordinate message and the generation of the coordinate attention. Given an input *X*, each channel undergoes encoding along the horizontal and vertical coordinates through pooling kernels of size (*H*, 1) or (1, *W*), respectively. Consequently, the output of channel *c* with height 1.0 can be represented as:

$$Z_c^h(h) = \frac{1}{W}\sum_{0 \le i \le W} x_c(h, i) \tag{3.19}$$

The output of channel c with width W can be expressed as:

$$Z_c^w(w) = \frac{1}{H}\sum_{0\le i\le H} x_c(w, i) \tag{3.20}$$

The transformations yield a pair of direction-aware feature maps by consolidating data along each of the two spatial directions. This is distinct from the singular feature vector generation approach employed in the channel attention (SE) method. Through these modifications, the attention module is capable of capturing long-term dependencies along one spatial direction while retaining precise positional information along the other spatial direction. As a result, the network becomes more adept at accurately identifying the object of interest.

After undergoing the information embedding transformation, this section performs a concatenation operation on the aforementioned transformations. Subsequently, the concatenated output is further processed using the 1×1 convolutional transformation function:

$$f = \delta\left(F_1([Z^h, Z^w])\right) \tag{3.21}$$

where [,] denotes the concatenate operation along the spatial dimension, δ represents the non-linear activation function, and f denotes the intermediate feature mapping that encodes spatial information in both horizontal and vertical directions. Furthermore, the other two 1×1 convolutional transformation functions F_1, f^h and f^w are employed to transform into tensors with the same number of channels as the input X, respectively:

$$g^h = \sigma(F_h(f^h)) \tag{3.22}$$

$$g^w = \sigma(F_w(f^w)) \tag{3.23}$$

Output Y:

$$y_c(i,j) = x_c(i,j) * g_c^h(i) * g_c^w(j) \tag{3.24}$$

Due to the Transformer's pronounced ability to capture global information, it exhibits superior performance in comprehending dense and occluded objects within intricate traffic environments. Consequently, we integrate the Swin Transformer's encoder into the head of each YOLOv5, effectively combining the two networks.

The primary technological breakthrough in the Swin Transformer lies in its adoption of localization and shifted windows. By employing non-overlapping windows for self-attention computation, localized self-attention is computed within each scale feature map's window. However, the computation between different scales has remained localized, leading to a deficiency in information interaction between windows. To overcome this limitation, the Swin Transformer incorporates shifted windows at varying levels, encompassing both W_MSA (Window Multi-head Self-Attention) and SW_MSA (Shifted Window Multi-head Self-Attention) (Liu et al., 2021).

We assume that the feature map passed into Swin Transformer Block is Z^{l-1} which passes through LayerNorm and MSA and then adds with $\mathrm{Z}^{\mathrm{l}-1}$ and adds to get $\widehat{Z}^l$. After passing through a LayerNorm and MLP, $\hat{\mathrm{Z}}^{\mathrm{l}}$ is directly connected to $\hat{\mathrm{Z}}^{\mathrm{l}}$ and added to obtain Z^{l}:

$$\widehat{Z}^l = W - MSA(LN(Z^{l-1})) + Z^{l-1}, \tag{3.25}$$

$$Z^l = MLP(LN(\widehat{Z}^l)) + \widehat{Z}^l, \tag{3.26}$$

$$\hat{\mathrm{Z}}^{\mathrm{l}+1} = \mathrm{SW} - \mathrm{MSA}(\mathrm{LN}(\mathrm{Z}^{\mathrm{l}})) + \mathrm{Z}^{\mathrm{l}}, \tag{3.27}$$

$$\mathrm{Z}^{\mathrm{l}+1} = \mathrm{MLP}(\mathrm{LN}(\hat{\mathrm{Z}}^{\mathrm{l}+1})) + \hat{\mathrm{Z}}^{\mathrm{l}+1}, \tag{3.28}$$

The regression modified SiamRPN, integrated with YOLOv5-CBAM-Transformer, assumes the role of target tracking. Within this model, the region proposal network (RPN) comprises two branches, each responsible for foreground and background

classification, as well as proposal regression, respectively.

Siamese-RPN employs a fully connected CNN without padding. The network consists of two branches: the template branch, which takes the target patch from the historical frame as input, and the detection branch, which uses the target patch from the current frame as input (Xu, Zhang & Brownjohn, 2021). To ensure compatibility with subsequent tasks, both branches share the same parameters in the CNN. We denote the feature maps of these two branches as $\varphi(x)$ and $\varphi(z)$, respectively (Zhang et al., 2021).

Similar to the RPN network in Faster R-CNN, when there are k ($k > 0$) anchors, the network is required to produce a channel map with dimensions $2k$ for object classification and a $4k$ channel map for anchor regression. Consequently, $\varphi(z)$ is initially split into two branches, $[\varphi(z)]_{cls}$ and $[\varphi(z)]_{reg}$, through two separate convolution operations, corresponding to $2k$ and $4k$ channels, respectively. Similarly, $\varphi(x)$ is also divided into two branches, $[\varphi(x)]_{cls}$ and $[\varphi(z)]_{reg}$, using convolution operations. The channels in $\varphi(x)$ remain unchanged, while a specialized convolution operation is applied using $[\varphi(z)]_{cls}$ and $[\varphi(z)]_{reg}$ as the convolution kernels. Convolution operations are performed on the feature maps $[\varphi(x)]_{cls}$ and $[\varphi(x)]_{reg}$, respectively. Finally, the outputs after the convolution consist of 17×17×2k and 17×17×4k channels.

$$A^{cls}_{w\times h\times 2k} = [\varphi(x)]_{cls} \star [\varphi(z)]_{cls} \tag{3.29}$$

$$A^{reg}_{w\times h\times 4k} = [\varphi(x)]_{reg} \star [\varphi(z)]_{reg} \tag{3.30}$$

where $\star$ represents the convolution operation. The final classification branch outputs a feature map with $2k$ channels, k is the number of anchors. This feature map will be grouped into pairs and split into k groups, each group has a score map with two channels that represents the scores of the foreground and the background, respectively. In a similar way, for the regression branch, the final output of $4k$ channel feature maps, each of which has a group with 4 channels that represents the center position and size of the

anchor, which is also divided into k groups.

Considering tracking as a one-shot detection, where z represents the template part and x denotes the detection part, the Siamese feature extraction subnet is represented by the function $\varphi(\bullet)$, and the RPN subnet is denoted by function $\zeta(\bullet)$. The one-shot detection can be expressed as follows:

$$\min_{W} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\zeta\big(\varphi(x_i; W); \varphi(z_i; W)\big), \ell_i) \tag{3.31}$$

Regarding proposal selection, SiamRPN utilizes the cosine window and scale change penalty to reorganize the proposal scores, ultimately obtaining the optimal proposal. Following the elimination of outliers, a cosine window is applied to mitigate large displacements, while a penalty is incorporated to suppress substantial changes in size and ratio.

$$Penalty = e^{k * \max\left(\frac{r}{r'}, \frac{r'}{r}\right) * \max\left(\frac{s}{s'}, \frac{s'}{s}\right)} \tag{3.32}$$

where k denotes a hyperparameter, r represents the aspect ratio of the proposal, r' denotes the height to width ratio of the last frame, and s and s' respectively signify the overall scale of the proposal and the last frame (Li et al., 2018; Rao et al., 2020).

Data association is a critical process in multitarget tracking, primarily focused on matching multiple targets between frames. This entails tasks such as identifying new targets, tracking disappearing targets, and matching vehicle IDs between the previous and current frames. Existing data association methods often draw upon operations research approaches. In this thesis, we adopt the classic Hungarian matching algorithm to accomplish the task of multi-target vehicle tracking. Below, we provide a step-by-step explanation of Hungarian matching algorithm:

Step 1. Determine the smallest element in each row of the matrix and subtract the

corresponding smallest element from each element in that row.

Step 2. Check if the algorithm's objective has been achieved. If not, proceed to the next step; otherwise, terminate.

Step 3. Determine the smallest element in each column of the modified matrix and subtract the corresponding smallest element from each element in that column.

Step 4. Cover all the 0s using the fewest possible vertical and horizontal lines.

Step 5. Check if the number of covered rows equals the order of the matrix. If so, the optimal matching solution is obtained. Otherwise, proceed to step 6.

Step 6. Find the minimum value in the uncovered portion. Subtract this value from each element in the uncovered rows and add it to each element in the covered columns. Return to step 4.

This algorithm is applied in the multitarget tracking problem to find the optimal matching solution for multiple targets in two frames (Kuhn, 2012).

3.1.4 Research Design for Distance Estimation

In our distance estimation model, we adopt YOLOv7 as the foundational architecture. Additionally, we integrate Swin Transformer and CBAM into the model to enhance the feature extraction capabilities further (Woo et al., 2019; Chienyao, Alexey & Mark, 2022; Liu et al., 2021).

The design of YOLOv7 aims to address two specific challenges. Firstly, it introduces the concept of gradient propagation routes, which facilitates a structured model re-referencing approach. This approach enables the analysis of structural re-referencing techniques that are pertinent to each network layer.

Training models with multiple output layers using a dynamic label assignment

technique poses additional challenges, particularly concerning the assignment of dynamic targets to the outputs of different branches. To address this challenge, a novel approach is proposed called the coarse-to-fine guided label assignment technique for labeling assignment. This technique offers a solution to overcome the issue of assigning dynamic targets to the outputs of various branches.

Our model, as illustrated in Figure 3.9, incorporates the CBAM to enhance the feature extraction process, thereby avoiding alterations to the original feature extraction (Kailin et al., 2022). Moreover, the inclusion of the Transformer enhances the model's capacity to comprehend global semantics, allowing YOLOv7, which primarily emphasizes local information processing, to achieve a more comprehensive understanding of traffic scenes. The CBAM encompassing the Channel Attention (CAM) and the Spatial Attention Module (SAM) (Woo, Park, Lee & Kweon, 2019). The CAM computes attention maps by analyzing the interdependencies among channels, determining which channels contain significant information for a given context. The channel attention mechanism allows the network to emphasize important channels while reducing the influence of less relevant ones. On the other hand, the SAM computes attention maps to highlight spatial regions containing relevant information. This mechanism enables the network to adaptively focus on specific parts of the image while downplaying background or less informative areas. The CAM focuses on capturing channel-wise relationships within the feature maps. It consists of an MLP followed by an element-wise summation and a sigmoid activation function. The MLP processes the input feature map and computes channel-wise attention weights that highlight informative channels and suppress less relevant ones. The attention weights obtained from the MLP are scaled using the sigmoid activation function to ensure they lie within the range of 0 to 1. The scaled attention weights are then applied element-wise to the original feature map to enhance informative channels,

$$M_C(F) = \sigma\left(MLP\big(AvgPool(F)\big) + MLP\big(MaxPool(F)\big)\right) = \sigma(W_1\left(W_0\big(F^c_{avg}\big)\right) +$$

$$W_1\big(W_0(F^c_{max})\big)) \tag{3.33}$$

where σ represents the sigmoid function, while W_0 and W_1 are the shared MLP weights for both inputs. $W_0 \in \mathbb{R}^{C/r \times C}$, and $W_1 \in \mathbb{R}^{C \times C/r}$. The ReLU activation function is applied following these weights.

The SAM analyzes spatial relationships within the feature maps to highlight important spatial regions. It involves two operations: max pooling and convolution. Max pooling captures global context by summarizing the most significant spatial information within each channel. Convolution captures local context by processing the feature map with a convolutional filter to capture spatial dependencies. The outputs of max pooling and convolution are combined using an element-wise summation. The combined outputs undergo a sigmoid activation to produce spatial attention weights. The spatial attention weights are applied to the feature map to highlight relevant spatial regions (Li et al., 2023),

$$M_s(F) = \sigma\big(f^{7\times7}([AvgPool(F); MaxPool(F)])\big) = \sigma(f^{7\times7}([F^s_{avg}; F^s_{max}])) \tag{3.34}$$

where σ denotes the sigmoid function and $f^{7\times7}$ represents a convolution operation with the filter size of 7×7.

CBAM has demonstrated its effectiveness in improving the discriminative power of CNNs and boosting their accuracy on various benchmarks and datasets. Its ability to adaptively emphasize salient features while suppressing irrelevant ones helps networks achieve better generalization and robustness, making CBAM a valuable tool for advancing the capabilities of deep learning models in the field of computer vision.

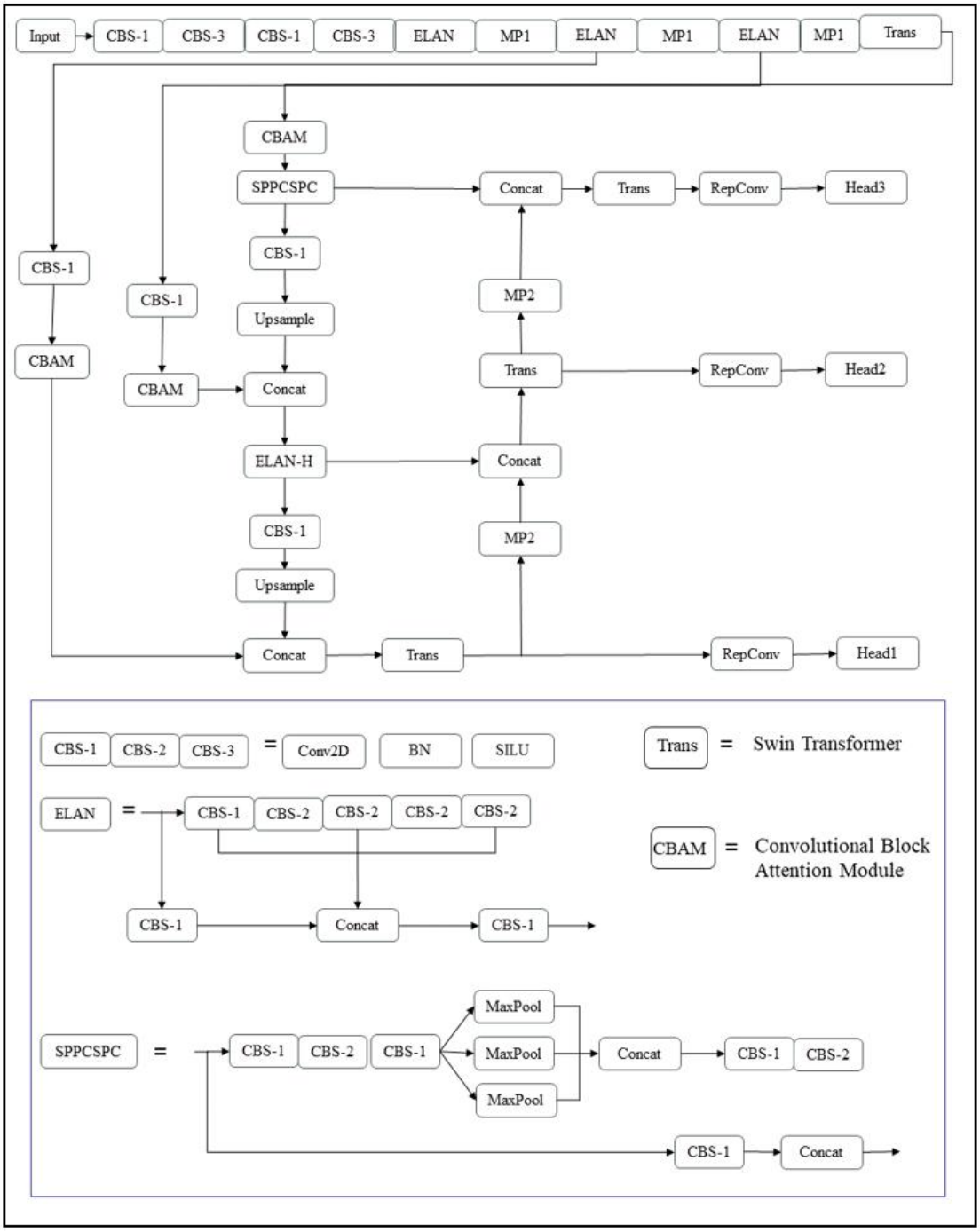


Figure 3.9 The architecture of YOLOv7-CBAM-Transformer

Nonetheless, the YOLOv7 network holds a significant advantage in extracting foundational features and visual structures. These low-level features encompass essential points, lines, and fundamental image elements at the patch level. These fundamental features exhibit distinct geometric characteristics and often emphasize

consistency or covariance under transformations such as translation and rotation. Once the fundamental visual components are identified, the emphasis shifts towards comprehending the advanced visual meaning. This centers on grasping the interconnections among these components, shaping them into objects, and perceiving how the spatial arrangement of objects generates a scene. Presently, the Transformer model is widely regarded as proficient and efficient in managing the intricate relationships among these components. As a result, we remove the last ELAN in the YOLOv7 backbone and some ELANs in the neck and instead integrate the Swin Transformer encoder. By implementing this operation, we can accentuate the benefits of the self-attention mechanism while simultaneously reducing computational overhead. Moreover, the introduction of Transformer in the neck allows for capturing correlations and significance between different regions, enhancing the model's ability to adapt to targets of varying sizes (Zhang, 2023).

As we implement YOLOv7-CBAM-Transformer, our goal is not only to detect the position of the vehicle but also to estimate the distance between the vehicle in front and the current position. To achieve this, we have extended the prediction vector to incorporate distance estimation information.

The original prediction vector contains bounding box anchor coordinates A (*x, y, w, h*) and category confidence C (c_1, c_2). In order to make the model realize the ranging function, we add the distance element D (*d*) to the prediction vector. The extended prediction vector is shown in the Figure 3.10.

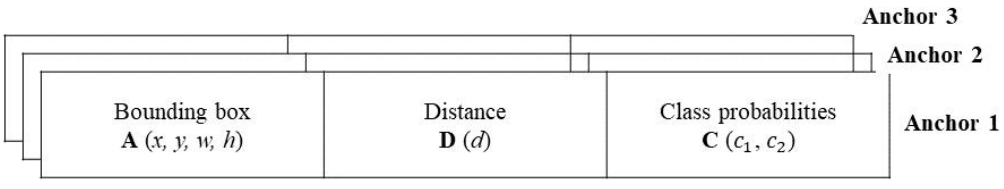


Figure 3.10 The extended prediction vector for distance estimation

The distance loss is defined as

$$l_{distance}(i,j) = \omega(d'_{i,j} - d_{i,j})^2 = \omega \sum_{k=0}^{c} C_{i,j,k}(d'_{i,j,k} - d_{i,j,k})^2 \quad (3.35)$$

where $C_{i,j,k}$ is k-th class probability in (i, j)-th cell. The weighting constant ω is introduced to balance the importance of the distance loss with other losses, preventing it from dominating the overall training process. In our experiment, we set ω to a value of 1×10^2.

3.2 Evaluation Methods

3.2.1 Evaluation Methods of Semantic Segmentation

The performance of the segmentation model is evaluated using the Intersection over Union (IoU) metric, aiming to evaluate the quality of semantic segmentation results by quantifying the alignment between predicted and ground truth masks.

In Table 3.1, p and p' represent actual and predicted positive samples, respectively. Likewise, n and n' represent actual and predicted negative samples. P denotes the total number of actual positive instances in the dataset, while N represents the total number of actual negative samples.

- *TP* is True Positive, judged as a positive sample, in fact also a test sample.
- *TN* is True Negative, judged as a negative sample, in fact a negative sample.
- *FP* is False Positive, judged as a positive sample, is actually a negative sample.
- *FN* is False Negative, judged as a negative sample, but in fact a positive sample

Table 3.1 Form of classification criteria

Actual / Predicted		Actual	
		p	n
Predicted	p'	TP	FP
	n'	FN	TN
Total		P	N

Mathematically, IOU is calculated as the ratio of the intersection of the predicted and ground truth masks to their union. Equation (3.36) outlines the computation of IoU as follows:

$$IoU = TP / (TP + FP + FN). \quad (3.36)$$

3.2.2 Evaluation Methods of Vehicle Tracking

In our experiments, we utilize the multitarget tracking accuracy (MOTA) and multiobject tracking precision (MOTP) to assess the algorithm's ability to continuously track objects. MOTA measures the accuracy and effectiveness of object tracking algorithms, particularly in the context of computer vision tasks like multiple object tracking. It quantifies how well an algorithm is able to track objects over time, taking into account various factors that contribute to tracking errors. It accounts for irregular accumulations in tracking, including false positives (FP) and false negatives (FN), among others. MOTA focuses on accurately judging multiple objects in consecutive frames and precisely determining their positions to achieve uninterrupted continuous tracking.

$$MOTA = 1 - \frac{FN+FP+ID_{sw}}{Total\ number\ of\ objects} \tag{3.37}$$

where ID_{sw} means identity switches occur when the algorithm associates two objects with each other incorrectly, resulting in tracking errors.

On the other hand, MOTP focuses on measuring the accuracy of the predicted object positions by quantifying the average displacement between the predicted and the corresponding ground truth positions of tracked objects. The formula to calculate MOTP involves summing up the distances between each predicted object position and its corresponding ground truth position and then dividing by the total number of matched object pairs:

$$MOTP= \frac{1}{N_{match}}\sum_{i=1}^{N_{match}}\frac{d_i}{n_i} \tag{3.38}$$

where N_{match} is the total number of matched object pairs; d_i is the Euclidean distance between the predicted position and the corresponding ground truth position for the *i*-th object pair; n_i is the number of frames in which the *i*-th object pair is present.

3.2.3 Evaluation Methods of Depth Estimation and Distance Estimation

Our depth estimation model is assessed using various metrics, including root mean squared error (RMSE), absolute relative error, and squared relative error, and delta accuracy. However, for the main evaluation of our distance estimation, we primarily focus on the RMSE：

$$ABsRel = \frac{1}{N}\sum\frac{|d_i-d_i^*|}{d_i} \tag{3.39}$$

$$SqRel = \frac{1}{N}\sum\frac{|d_i-d_i^*|^2}{d_i} \tag{3.40}$$

where d_i and d_i^* are the ground truth and predicted depth at pixel *i* and *N* is the total number of pixels.

$$RMSE=\sqrt{\frac{1}{N}\sum_{i=1}^{N}|d_i - d_i^*|^2} \quad (3.41)$$

$$RMSE\_log = \sqrt{\frac{1}{N}\sum_{i=1}^{N} log(d_i) - log(d_i^*))^2} \quad (3.42)$$

where d_i is the real depth information, d_i^* is the predicted depth value.

The delta accuracy δ_i of d_i , $max(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}) = \delta < t$ for threshold values *t = 1.25,* $1.25^2, 1.25^3$;

Select the maximum value of $\frac{d_i}{d_i^*}$ and $\frac{d_i^*}{d_i}$, and calculate the percentage of pixels in d_i that are smaller than the threshold *t* in the total pixels. The closer it is to 1.0, the better the effect, so the larger the threshold, that is, the result obtained when the threshold is 1.25^3 is better. Better than the result with a threshold of 1.25

3.3 The Originality of This Thesis

The originality of scene segmentation:

(1) The datasets used for scene segmentation were specifically collected and labeled for New Zealand traffic scenes, ensuring a more suitable model for local environments.

(2) This work represents the application of CapsNet for scene segmentation for understanding higher level semantic, showcasing outstanding performance in this context.

The originality of depth estimation:

(1) The use of Swin Transformer as an encoder for depth estimation models, contributing to improved performance.

The originality of vehicle tracking:

(1) The datasets employed for vehicle tracking were uniquely created and annotated for New Zealand traffic scenes, resulting in a more tailored model for local conditions.

(2) Combination of attention modules (CBAM, SE, CA) to YOLOv5, enhancing local feature extraction capabilities, combined with Transformer to improve global feature understanding and adaptability to objects of different sizes.

(3) Utilization of SiamRPN along with the Hungarian algorithm for multi-objective vehicle tracking, enabling efficient and accurate tracking of multiple targets.

The originality of distance estimation:

(1) Customized datasets of New Zealand traffic scenes were used for distance estimation, ensuring a model suitable for local scenes.

(2) Incorporation of the attention module (CBAM) and Transformer into YOLOv7, leading to improved global feature comprehension and adaptability to objects of varying sizes. We also added extented distance estimation vector to the model for distance regression.

Chapter 4 Results

In this chapter, we will present the comprehensive methodology for vehicle-related scene understanding. Here, we will establish the experimental environment and discuss the results of scene segmentation, vehicle tracking, depth estimation and distance estimation. The chapter aims to provide a thorough explanation of the methodologies and the obtained outcomes for each aspect of the study.

4.1 Experimental Parameters and Environment

4.1.1 Experimental Parameters and Environment for Semantic Segmentation

Coping with the substantial number of parameters that require processing is among the most formidable challenges when training a deep learning model. These parameters encompass various aspects, such as the width and depth of the network, the connection patterns, and the design of hyperparameters, including the critical task of debugging the loss function. In addition to the parameters mentioned earlier, other crucial factors such as learning rates, batch sample sizes, and optimizer settings also play a significant role. The substantial number of parameters in the network model can directly or indirectly influence its effective tolerance during the training process. Enhancing each of these factors individually can be a time-consuming and resource-intensive task.

Thus, it is essential to prioritize sensible debugging based on hyperparameters to optimize the model effectively and efficiently. The main focus of this work centres on monitoring and understanding the traffic environment surrounding the vehicle. It employs the Intel Core i7-8550 processor to train deep neural networks, and the system is equipped with 8 GB of RAM for efficient file storage. In this thesis, digital photos are pre-processed and network refinement is conducted using the MATLAB 2019a environment. To train the neural network, specific parameters listed in Table 4.1 are established.

Ensuring rapid convergence of the loss function during the training phase critically relies on selecting an appropriate initial weight for the network. Nonetheless, when randomly initializing the network weights, there is no assurance that the starting weights will be in a favourable state for each initialization. In the event of an issue with the initial weight setting, it is possible that the loss function might converge to a local minimum. Hence, selecting an appropriate initial weight is crucial for achieving a global

optimum model. To address this concern, the AdaDelta optimizer is employed in this model. AdaDelta, an extension of AdaGrad, was introduced by Zeiler in 2012 to tackle the issue of AdaGrad monotonically decreasing learning rate. Unlike AdaGrad, which computes all gradient squares, AdaDelta limits the window size for calculating previous gradients. Furthermore, AdaDelta eliminates the need to set a default learning rate in the update rule, thus accelerating model training in the short- and medium-term. With the AdaDelta optimizer, the loss fluctuations around local minimums are managed effectively during late-stage training.

The learning rate is a crucial hyperparameter in model training that significantly impacts the training process. In general, using an appropriate learning rate or a set of well-chosen learning rates can lead to faster model training and improved accuracy. An excessively high or low learning rate directly impacts the model's convergence. A very low learning rate results in slow training as the network's weights are updated only minimally. Conversely, a very high learning rate may cause training to diverge or result in unstable and erratic changes in the model's parameters. Striking the right balance in setting the learning rate is crucial for effective model training. Nevertheless, an excessively high learning rate can have undesirable effects on the loss function. In this thesis, we have set the initial learning rate to 1.

Choosing a right Max epoch is also a importance for training a good deep learning model. Training a deep learning model for too many epochs can lead to overfitting. Overfitting occurs when the model becomes too specialized in the training data and fails to generalize well to new, unseen data. By limiting the number of epochs, you can prevent the model from memorizing the training data and promote better generalization. Also, training deep learning models can be computationally expensive and time-consuming. Choosing the appropriate number of epochs ensures that you strike a balance between training the model long enough to learn meaningful patterns and avoiding excessive computational costs.

While the learning rate directly influences the model's convergence state, the batch size plays a significant role in the model's generalization performance. Larger batch sizes can help reduce gradient noise, leading to more stable training and smoother convergence of the model. However, very large batch sizes might result in suboptimal generalization. Moreover, larger batch sizes can lead to faster training since the model processes more samples in parallel. However, excessively large batch sizes might slow down convergence or cause memory constraints. Also, smaller batch sizes can promote better generalization as they introduce more noise to the optimization process, preventing the model from overfitting to the training data. At the same time, smaller batch sizes might help the model avoid getting trapped in local minima, making it more likely to find better global minima during training.

CheckpointPath is configured to a temporary location, where the network checkpoint is automatically saved after the completion of each training iteration. This process ensures that the model's progress is periodically saved, providing a safety net in case of unexpected interruptions such as system failures or power losses during training. If training is halted unexpectedly, the stored checkpoint can be utilized to restore and continue the training process from the most recent point of progress, thereby preventing the loss of valuable training data and facilitating smoother training resumption.

Table 4.1 Training Parameters

Parameters	*Description and Setting*
Adadelta	The last parameter update contributed to the current iteration of the stochastic gradient descent with Adadelta Optimizer.
Initial Learn Rate	Initial learning rate for training. If the learning rate is too low, it will waste too much time. If the learning rate is too high, the training will not reach the optimal level. It is set to 1.
Max Epochs	The maximum number of epochs for training. The epoch representation model completes a training throughout the training set. It is set to 30000.
Batch Size	Size of the batch to use for each training iteration. It is set to 1.
Checkpoint Path	Path for saving the checkpoint networks. Set it to *root path of* this project.

Throughout the training process, we carefully monitored the progress of epochs and the associated loss for our four tasks. Figure 4.1 presents the data spanning

iterations 100 to 1900 of semantic segmentation. As the training iterations progressed, we observed a consistent decrease in the loss, indicating that the model was gradually improving its performance and optimizing its parameters.

```
03:44:02: train iter: 100 loss: 0.0709
03:45:29: train iter: 200 loss: 0.0416
03:46:56: train iter: 300 loss: 0.0336
03:48:23: train iter: 400 loss: 0.0309
03:49:50: train iter: 500 loss: 0.0287
03:51:18: train iter: 600 loss: 0.0274
03:52:45: train iter: 700 loss: 0.0236
03:54:12: train iter: 800 loss: 0.0174
03:55:38: train iter: 900 loss: 0.0148
03:57:05: train iter: 1000 loss: 0.0161
03:58:32: train iter: 1100 loss: 0.0141
03:59:59: train iter: 1200 loss: 0.0142
04:01:26: train iter: 1300 loss: 0.0148
04:02:53: train iter: 1400 loss: 0.0135
04:04:20: train iter: 1500 loss: 0.0129
04:05:47: train iter: 1600 loss: 0.0125
04:07:13: train iter: 1700 loss: 0.0123
04:08:40: train iter: 1800 loss: 0.0124
04:10:07: train iter: 1900 loss: 0.0118
```

Figure 4.1 Partial training data

4.1.2 Experimental Parameters and Environment for Depth Estimation

The depth estimation experiment was conducted using Microsoft Windows 10 Operating System, with the PyTorch deep learning development framework and Python as the primary development language. The depth estimation experiment utilized the Intel CORE i7 CPU, with the programming tool Google Colab, leveraging its own K80 GPU acceleration. During training, the Adam optimizer was employed with a batch size of 1. The initial learning rate was set to 0.0001, and the training process was conducted for 30 epochs. This configuration allowed for efficient training and evaluation of the depth estimation model, making use of both CPU and GPU resources to achieve accurate and timely results.

4.1.3 Experimental Parameters and Environment for Vehicle Tracking

The vehicle tracking experiment utilized MS Windows 10 operating system along with the PyTorch deep learning development framework, employing Python as the primary programming language. These tools provided a robust and versatile environment for implementing and evaluating the vehicle tracking model, enabling efficient processing and analysis of traffic scenes and object tracking. Intel CORE i7 CPU was employed, and Google Colab served as the programming tool with its own K80 GPU acceleration, ensuring efficient computation and model training. Adam optimization algorithm was utilized during training, with a batch size of 16 and adjusted beta1 parameter for momentum and beta2 parameter set to 0.999. The initial learning rate was set to 0.0001, and the model underwent training for a total of 150 epochs, optimizing the tracking performance over multiple iterations.

4.1.4 Experimental Parameters and Environment for Distance Estimation

For the distance estimation experiment, we employed the Windows10 operating system along with the PyTorch deep learning development framework and Python as the programming language. The experiment utilized an Intel CORE i7 CPU and made use of Google Colab as the programming tool, benefiting from its own K80 GPU acceleration for efficient computation. We set the hyperparameters as follows: iterations=5000, batch_size=1, and learning_rate=0.01 to effectively train our modified YOLOv7 model for accurate distance estimation.

4.2 Experimental Results

Deep learning was utilized to achieve comprehensive scene understanding in both 2D and 3D aspects in our research. A series of experiments were conducted to assess the effectiveness of deep learning in various tasks, including scene segmentation, vehicle

tracking, distance estimation, and depth estimation. The outcomes of these experiments showcase the remarkable advantages of employing deep learning for enhancing scene understanding and analysis. In particular, by incorporating CapsNet and introducing Transformer, attention modules, and regression modifications adding to CNN, we aim to address the limitations of traditional CNN models. This approach enables the development of a more robust and comprehensive model capable of understanding and processing higher-level semantic information effectively.

4.2.1 Results of Semantic Segmentation

The semantic segmentation capsule model generates outputs representing various classes, including buildings, sky, road, and trees, as depicted in Figure 4.2. However, the proposed model may not perform optimally on vehicles, especially in scenes where multiple cars overlap.

Based on the experimental data presented in Table 4.2, our proposed model demonstrates exceptional performance in both the sky and building categories, achieving accuracy rates of 96.06% and 96.82%, respectively. The segmentation results of this model for the two segmentation objects, roads and trees, are also commendable, achieving accuracy rates of 79.67% and 86.3%, respectively.

Figure 4.3 displays the training loss of our model. As the number of training iterations increases, the loss steadily decreases until it converges at 10,000 training iterations. After 30,000 iterations, the training loss reaches 0.014.

To validate the performance of our model, we conduct similar experiments on U-Net and SegNet, two segmentation networks known for their excellence in semantic segmentation. In order to establish an equitable comparison between the capsule network and CNN, we apply linear upsampling instead of deconvolution in U-Net and SegNet. Additionally, we reduce the number of intermediate convolution outputs to 16

and the number of levels to 4. This adjustment allows for a more equitable evaluation of the performance of both models. We ensured that U-Net and SegNet had the same number of parameters as our suggested model. For the loss function, we utilized pixel-wise softmax cross entropy. Pixel-wise softmax cross entropy computes the loss for each pixel individually, which means it can provide fine-grained feedback to the network. This allows the model to learn to classify each pixel accurately, leading to precise segmentation results. The experimental results are presented in Table 4.2.

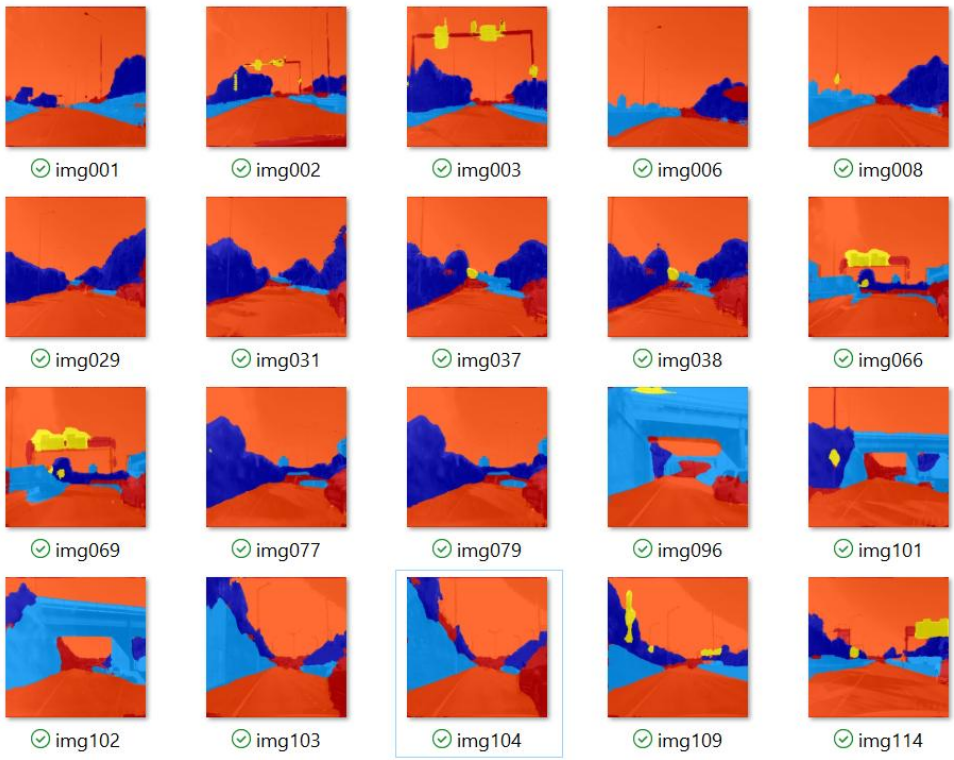


Figure 4.2 The results of capsule network segmentation

The experimental results reveal that U-Net achieves a significantly higher IoU of 60.58% compared to SegNet. Furthermore, U-Net outperforms SegNet by at least 11% in each class. When comparing U-Net to the mean IoU of our model on this dataset, our model performs only 0.43 percent better than U-Net. However, our model achieved comparable accuracy to SegNet while utilizing fewer parameters, resulting in faster

training. Nevertheless, there is room for improvement in the segmentation performance of vehicles and traffic signs. This might be attributed to the lower pixel values of vehicles in the image and the limited number of training samples for TrafficSigns. Further research and data augmentation techniques could help enhance the model's performance in these specific areas.

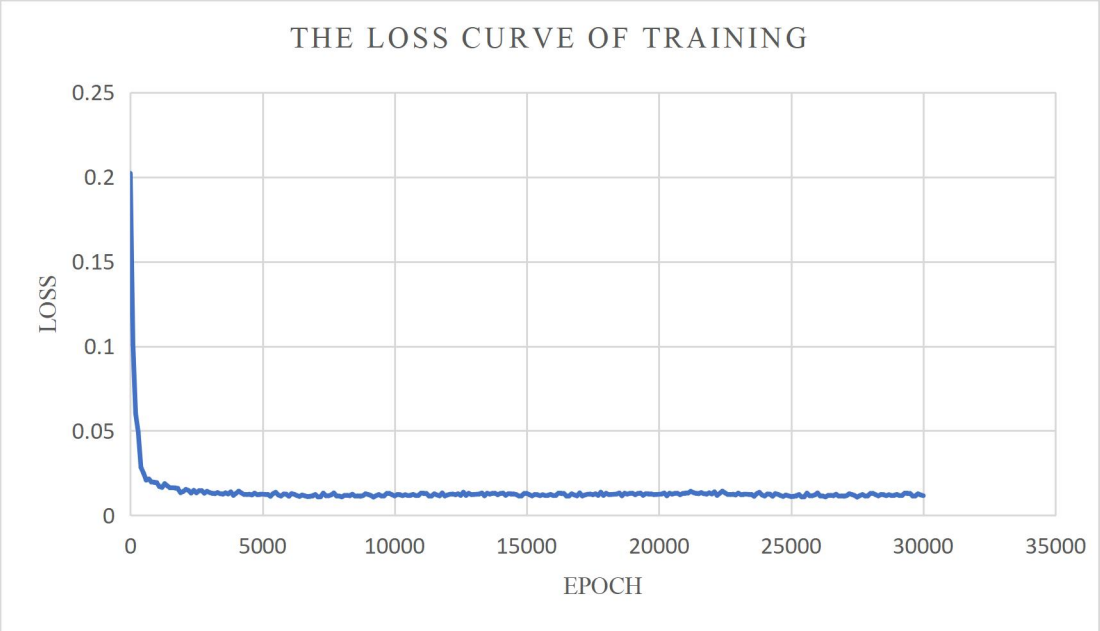


Figure 4.3 The loss curve of training process

To conduct a comprehensive evaluation of our model, we analyzed its performance using both a single vector feature representation and a conventional dynamic routing method. To perform the same experiment, we replace the appearance matrix with a vector using either dynamic routing (CapVec-DR) or our proposed VS routing (CapVec-VS) method.

Among the models adopting a single vector feature expression approach, CapVec-DR Dynamic routing mechanism exhibits the lowest accuracy, as demonstrated in Table 4.3. CapVec-VS, with a more refined routing method, outperforms CapVec-DR by 1.18%. Nonetheless, both of these models utilizing vector feature expression exhibit lower accuracy when compared to our proposed model. Our proposed model

outperforms the other two models in terms of performance.

Figure 4.4 illustrates a comparison of the test loss between CapVec-DR, CapVec-VS, and our model. CapVec-VS exhibited the highest loss value in the test after the first 2,000 iterations, while our model achieved the lowest loss value. While the drop curves of CapVec-VS and our model show similarities, it is noteworthy that the CapVec curve consistently remains lower at each point compared to the other two models. This indicates that the VS routeing mechanism might lead to faster convergence of the model compared to the dynamic routeing mechanism. Although CapVec-DR and CapVec-VS show similar loss values after 30,000 iterations, our model achieves the lowest loss value among the three. In comparison to the average losses from the experiments, our model achieved the lowest value of 0.38. On the other hand, CapVec-DR and CapVec-VS had average losses of 0.65 and 0.57, respectively.

Table 4.2 Comparison of different models in the same dataset

Networks	Vehicle	Road	Sky	Building	TrafficSign	Tree	Mean IoU
SegNet	28.98%	64.5%	82.87%	84.81%	27.52%	74.57%	60.54%
U-Net	40.23%	77.28%	95.98%	96.67%	48.63%	86.28%	74.18%
Ours	39.26%	79.67%	96.06%	96.82%	49.56%	86.31%	74.61%

Table 4.3 Comparisons between two model variants

Networks	Vehicle	Road	Sky	Building	TrafficSign	Tree	Mean IoU
CapVec-DR	36.52%	77.64%	94.18%	94.29%	47.31%	84.17%	72.35%
CapVec-VS	37.9%	79.17%	95.25%	95.75%	48.89%	84.24%	73.53%
Ours	39.26%	79.67%	96.06%	96.82%	49.56%	86.31%	74.61%

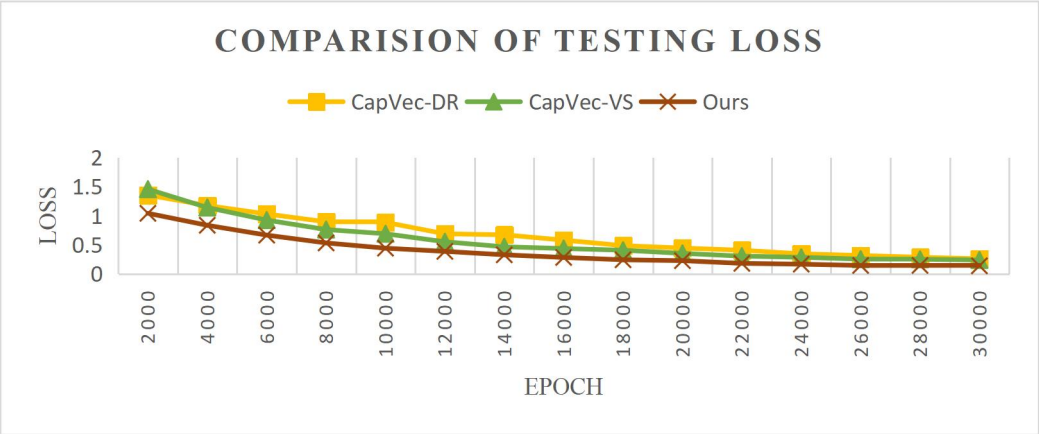


Figure 4.4 Comparison of testing loss

Furthermore, we observe the performance of our model by retraining it using the Cityscapes and CamVid datasets in Table 4.4. A comparative analysis is conducted against other state of the art traffic scene semantic segmentation models. Notably, our model achieves a significant improvement in mIoU on the Cityscapes dataset. Specifically, our model's mIoU surpasses that of BiseNet v2 (Yu et al., 2021) by 2.19% and exceeds DFANet's (Li et al., 2019) mIoU by 5.49%.

In a reciprocal evaluation, we also subject BiseNet v2 (Yu et al., 2021) and DFANet (Li et al., 2019) to training with our dataset. The results substantiate our model's superiority over BiseNet v2 (Yu et al., 2021), exhibiting a mIoU advancement of 2.63%.

In conclusion, the experiment revealed that the capsule network-based model outperforms classic semantic segmentation models (SegNet and U-Net) and state of the art models (BiseNet v2 and DFANet). Furthermore, the use of a combination of appearance and pose expressions in the model leads to improved scene segmentation ability compared to using a single vector expression and traditional dynamic routing methods.

Table 4.4 Comparing our model to other state of the art models in different datasets

Dataset	**Model**	**mIoU**
CityScapes	BiseNet v2 (Yu et al., 2021)	72.62%
	DFANet (Li et al., 2019)	70.32%
	Our model	75.81%
CamVid	BiseNet v2 (Yu et al., 2021)	68.79%
	DFANet (Li et al., 2019)	64.75%
	Our model	69.24%
Our dataset	BiseNet v2 (Yu et al., 2021)	71.98%
	DFANet (Li et al., 2019)	74.21%
	Our model	74.61%

4.2.2 Results of Depth Estimation

The experiments are conducted using the KITTI dataset, which consists of calibrated video registered to LiDAR measurements of a city, acquired from a moving vehicle. The KITTI dataset is employed for evaluating the performance of computer vision

technologies, such as stereo, optical flow, visual odometry, 3D object detection, and 3D tracking. The dataset comprises in-vehicle environment and motorway scenes, featuring up to 15 vehicles and 30 pedestrians per image, along with varying levels of occlusion and truncation. The dataset consists of 389 pairs of stereo images and corresponding optical flow maps, a visual ranging sequence spanning 39.2 kilometers, and over 200,000 sampled and synchronized 10Hz images with 3D labeled objects. The original dataset was classified into categories such as 'Road,' 'City,' 'Residential,' 'Campus,' and 'Person.' KITTI comprises a total of 151 sequences, each accompanied by raw data captured from the cameras for each frame. The resolution of the rectified RGB images varies significantly depending on the calibration settings, but it is approximately 1242×375 pixels.

As shown in Figure 4.5, we randomly extracted 5,000 data samples from the KITTI dataset for the purpose of training and testing. The dataset is then split into a training set and a test set using a 7:3 ratio. The pixel dimensions of all training images are resized to 320×1024.

In this experiment, the swin transformer is employed as the encoder, while a CNN serves as the decoder's network structure to accomplish depth estimation. The primary objective is to leverage the transformer-based encoder to address the limitations of CNN in global feature extraction. The generated depth maps from our network are illustrated in Figure 4.8.

In order to assess the network's robustness and accuracy effectively, we consider eight evaluation indicators throughout the training process. These indicators include training loss, a_1, a_2, a_3, Absolute Relative Error, RMSE, RMSE_log, and Squared Relative Error. In Figure 4.6, the training curve is shown for a learning rate of 0.0001, batch size of 1, and epoch of 30.

Figure 4.5 The original RGB images training data

Up to epoch 15, we observe a clear decreasing trend in the training loss curve in Figure 4.6, and all error curves in Figure 4.7 display a gradual reduction that X-axis represents epoch. Simultaneously, the curves of a_1, a_2, and a_3 show a smooth increase. The minor oscillations observed in the falling and rising curves are attributed to a slight imbalance in the randomly sampled data from the KITTI dataset. After epoch 15, the training curves of each indicator start to stabilize, indicating the successful convergence of the network.

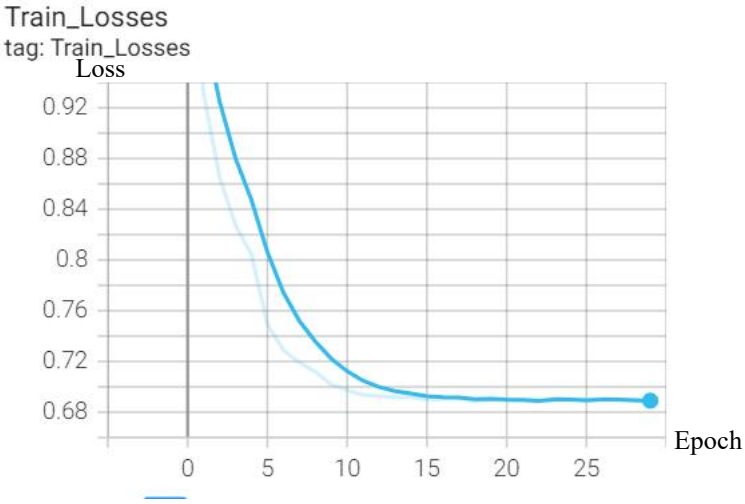


Figure 4.6 The training loss curves of depth estimation

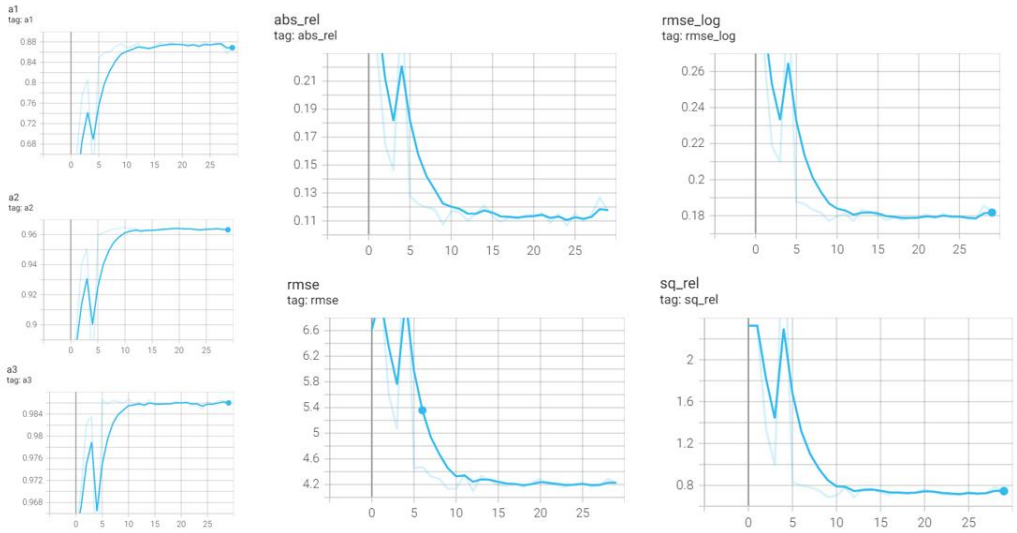


Figure 4.7 The evaluation curves of depth estimation

During the optimization process, the loss function undergoes reduction, reflecting the network's improvement. The user-selectable hyperparameters, such as the learning rate, batch size, and number of epochs, play a vital role in influencing the training outcome. These hyperparameters are essential components in determining the model's performance and convergence during training.

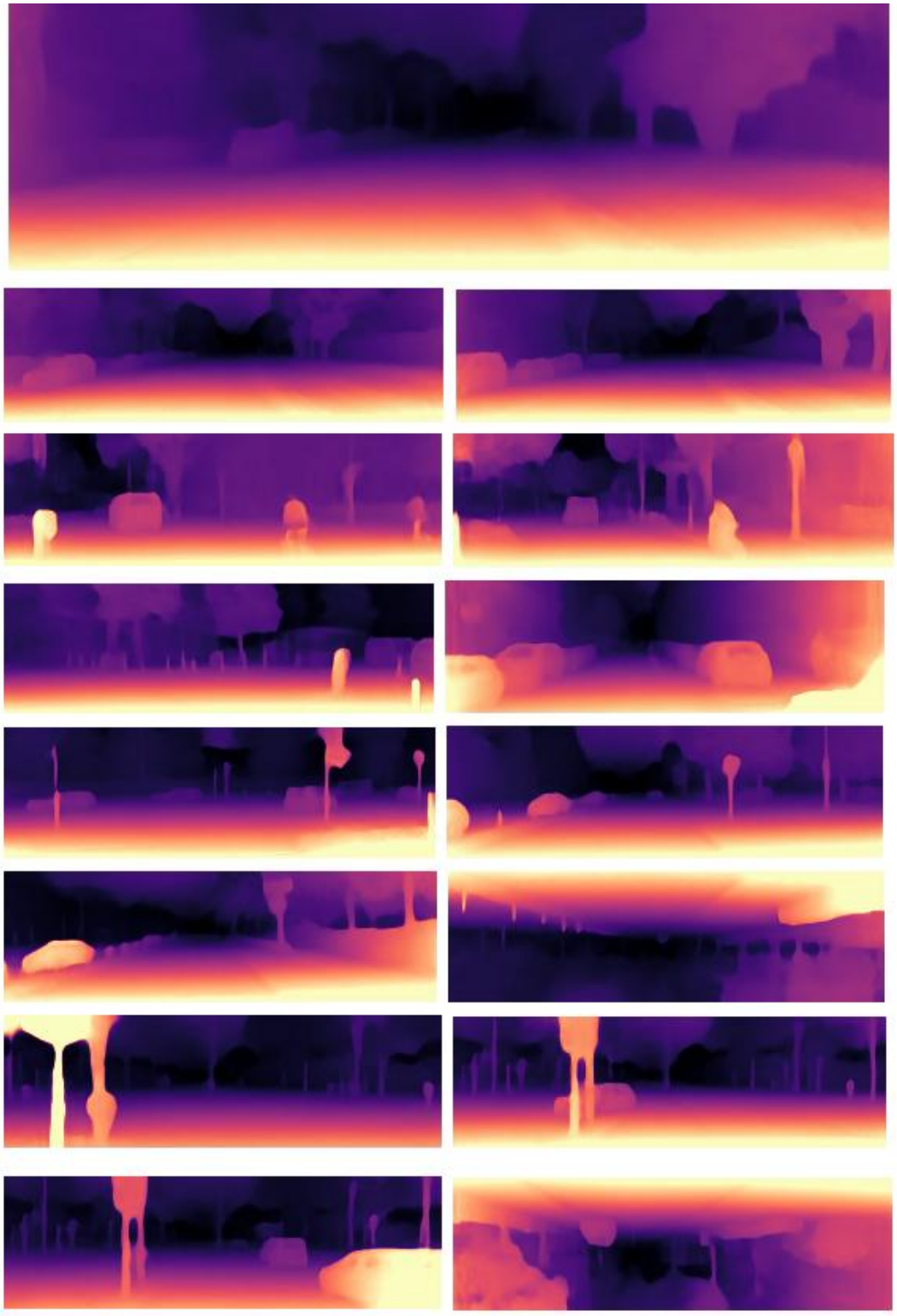


Figure 4.8 The depth estimation of traffic scenes with color maps at pixel level

Therefore, we conducted an evaluation of the model under different epoch settings. Specifically, we trained the model with epoch values of 20, 25, 30, and 35, and observed the changes in the evaluation metrics. The results are presented in Table 4.5. When the epoch was set to 20, the model achieved the highest and most favourable results for a_1 and a_2 metrics. Setting the epoch to 30 resulted in the lowest training loss of 0.668, and it also led to relatively lower absolute relative error and RMSE log compared to other settings. However, the benefits of the 30-epoch setting were more prominent for RMSE and Squared relative error metrics. Notably, the RMSE value reached 4.083 in our early experimental findings.

Table 4.5 The validation results in different settings of epoch

	Training loss	a_1	a_2	a_3	abs_rel	RMSE	RMSE_log	sq_rel
EPOCH20	0.689	**0.8748**	**0.9642**	0.986	**0.114**	4.291	**0.1792**	0.7615
EPOCH25	0.6882	0.873	0.9639	**0.9863**	0.1155	4.231	0.1794	0.7443
EPOCH30	**0.668**	0.869	0.9627	0.9857	0.1167	**4.083**	0.1824	**0.7397**
EPOCH35	0.671	0.862	0.9615	0.9854	0.1173	4.211	0.1931	0.7429

The results demonstrate that the model accurately determines the distance relationships in the scene. Additionally, the output images show clear edges of vehicles, buildings, traffic lights, and other objects. The color maps also provide fairly decent depth information for the primary objects. Overall, the model's performance is impressive, showcasing its ability to understand the scene and provide accurate depth estimation.

To ensure the stability and reliability of our network, we utilize Root Mean Squared Error (RMSE), Absolute Relative Error (AbsRel), and Squared Relative Error (SqRel) as evaluation metrics to assess the performance of DensDepth (Alhashim et al., 2018) and our proposed models, as presented in Table 4.6, while training on the same dataset. DensDepth represents a cutting-edge neural network that employs CNN for depth estimation. Similar to our network, it follows an encoder-decoder architecture. The key distinction lies in our network's utilization of a Transformer-based encoder, while

DensDepth utilizes a CNN-based encoder called DenseNet-169. In DensDepth, the encoder takes the input RGB image and transforms it into a feature vector using the pre-trained DenseNet-169 network, which was originally trained on ImageNet. Subsequently, this feature vector undergoes a sequence of up-sampling layers to construct the final depth map, which is produced at half the input resolution. These up-sampling layers, along with their corresponding skip-connections, collectively constitute the decoder of the DensDepth model. Upon comparison, our model outperforms DensDepth in every evaluation metric, particularly in RMSE and Rq_rel, where it achieves significantly better results, with improvements of 0.32 and 0.129, respectively.

Table 4.6 Comparison of the performance of our method and DensNet

	DensDepth (Alhashim et al., 2018)	**Ours**
Loss	0.704	0.688
a1	0.853	0.869
a2	0.959	0.962
a3	0.9849	0.985
Abs_rel	0.125	0.117
RMSE	4.548	4.228
RMSE_log	0.188	0.182
Sq_rel	0.874	0.745

In conclusion, in this experiment, we found that our model utilizing the swin transformer as the encoder demonstrates excellent performance in accomplishing the depth estimation task. At the same time, we also verified that the our transformer-based model is more suitable for the depth estimation task than the CNN-based encoder.

4.2.3 Results of Vehicle Tracking

We collected a substantial amount of visual data from traffic scenes using a driving recorder and created a dataset focusing on vehicles as the target objects. From this dataset, we selected 3,000 high-quality images for processing and labeling, dividing them into a training set and a test set with a 3:1 ratio. Additionally, we applied data augmentation techniques such as rotation, flipping, and translation to enhance the dataset's diversity.

Through the experimental results of vehicle detection using YOLOv5-CBAM-Transformer observe that the model accurately detects both larger vehicles nearby and smaller ones in the distance. Moreover, there is no noticeable shift in the position of the bounding box, even for smaller vehicles in the distance. The effectiveness of this vehicle detection task provides crucial support for subsequent vehicle tracking, as illustrated in Figure 4.9. The tracking boxes' scales in the vehicle tracking task are highly adaptive, adjusting according to the varying distances between vehicles. Our model assigns the same ID to the same target, allowing for effective detection and tracking of vehicles even in the presence of occlusions.

The proposed model employs three loss functions to evaluate its performance in different aspects: bounding box attribute, object confidence, and class probability score, as depicted in Figure 4.10. The bounding box loss assesses how accurately the model predicts the position of the bounding box through regression. The object confidence loss evaluates the model's confidence in detecting objects and how accurate its predictions are for the box containing an object. The classification loss measures the model's ability to distinguish between objects and backgrounds. In addition, we use precision and recall as metrics to assess the quality of the results. Precision indicates the proportion of true positive predictions among all positive predictions made by the model, while recall measures the proportion of true positive predictions among all actual positive instances in the dataset. These evaluation metrics help us understand the model's performance

comprehensively.

During the training process, the curve trends of bounding box loss, objective loss, and classification loss, as depicted in Figure 4.10, follow a similar pattern. In the initial 50 epochs, all three loss curves experience a rapid decline, indicating the model's rapid learning and adjustment. Subsequently, from 50 epochs to 150 epochs, the three loss curves gradually stabilize, suggesting that the model starts to converge and fit the data. Simultaneously, the precision and recall curves show a tendency to plateau at around the 75th epoch and 50th epoch, respectively. To comprehensively evaluate the model's performance, we trained the model using various Intersection over Union (IOU) thresholds, ranging from 0.50 to 0.95. The final average precision of the model for the detection of 2214 vehicles is 0.995, as shown in Figure 4.11, indicating the high accuracy and robustness of the network in detecting vehicles.

During the evaluation of vehicle tracking performance, we vary the location error thresholds during the training process to compute precision values for each threshold and assess the model's performance based on the area under the curve. Additionally, we calculate the ratio of frames successfully tracked in the sequence to the total number of frames at different overlap rate thresholds. In both Figure 4.12 and Figure 4.13, the plotted points on the curves are positioned above the diagonal lines, indicating that the model's performance is satisfactory. These results signify that the model is capable of accurately tracking vehicles, with the achieved performance surpassing the expected baseline.

Figure 4.9 Experimental results of object tracking

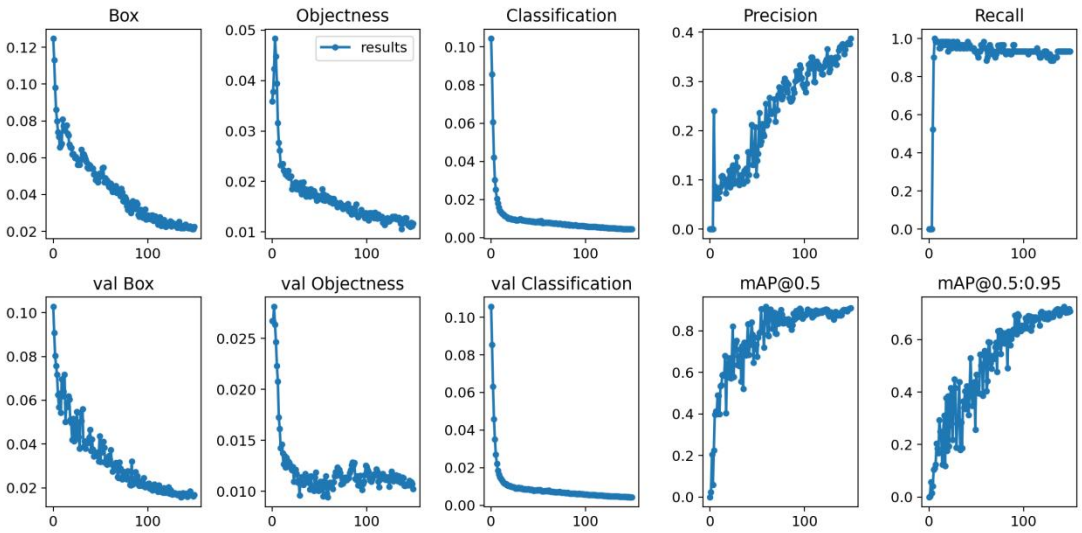


Figure 4.10 Evaluations of vehicle detection with multiple methods (X-axis represents epoch)

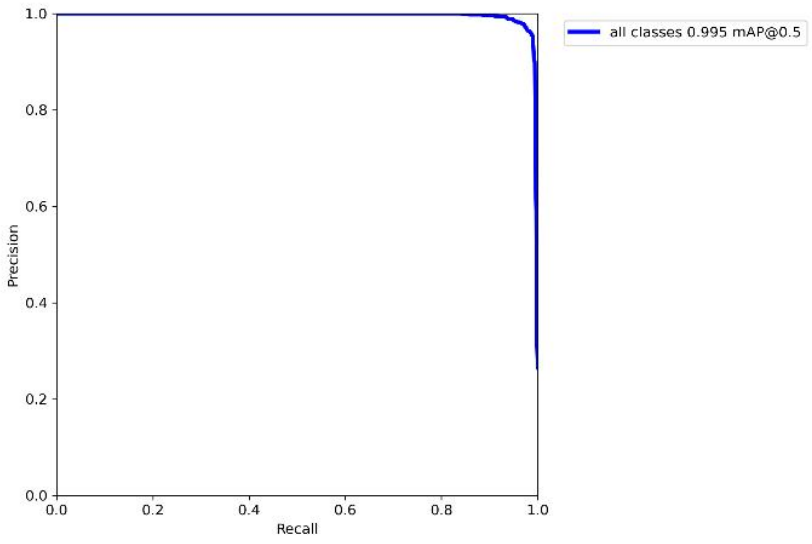


Figure 4.11 The mean average precision curve for a detection task

By comparing the state-of-the-art models SSD, YOLOv4, YOLOv5 and our modified networks YOLOv5-CA, YOLOv5-SE, YOLOv5-CA-transformer, YOLOv5-SE-Transformer, YOLOv5-CBAM and YOLOv5-CBAM-Transformer are shown in Table 4.7 that in the case of a batch size of 16, our proposed

YOLOv5-CBAM-Transformer is excellent in FPS and mAP. We can also know from the comparison between the combination of YOLOv5 plus various attention modules and the results of adding Transformer to them that Transformer can effectively improve the detection average precision and mean average precision of our three modified YOLOv5 (YOLOv5-CA, YOLOv5-SE and YOLOv5-CBAM). In order to ensure the accuracy and robustness of this model, we also combined our modified SiamRPN with YOLOv5-CBAM, DaSiamese with YOLOv5-CBAM and YOLOv5-CBAM-Transformer, compared the performance of our model (YOLOv5-CBAM-Transformer + modified SiamRPN) in Table 4.8. DaSiamRPN+YOLOv5-CBAM MOTA performed the best among the other three models, but it was still 1.10% lower than the proposed model. For MOTP, the performance of YOLOv5-CBAM + modified SiamRPN is the best among the other three models, but it is 0.30% lower than the proposed model. Moreover, from the results, we can see that under the premise of combination of our proposed YOLOv5-CBAM-Transformer which can receive the best detection result than other models, the effect of using modified SiamRPN is better than that of DaSiamRPN in MOTA (5.2% higher) while they are not much different on MOTP.

The proposed model takes use of a large number of training data from our traffic scene in Auckland and provides convenience for future research on traffic signs and road conditions. Secondly, this model combines the two our modifications of advanced deep learning models of YOLOv5-CBAM-Transformer and the SiamRPN for the first time to realize the understanding of the traffic scenes. The experimental results show that the model is satisfactory in detecting and tracking different sizes of vehicles in the distance in complex vehicle-related scene, and the bounding box has strong adaptability to vehicles of different sizes in dynamic traffic scenes. This model added CBAM and Transformer encoder to YOLOv5 for improving the detection performance and applies Hungarian algorithm to achieve multi-object tracking, so that the model is able to efficiently detect and track multiple vehicles in a complex traffic environment.

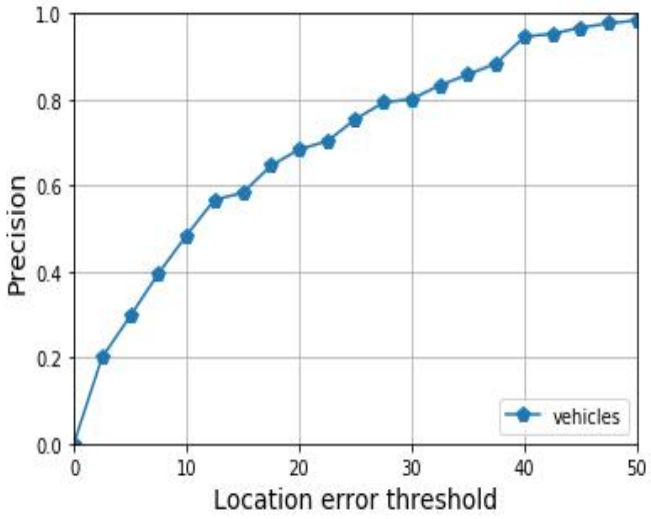


Figure 4.12 The curve related to the thresholds of location errors and precisions

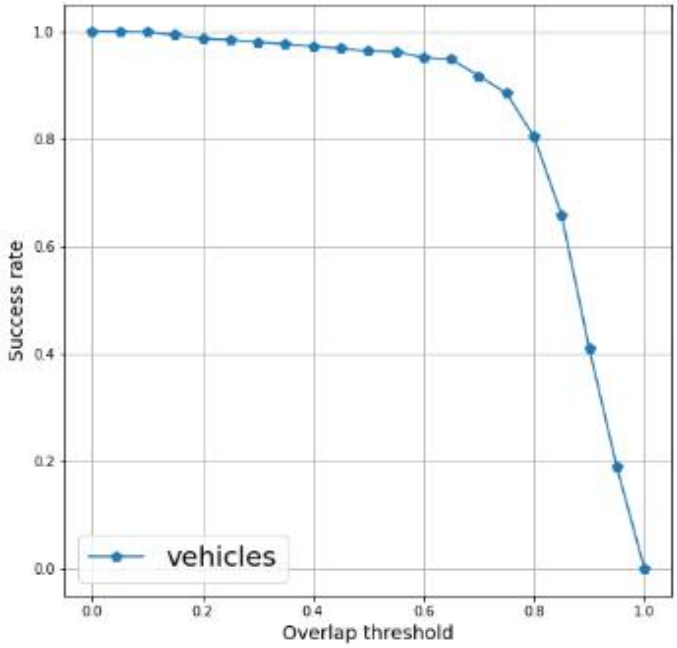


Figure 4.13 The curve reflected the relationship between the overlapping thresholds and success rates

Table 4.7 Comparisons of different detection methods

Methods	**mAPs_0.5**	**mAPs_0.5:0.95**	**FPS**	**Batch sizes**
SSD	98.10	85.60	39	16

YOLOv4	97.60	77.90	35	16
YOLOv5	98.40	87.30	37	16
YOLOv5-CA	97.80	82.10	36	16
YOLOv5-CA-Transformer	97.80	82.30	35	16
YOLOv5-SE	95.90	72.20	37	16
YOLOv5-SE-Transformer	96.50	77.80	37	16
YOLOv5-CBAM	98.90	88.10	36	16
YOLOv5-CBAM-Transformer	**99.50**	**88.70**	**37**	**16**

Table 4.8 Comparisons of various tracking methods

Models	**MOTA (%)**	**MOTP (%)**	**MT**	**ML**	**FP**	**FN**	**FM**
YOLOv5-CBAM-Transformer + DaSiamRPN	33.70	74.10	21.2	39.7	307	3998	84
YOLOv5-CBAM +DaSiamRPN	37.80	75.90	19.4	36.2	311	14427	171
The proposed YOLOv5-CBAM+Modified-SiamRPN	37.30	76.40	12.6	45.8	591	12769	198
The proposed YOLOv5-CBAM-Transformer+Modified-SiamRPN	38.90	76.70	15.5	32.7	276	15962	247

In summary, in this experiment, although the detection speed of our proposed YOLOv5-CBAM-Transformer is not the fastest in FPS, the detection precisions under the same conditions is higher than that of other compared models. And we found that when YOLOv5-CBAM is with the Swin Transformer, it can effectively improve the accuracy by 1.1%. Moreover, compared to other Siamese Networks (DaSiamRPN), our

modification of SiamRPN performs more prominently in vehicle tracking tasks.

4.2.4 Results of Distance Estimation

This thesis presents a novel vehicle detection and distance estimation model for low-cost monocular cameras, enhanced with an attention module and Transformer, utilizing deep learning techniques. The experimental setup involved using Python 2.7, an RTX5000 GPU, and 32GB RAM. The example data used in the experiments was sourced from the KITTI dataset. The KITTI dataset comprises both intrinsic and extrinsic characteristics of the in-car camera, along with the coordinates, width, and height of the detection boxes. For the development of our deep learning model, we randomly selected 4,000 samples and divided them into a 7:3 ratio for training and testing. The results presented in this section correspond to state-of-the-art approaches. Figure 4.14 illustrates the satisfactory vehicle recognition and distance estimation performance achieved by our modified YOLOv7 (YOLOv7-CBAM-Transformer) with the extended prediction vector (Khan et al., 2017).

To train our YOLOv7-CBAM-Transformer, we set the following parameters: epochs=5000, batch size=1.0, and learning rate=0.01. Figure 4.15 illustrates the network training process, showing that validation loss decreases steadily between 0 and 1000 iterations. After reaching 1000 epochs, the loss curves stabilize at 0.082.

Table 4.9 provides a quantitative comparison of the KITTI-constructed dataset for each of the evaluation measures listed. Several YOLO models and the transformer model are evaluated in this comparison. The results indicate that YOLOv7 outperforms all previous YOLO models, including the transformer. Moreover, our YOLOv7-CBAM-Transformer, which incorporates the CBAM, outperforms the original YOLOv7. The combination of the CBAM with YOLOv7 demonstrates significantly improved results. Notably, the addition of the Swin Transformer leads to a reduction of 0.382 in RMSE compared to the previous model. Overall, our

YOLOv7-CBAM-Transformer achieves a total reduction of 0.456 in RMSE compared to the original YOLOv7 model.

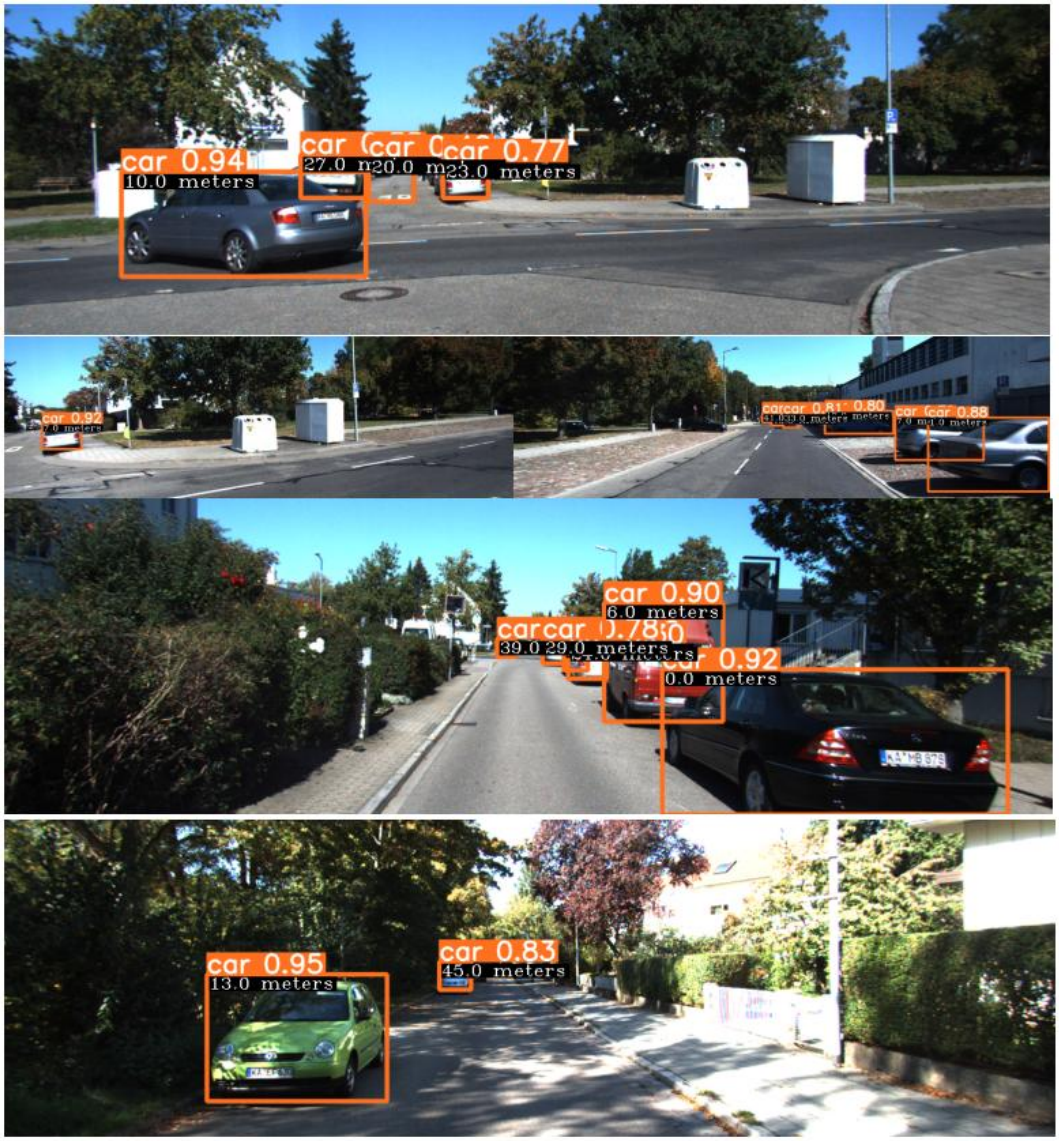


Figure 4.14 The example of vehicle detection and distance estimation using YOLOv7-CBAM-Transformer

Additionally, the distances were divided into three categories: 0-10m, 10-20m, and >20m. In Table 4.10, we obtain the average RMSE for each group. Our YOLOv7-CBAM-Transformer outperforms the original YOLOv7 in 0-10m and 10-20m

distance categories. In summary, the data presented in Table 4.9 and Table 4.10 suggest that the YOLOv7-CBAM-Transformer model is more effective in handling object detection and distance estimation tasks. Moreover, the model with Transformer is reduced by at least 0.2 in the RMSE of each distance category compared with YOLOv7-CBAM; compared with the original YOLOv7, it is reduced by at least 0.303 in distance caegory of 0-10m and 10-20m.

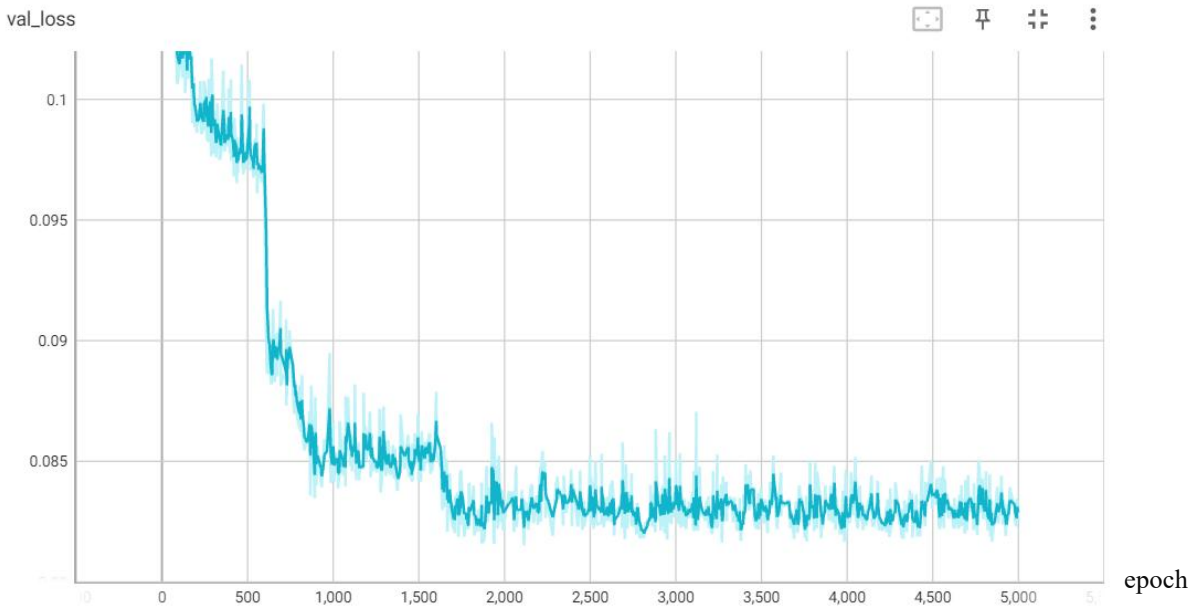


Figure 4.15 The diagram of training process of YOLOv7-CBAM. The blue curve indicates the validation loss.

Table 4.9 Comparative Analysis of Multiple Deep Neural Networks

Models	RMSE
YOLOv5	4.121
YOLOv6	4.483
YOLOv7	4.157
YOLOv7-CBAM	4.083
YOLOv7-CBAM-Transformer	3.701
Swin Transformer	4.776

Table 4.10 Average RMSE of different neural networks in different distance categories

Models	0-10m	10-20m	>20m
YOLOv7	4.502	4.357	3.612
YOLOv7-CBAM	4.499	3.667	4.083
YOLOv7-CBAM-Transformer	4.199	3.021	3.883

Table 4.11 Comparing our model to other state of the art model of distance estimation

Model	RMSE
GC-ASPP-YOLOv3-D (Lian et al., 2022)	3.985
Ours (YOLOv7-CBAM-Transformer)	**3.701**

In the context of identical dataset, we conducted a comparative analysis between our model and another state of the art distance estimation model GC-ASPP-YOLOv3-D in Table 4.11 (Lian et al., 2022). The results highlight the distinct advantages of our

model, particularly in terms of RMSE, where it outperforms GC-ASPP-YOLOv3-D.

In summary, in this experiment, we found that in addition to YOLOv7 being more suitable for vehicle distance estimation in KITTI dataset than YOLOv5, YOLOv6 and original Swin Transformer, adding CBAM can successfully further reduce the estimation error of the model. Moreover, we found that YOLOv7 with CBAM is more suitable for distance estimation within 20 meters, while the original YOLOv7 is more prominent when estimating the distance of vehicles beyond 20m. By further improvement, our proposed YOLOv7-CBAM-Transformer produced the better results even compare with YOLOv7-CBAM.

Chapter 5 Analysis and Discussions

In this chapter, we will explore how design decisions have influenced the performance of our models. We will also delve into the reasoning behind these decisions and evaluate whether they have yielded the anticipated outcomes. Additionally, we analyze and compare the values of various parameters utilized in the model to gain insights into their impact on performance.

We have directly introduced the results of scene understanding. In this chapter, we analyse and discuss the output of each of the four tasks: scene segmentation, vehicle tracking, depth estimation and distance estimation by comparing the data from a large number of ablation experiments. The analysis results favorably demonstrate that our final model structure allows for maximum optimization of the model's performance.

5.1 Analysis and Discussion of Vehicle Tracking

In Chapters 3 and Chapter 4, we describe the network structure and experimental results of vehicle tracking in detail. In this chapter, we perform ablation experiments using different network structures and training parameters to verify that our final model is optimal and robust.

Firstly, we discuss the size of the network, the memory footprint and the number of parameters by comparing YOLOv5 with several variants of YOLO. It is well known that lighter, faster and easier-to-deploy models are preferred, provided they perform well. This is because the lighter and faster models are less complex and can be trained with much less computation and faster results.

The existing models are generally made lighter by reducing the number of flops, memory and parameters and faster by adding a shuffle channel and channel clipping to the YOLOv5 head. By removing the Focus layer and quadruple slice operations, the model quantization accuracy is reduced to within acceptable limits, to the point where the model is easier to deploy.

YOLO-fastest, Shufflev2-YOLOv5, YOLOv4-Tiny, and YOLOv3-Tiny are four models that have been structurally altered to make the YOLO family of models lighter and faster.

Shufflev2-YOLOv5 network structure mainly uses a shuffle block with a shuffle channel as the backbone and a YOLOv5 head as the header. The design philosophy

behind Shufflev2 involves streamlining the architecture to enhance efficiency and performance. This includes removing the Focus layer to eliminate multiple slice operations and avoiding redundant usage of the C3 Layer and the high channel C3 Layer. The C3 Layer is an improved version of YOLOv5 CSPBottleneck, known for its simplicity, speed, and superior results with minimal losses. These optimizations contribute to the overall effectiveness and computational efficiency of Shufflev2. Nonetheless, the C3 Layer utilizes multiple separate convolutions, and experiments have demonstrated that frequent use of C3 Layer and higher channel counts in C3 Layer occupy more cache space, leading to slower operations. Furthermore, the yolov5 head's channel clipping is also implemented. Additionally, the ShuffleNetv2 backbone removes the 1024 conv and 5×5 pooling. These optimizations are implemented to improve the overall efficiency and speed of the model.

The YOLO-Fastest network utilizes ShuffleNetV2 as its backbone, the light-FPN network as its Neck part, and consists of three components in the head: classification, regression, and detection. Notably, YOLO-Fastest extensively employs depthwise separable convolution to achieve remarkable speed, reduce the number of parameters, and facilitate efficient deployment on mobile devices.

The YOLOv4-Tiny structure is a lightweight version of YOLOv4, comprising only 6 million parameters, which is one-tenth of the original model. This reduction in parameters significantly improves the detection speed of the model. The complete network structure consists of 38 layers, including three residual units, a LeakyReLU activation function, two feature layers dedicated to target classification and regression, and a Feature Pyramid Network (FPN) to efficiently merge the essential feature layers. Moreover, the network incorporates the CSPnet structure, employing channel splitting in the feature extraction network. This process divides the output feature layer channels after a 3x3 convolution into two parts, and then selects the second part for further processing.

The YOLOv3-Tiny structure is a simplified version of YOLOv3, aimed at reducing network complexity. In contrast to YOLOv3, the tiny version significantly compresses the network and omits the use of a res layer, retaining only two different scales of YOLO output layers. For a comprehensive comparison of the five models based on input size, FLOPs, and parameters, refer to Table 5.1. We can see that YOLO-fastest and Shufflev2-YOLOv5 require the smallest input size. At the same time, YOLO-Fastest has lower flops and parameters than shufflev2-YOLOv5. This shows that YOLO-Fastest is a much simpler network. However, with an input size of 416×416, Shufflev2-YOLOv5 requires fewer flops and parameters than YOLOv4-Tiny and YOLOv3-Tiny. In contrast, the network structure and computation of YOLOv5 are slightly higher than that of the other YOLO variants in Table 5.1.

Table 5.1 A comparison of the five models in terms of input size, FLOPs (floating-point operations), and parameters.

Network	Input size	Flops	Parameters
YOLO-fastest	320×320	0.25G	0.35M
Shufflev2-YOLOv5	320×320	1.43G	1.62M
Shufflev2-YOLOv5	416×416	2.42G	1.62M
YOLOv4-tiny	416×416	5.62G	8.86M
YOLOv3-tiny	416×416	6.96G	6.06M
YOLOv5	640×640	17.0G	7.30M

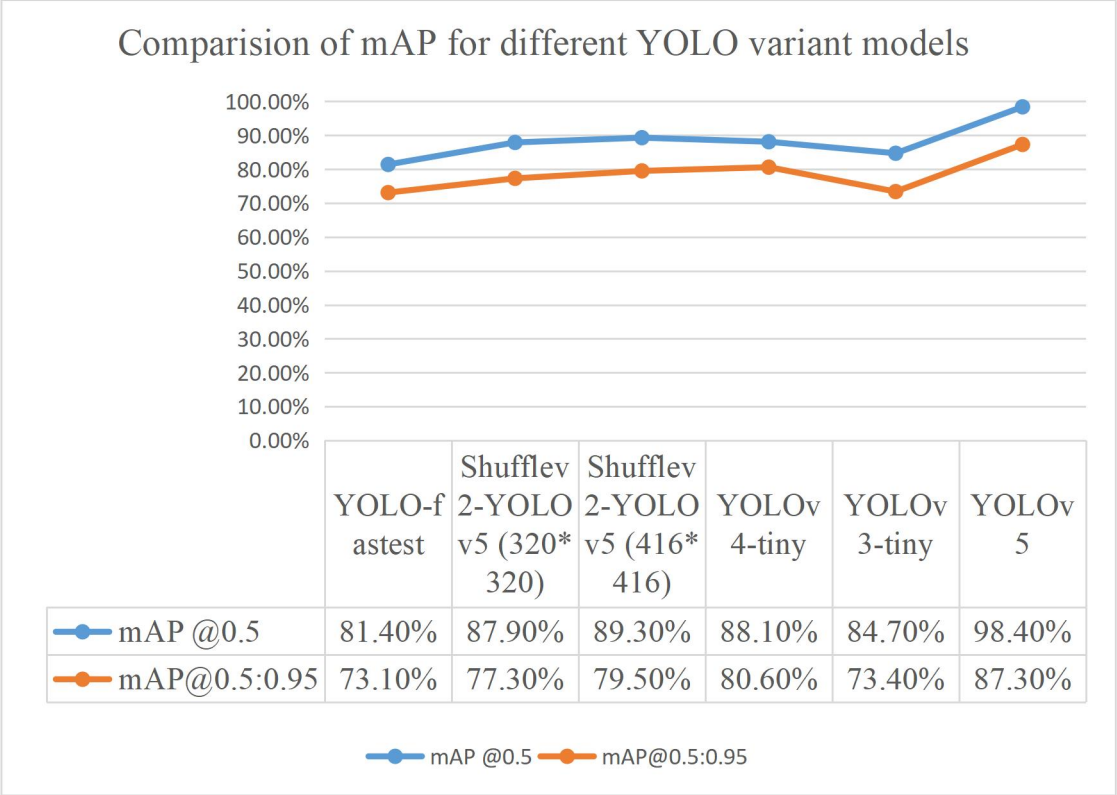
In order to evaluate the six models more fairly, the two evaluation indicators that mAP@0.5 and mAP@0.5:0.95 were compared in Figure 5.1. The mAP@0.5:0.95 denotes the average mAP at different IoU thresholds from 0.5 to 0.95 in steps of 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95). Similarly, mAP@0.5 denotes the average mAP at different IoU thresholds from 0.5 to 1. The comparison of the results

reveals a significant difference in mAP (mean Average Precision) when using different IOU (Intersection over Union) thresholds. Specifically, the mAP when the IOU threshold is greater than 0.5 is notably higher compared to when the IOU threshold falls within the range of 0.5 to 1. This disparity occurs because the mAP calculation at IOU thresholds from 0.5 to 0.95 excludes the IOU threshold of 1, leading to a rounded-off effect in the calculation process.

We also found that YOLO-fastest with the fewest flops and parameters only obtained 81.40% and 73.10% on mAP@0.5 and mAP@0.5:0.95. In contrast, YOLOv5, which required the most Flops and parameters, gains 98.40% and 87.30% on mAP@0.5 and mAP@0.5:0.95. The rest of the models (Shufflev2-YOLOv5, YOLOv4-Tiny and YOLOv3-Tiny) performed below YOLOv5. Compared to Shufflev2-YOLOv5 with different input sizes, Shufflev2-YOLOv5 with a larger input size outperforms the smaller input size by more than 2% on both mAP@0.5 and mAP@0.5:0.95.

Therefore, though the streamlined network structure and parameters reduce the computational effort, the model performance suffers. To ensure the best performance of our model, YOLOv5 is the best choice.

As described in Chapter 3, to further improve network performance, we have added Swin Transformer to our improved attention module based network YOLOv5-CBAM. As setting different training parameters directly affects the accuracy and stability of the network, we have analyzed YOLOv5-CBAM and YOLOv5-CBAM-Transformer with respect to the two parameters, batch size and epoch, in order to find the optimal parameters.



	YOLO-fastest	Shufflev2-YOLOv5 (320*320)	Shufflev2-YOLOv5 (416*416)	YOLOv4-tiny	YOLOv3-tiny	YOLOv5
mAP @0.5	81.40%	87.90%	89.30%	88.10%	84.70%	98.40%
mAP@0.5:0.95	73.10%	77.30%	79.50%	80.60%	73.40%	87.30%

Figure 5.1 Comparison of the mAP of different YOLO variant models with different IoU thresholds

We trained the networks YOLOv5-CBAM and YOLOv5-CBAM-Transformer by setting the batch size to 4, 8, 16, 32 and epoch to 100, 110, 120, 130, 140, 150, 160, 170 and 180 while keeping other variables constant. We obtained the most satisfactory mAP@0.5, which is 99.5, at a batch size of 16 and an epoch of 150 that in YOLOv5-CBAM-Transformer. We did the same comparison experiment on YOLOv5-CBAM, and the results showed that mAP@0.5 reached its maximum at a batch size of 4 and an epoch of 130. By comparing the optimal mAPs in Table 5.2 and Table 5.3, we can demonstrate that Swin Transformer has a strong ability of helping our improved model YOLOv5-CBAM obtain meaningful enhancement.

Table 5.2 Comparison of mAPs of YOLOv5-CBAM-Transformer in different batch sizes and epochs

BatchSize\Epoch	100 epochs	110 epochs	120 epochs	130 epochs	140 epochs	150 epochs	160 epochs	170 epochs	180 epochs
Batch size =32	62.0	62.9	65.1	70.1	69.8	73.1	69.7	68.8	66.5
Batch size =16	90.7	92.0	96.3	97.9	99.1	**99.5**	98.3	97.6	95.9
Batch size =8	93.4	93.2	93.5	92.2	94.0	94.3	95.9	98.4	96.2
Batch size =4	70.9	77.7	76.9	77.2	80.8	82.2	83.7	88.6	85.4

Table 5.3 Comparison of mAPs of YOLOv5-CBAM in different batch sizes and epochs

BatchSize\Epoch	100 epochs	110 epochs	120 epochs	130 epochs	140 epochs	150 epochs	160 epochs	170 epochs	180 epochs
Batch size =32	65.4	69.9	67.4	64.7	66.9	65.8	64.2	60.5	57.3
Batch size =16	63.6	66.6	67.3	69.0	69.8	67.4	67.9	66.3	63.2
Batch size =8	85.5	87.9	88.7	88.3	86.9	83.1	81.0	77.4	80.6
Batch size =4	90.9	92.6	95.3	**98.9**	96.5	95.9	96.8	96.0	95.1

In Chapter 3 and Chapter 4, our final vehicle tracking model was implemented using YOLOv5-CBAM-Transformer + regression modified SiamRPN. Not only did we

show that YOLOv5-CBAM-Transformer with modified SiamRPN performs better than the model without Transformer by comparing a large amount of experimental data, but we also observed some frames in the video.

The result shows in Figure 5.2 that both models perform well in tracking obscured vehicles and smaller vehicles in the distance. However, YOLOv5-CBAM+modified SiamRPN experienced missed detections on medium-sized vehicles, while YOLOv5-CBAM-Transformer+modified SiamRPN detected all vehicles on the scene very well.

In Figure 5.3, we present a performance comparison between the two models in Scene 2. By looking at them, we see that both models are accurate at tracking vehicles at long distances. However, YOLOv5-CBAM-Transformer+modified SiamRPN successfully detects vehicles that are turning. This proves that YOLOv5-CBAM-Transformer+modified SiamRPN can not only detect vehicles travelling in the same direction and the opposite direction but can also successfully make detections from other viewpoints without any drift in the detection frame.

In Figure 5.4, the YOLOv5-CBAM+modified SiamRPN incorrectly detects the distant bridge as a vehicle; despite the distance and small size, the YOLOv5-CBAM-Transformer+modified SiamRPN does not show this false detection.

In summary, we have verified through testing and comparison that the YOLOv5-CBAM-Transformer+modified SiamRPN can accurately track vehicles from long range, close range and side view. At the same time, the YOLOv5-CBAM-Transformer+modified SiamRPN can accurately track obscured vehicles and ensure no offset in the detection frame.

(a) YOLOv5-CBAM+modified SiamRPN

(b) YOLOv5-CBAM-Transformer+modified SiamRPN

Figure 5.2 Comparison of the performance of YOLOv5-CBAM-Transformer with modified SiamRPN and YOLOv5-CBAM with modified SiamRPN on the scene1.

(a) YOLOv5-CBAM+modified SiamRPN

(b) YOLOv5-CBAM-Transformer+modified SiamRPN

Figure 5.3 Comparison of the performance of YOLOv5-CBAM-Transformer with modified SiamRPN and YOLOv5-CBAM with modified SiamRPN on the scene2.

(a) YOLOv5-CBAM+modified SiamRPN

(b) YOLOv5-CBAM-Transformer+modified SiamRPN

Figure 5.4 Comparison of the performance of YOLOv5-CBAM-Transformer with modified SiamRPN and YOLOv5-CBAM with modified SiamRPN on the scene2.

5.2 Analysis and Discussion of Distance Estimation

In this chapter, we conduct a comprehensive analysis of the model's stability and robustness by exploring various variants of the base YOLOv7 model within the same experimental environment. We incorporate different modules and specific functional

layers and compare them with our proposed YOLOv7-CBAM-Transformer. Additionally, we carried out experiments to validate the reliability of the hyperparameters used in the models during the experiments. Moreover, we assess the performance of different models at various distance ranges. Through these experiments, we aim to showcase the superiority of our final model, demonstrating its outstanding performance and effectiveness.

Initially, we selected YOLOv7 as the main network structure for distance estimation. This decision was driven by the fact that YOLOv7 offers substantial enhancements in real-time object detection accuracy without significantly increasing the inference cost. The experimental results presented in Chapter 4 demonstrate that YOLOv7 surpasses other well-known object detectors in terms of detection accuracy. In summary, YOLOv7 offers a faster and more robust network architecture, efficient feature integration, accurate target detection performance, a robust loss function, and higher efficiency in label assignment and model training. It outperforms other models, making it a compelling choice for our research. YOLOv7-X is an extension of YOLOv7 that incorporates a module scale and a composite scaling method to adjust the depth and width scale of the entire model. These modifications aim to enhance the performance and adaptability of the network.

Model scaling serves the primary purpose of adjusting crucial properties of the model to create a model that suits various applications. This process involves optimizing the model's width (number of channels), depth (number of stages), and resolution (input image size) to achieve the desired performance and adaptability for specific tasks. In conventional methods using cascade-based architectures, the scaling factors are not analyzed independently but considered together. For instance, increasing the model depth may affect the ratio between input and output channels of the transition layer, potentially impacting the model's hardware utilization. This is the reason YOLOv7 was selected as the base model for this experiment. The composite scaling

method ensures that the model's properties remain consistent with the initial design, thereby preserving its optimal structure.

We also referenced YOLOv7-Tiny to participate in the comparison experiments. The 'small' suffix for computer vision models indicates that they are optimized for edge AI and deep learning workloads, making them more lightweight and efficient. In contrast to the other YOLOv7 variants, the edge-optimized YOLOv7-Tiny utilizes leaky ReLU as the activation function, while the other models employ SiLU as the activation function.

Table 5.4 Comparison of the size and performance of YOLOv7 and its variant models on the KITTI dataset

	Parameter	Flops	Precision	Recall	mAP@0.5	mAP@0.5:0.95
YOLOv7	36.9M	104.7G	94.8%	92.1%	96.9%	73%
YOLOv7-X	71.3M	189.9G	93.5%	91.3%	95.1%	70.5%
YOLOv7-Tiny	6.2M	13.8G	84.3%	80.5%	87.3%	60.8%

Table 5.4 shows that although YOLOv7-X utilizes a composite scaling method with depth and width scales, it does not yield significant improvements on the KITTI dataset. YOLOv7 slightly outperforms YOLOv7-X and significantly outperforms YOLOv7-Tiny in evaluation metrics such as precision, recall, and mAP. Moreover, YOLOv7 achieves better performance than YOLOv7-Tiny while using less than 50% of the parameters. Additionally, YOLOv7 outperforms YOLOv7-X while utilizing 50% fewer parameters. As a result, YOLOv7 can be considered the most suitable model among the three for ensuring detection accuracy and computational efficiency in this experiment.

In order to enhance the detection performance of the YOLOv7 model and maximize

its accuracy for subsequent distance estimation tasks, we attempted to incorporate four different modules (SkAttention, CBAM, and GAMAttention) into YOLOv7 and compared their effects (Li, Wang, Hu & Yang, 2019; Liu, Shao & Hoffmann, 2021; Woo et al., 2019).

In a typical convolutional neural network, the artificial neurons in each layer are designed to have the same size for their receptive fields. It is widely recognized in the neuroscience community that the receptive field size of visual cortical neurons is influenced by the stimulus, a factor that is often overlooked when constructing CNNs. A dynamic selection mechanism has been proposed for CNNs, enabling each neuron to adaptively adjust its receptive field size based on multiple scales of input information. The proposed building block is called a Selective Kernel (SK) unit, which incorporates multiple branches with different kernel sizes. These branches are fused using softmax attention, guided by the information present in each of these branches. The varying attention given to these branches leads to different effective receptive field sizes for neurons in the fusion layer. Multiple SK units are stacked in a deep network known as Selective Kernel Networks (Li, Wang, Hu & Yang, 2019).

The first improvement is the SK Attention mechanism, which draws inspiration from the SENet (Squeeze-and-Excitation) and dynamically fuses the outputs of individual convolutional kernels by calculating channel weights adaptively for each kernel. This allows the network to emphasize the target to be detected and enhances the detection results.

To overcome the limitations of previous methods that neglect preserving information in channels and spatial dimensions, we introduce a novel global scheduling mechanism called GAM Attention. This approach aims to enhance the performance of deep neural networks by minimizing information reduction and amplifying global interaction representation. GAM Attention incorporates a multilayer perceptron 3D permutation for channel attention and a convolutional spatial attention submodule (Liu,

Shao & Hoffmann, 2021). Additionally, the third enhancement involves integrating CBAM, which was discussed comprehensively in Chapter 3 and Chapter 4.

Table 5.5 Comparison of the size and performance of YOLOv7 and its variant models on the KITTI dataset

	mAP@0.5	mAP@0.5:0.95
YOLOv7_SKAttention	90.3%	62.9%
YOLOv7_CBAM	97.7%	75.4%
YOLOv7_GAMAttention	96.0%	66.2%

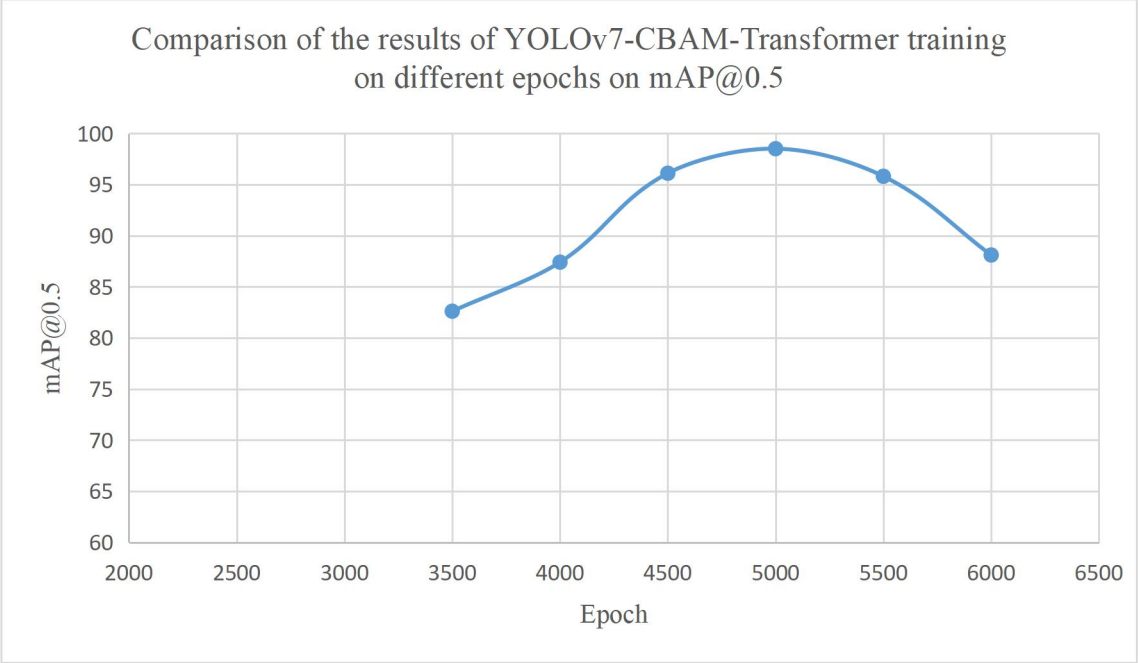


Figure 5.5 Comparison of the results of YOLOv7 training on different epochs on mAP@0.5

The results show in Table 5.5 that the combination of YOLOv7 and SKAttention was disappointing, as the mAP@0.5 of the model with SKAttention inserted dropped by 6.3% compared to the original YOLOv7. At the same time, the addition of

GAMAttention to YOLOv7 did not result in an increase in the expressiveness of the model. However, CBAM gave a 0.8% boost to YOLOv7 on mAP@0.5 and a 2.4% boost on mAP@0.5:0.95. Therefore, this experiment can show that the combination of YOLOv7 and CBAM can effectively improve the detection accuracy and contribute to the accuracy of the subsequent distance estimation.

Therefore, we finally chose the best-performing CBAM combined with YOLOv7 as the basic framework, and added Transformer to further improve the ability of the model.

Due to the large image size of the KITTI dataset, we decided to choose a more significant epoch number and a smaller batch size to ensure the training effect of YOLOv7_CBAM_Transformer. We experimented with different epoch numbers (*epoch* = 3500, 4000, 4500, 5000, 5500 and 6000) while fixing *batch_size* = 1 and learning_ rate = 0.01. As shown in Figure 5.5, the curve tends to increase from 3,500 to 5,000, and the accuracy decreases significantly from 5,000 to 6,000.

Table 5.6 Comparison of the performance of different model in different distance

	0-10m	10-20m	>20m	Average
YOLOv7	4.502	4.357	**3.612**	4.157
YOLOv7_SKAttention	4.790	4.533	4.574	4.632
YOLOv7_CBAM	4.499	3.667	4.083	4.083
YOLOv7_GAMAttention	4.621	3.695	4.527	4.281
YOLOv7-CBAM-Transformer	**4.199**	**3.021**	3.883	**3.872**

At the end of the experiment, our modified YOLOv7 models evaluate their RMSE

at different distance ranges. We found that in Table 5.6, YOLOv7 is suitable for estimating the distance from the vehicle to more than 20m among the four models that did not include Transformer in the comparison. In contrast, YOLOv7-CBAM is suitable for the distance of 0-10m vehicles in the range of 0-10m, and YOLOv7_GAMAttention is more accurate than other models in measuring the distance in the range of 10-20m. In terms of the average RMSE of the three ranges, YOLOv7_CBAM still has an advantage. Compared to the original YOLOv7 range, YOLOv7_CBAM improves the average RMSE by about 0.02. After adding Transformer, the model is higher than other networks without Transformer in 0-10m, 10-20m and Average.

YOLOv7-CBAM:

YOLOv7-CBAM-Transformer:

Figure 5.6 Distance estimation performance of different models on the same frame (group A)

For a more intuitive analysis of the performance of the different models in distance estimation, the performance of the YOLOv7-CBAM-Transformer and the YOLOv7-CBAM in four different traffic scenes are listed in Figure 5.6, Figure 5.7, Figure 5.8 and Figure 5.9, respectively. The orange box shows the confidence of the

vehicle detection, and the black box shows the estimated distance. Although our estimated distance values were kept to three decimal places, we kept only integer places in the resulting image to avoid the result boxes obscuring each other and cluttering the resulting image. When calculating the RMSE, we still use the results with three decimal places retained.

In group A, the YOLOv7-CBAM-Transformer accurately predicts the distances without any error of object 1 and object 6. According to the data shown in Table 5.7, when compared to YOLOv7-CBMA, the YOLOv7-CBAM-Transformer is more accurate when compared to the YOLOv7-CBAM for the other objects. At the same time, the YOLOv7-CBAM appears to miss the detection of distant vehicles during the detection process, which is one of the reasons for the significant difference in the mean absolute error between the two models in the scene of group A.

Table 5.7 Comparison of the estimated distance results of the two models in group A in same scene

	Estimated distance (YOLOv7-CBAM)	Estimated distance (YOLOv7-CBAM-Transformer)	True distance
Object1 (group A)	4 m	1 m	1 m
Object2 (group A)	11 m	7 m	9 m
Object3 (group A)	20 m	17 m	14 m
Object4 (group A)	44 m	33 m	36 m

Object5 (group A)	Nan	40 m	39 m
Object6 (group A)	49 m	41 m	41 m
Mean absolute error	5.4	1.5	/

Figure 5.7 Distance estimation performance of different models on the same frame (group B)

Table 5.8 Comparison of the estimated distance results of the two models in group B in same scene

	Estimated distance (YOLOv7-CBAM)	Estimated distance (YOLOv7-CBAM-Transformer)	True distance

Object1 (group B)	6 m	7 m	7 m
Object2 (group B)	7 m	Nan	8 m
Mean absolute error	1.0	4.0	/

YOLOv7-CBAM:

YOLOv7-CBAM-Transformer:

Figure 5.8 Distance estimation performance of different models on the same frame (group C)

A total of 2 objects are in the scene in group B. The YOLOv7-CBAM successfully detects two objects, while the YOLOv7-CBAM-Transformer fails to detect object2, which is heavily obscured by object1. Although the YOLOv7-CBAM-Transformer accurately estimated the distance of object1, its mean absolute error, which is shown in Table 5.8 was significantly higher due to the missed detection.

In the scene of group C shown in Figure 5.8, both models successfully detected all six vehicles and performed very similarly on each object. Overall, the YOLOv7-CBAM-Transformer performs slightly better than the YOLOv7-CBAM in this scene. When comparing the mean absolute error in Table 5.9, the YOLOv7-CBAM-Transformer is 0.33 lower than the YOLOv7-CBAM.

Table 5.9 Comparison of the estimated distance results of the two models in group C in same scene

	Estimated distance (YOLOv7-CBAM)	Estimated distance (YOLOv7-CBAM-Transformer)	True distance
Object1 (group C)	0 m	0 m	1 m
Object2 (group C)	6 m	6 m	8 m
Object3 (group C)	14 m	14 m	13 m
Object4 (group C)	20 m	21 m	21 m
Object5 (group C)	30 m	29 m	28 m
Object6 (group C)	29 m	39 m	35 m
Mean absolute error	1.83	1.5	/

YOLOv7-CBAM:

YOLOv7-CBAM-Transformer:

Figure 5.9 Distance estimation performance of different models on the same frame (group D)

Table 5.10 Comparison of the estimated distance results of the two models in group D in same scene

	Estimated distance (YOLOv7-CBAM)	Estimated distance (YOLOv7-CBAM-Transformer)	True distance
Object1 (group D)	11 m	10 m	9 m
Object2 (group D)	/	27 m	25m
Object3 (group D)	25 m	23 m	21 m
Object4	26 m	20 m	/

(group D)			
Mean absolute error	19	8.3	/

In Table 5.10, the highest mean absolute error was achieved for both models in group D, as both models incorrectly detected object4 (landmark) as a vehicle. In addition, the YOLOv7-CBAM has a missed detection on object2. Therefore, although it performs well on the other objects, the omissions and misdetections still significantly impact the global performance.

In summary, by analysing the performance of the different models in the vehicle scene, we found that the accuracy of vehicle detection has a crucial impact on the subsequent distance estimation, especially in the case of missed and wrong detections, which can have a fatal impact on the overall assessment of distance estimation. Therefore, for this experiment, the selection of a correct vehicle detection model plays an extremely positive role in the distance estimation task and maximises the accuracy of the distance estimation. Moreover, after comparison, it is found that adding our proposed YOLOv7-CBAM-Transformer is more suitable for the detection and ranging of vehicles at different distances than YOLOv7-CBAM and is more accurate in the ranging of long-distance vehicles.

5.3 Analysis and Discussion of Depth Estimation

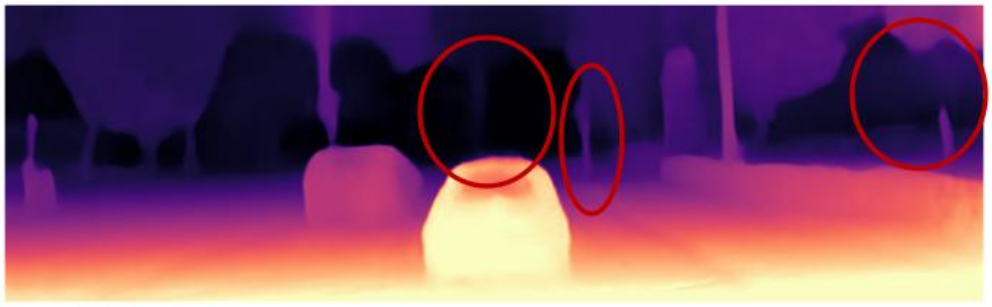
In this part, we provide a comprehensive demonstration of the stability and reliability of our depth estimation models. We investigate the model utilizing Transformer as encoder and the model utilizing CNN as encoder separately, delving into the details of their performance and robustness.

In Figure 5.10, Figure 5.11 and Figure 5.12, we have selected the three most

typical traffic scenes to analyse, to the extent that the performance of the two models can be more intuitively understood. In the scene in Figure.5.10 (Group A), both models compute depth for most of the objects in the scene. However, the Transformer as encoder model is much more sophisticated in terms of detail handling. We have marked the difference between the two depth maps in Figure 5. 10, Figure 5. 11 and Figure 5. 12 with red circles. We can see that the Transformer as encoder model successfully and completely detects the large trees in the middle and on the right side of the scene in Figure 5.10. In contrast, the CNN as encoder model does not detect the trees in the middle of the scene and does not completely detect the trees on the right side of the scene.

Original image

CNN as encoder



Transformer as encoder

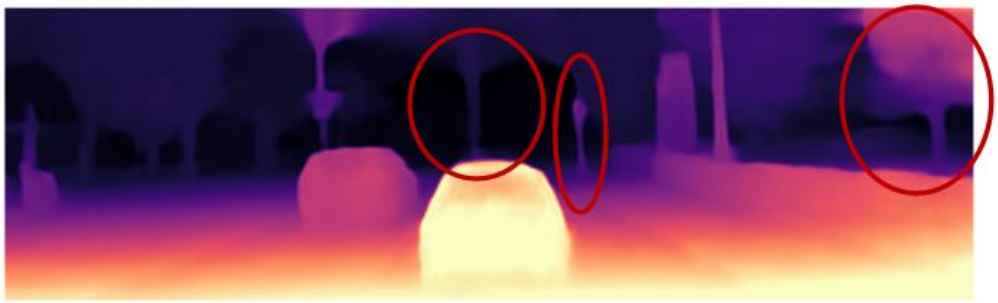


Figure 5.10 Comparison of generated depth images by using Transformer and CNN as the encoder of Models (group A)

In Figure 5.11, we find more advantages of Transformer as encoder. The most obvious one is the stop sign on the far right of the scene, and the pedestrian next to the car. The model with Transformer as encoder also gives full detail to the tree trunk on the right side of the scene. Secondly, the vehicles in the middle of the scene in the distance are also detected more completely by Transformer-based model.

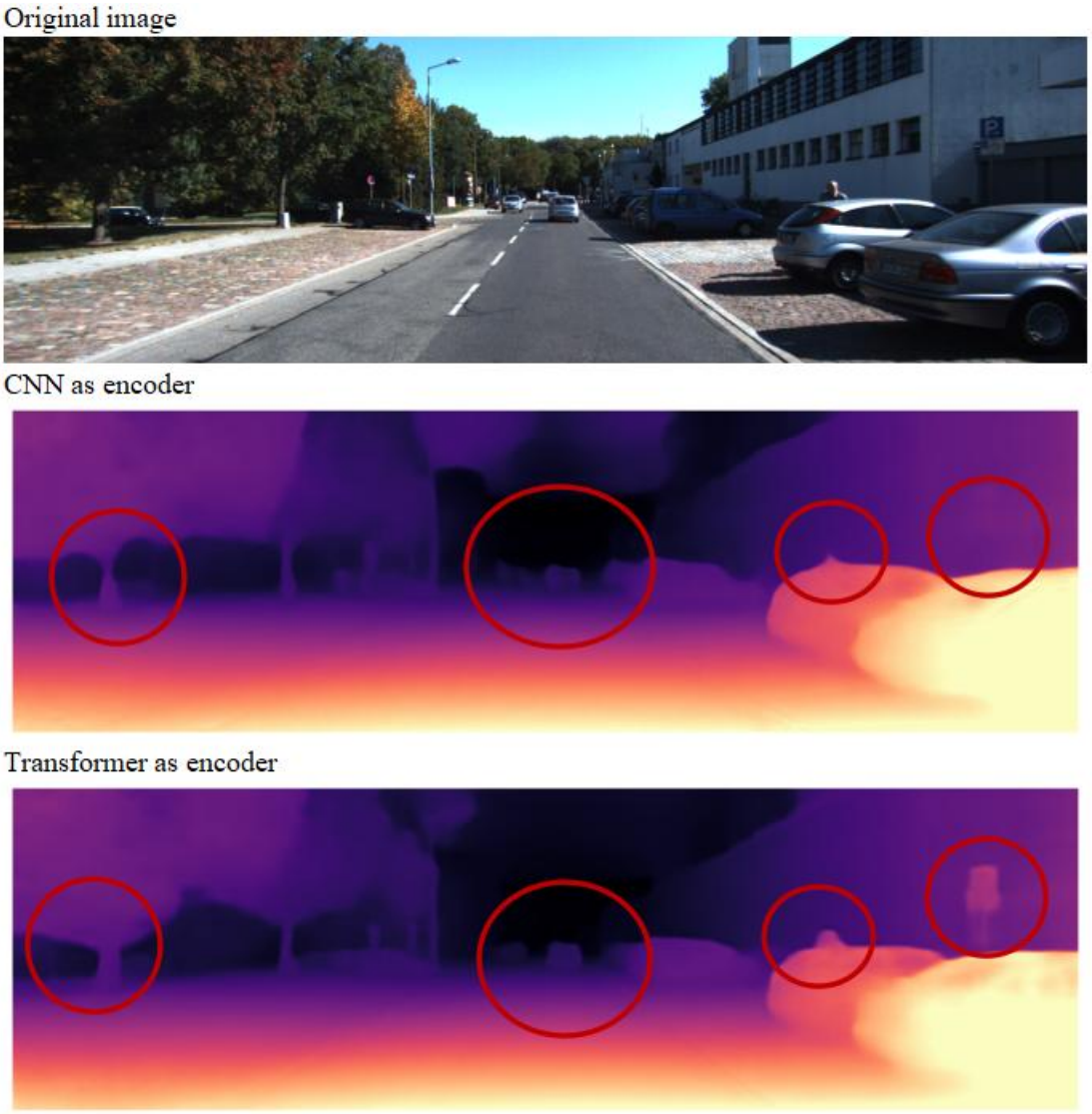


Figure 5.11 Comparison of generated depth images by using CNN and Transformer as the encoder of model (group B)

In Figure 5.12, CNN-based model computes the depth of the sky in the distance to be the same as the depth of the trees in the near distance, except for the treatment of the stop sign hanging from the light pole. In contrast, Transformer-based model handles these details very well.

Through a comprehensive series of comparisons, it becomes evident that the model employing Transformer as an encoder is more proficient in accurately estimating the depth of vehicle-related scenes in the KITTI dataset.

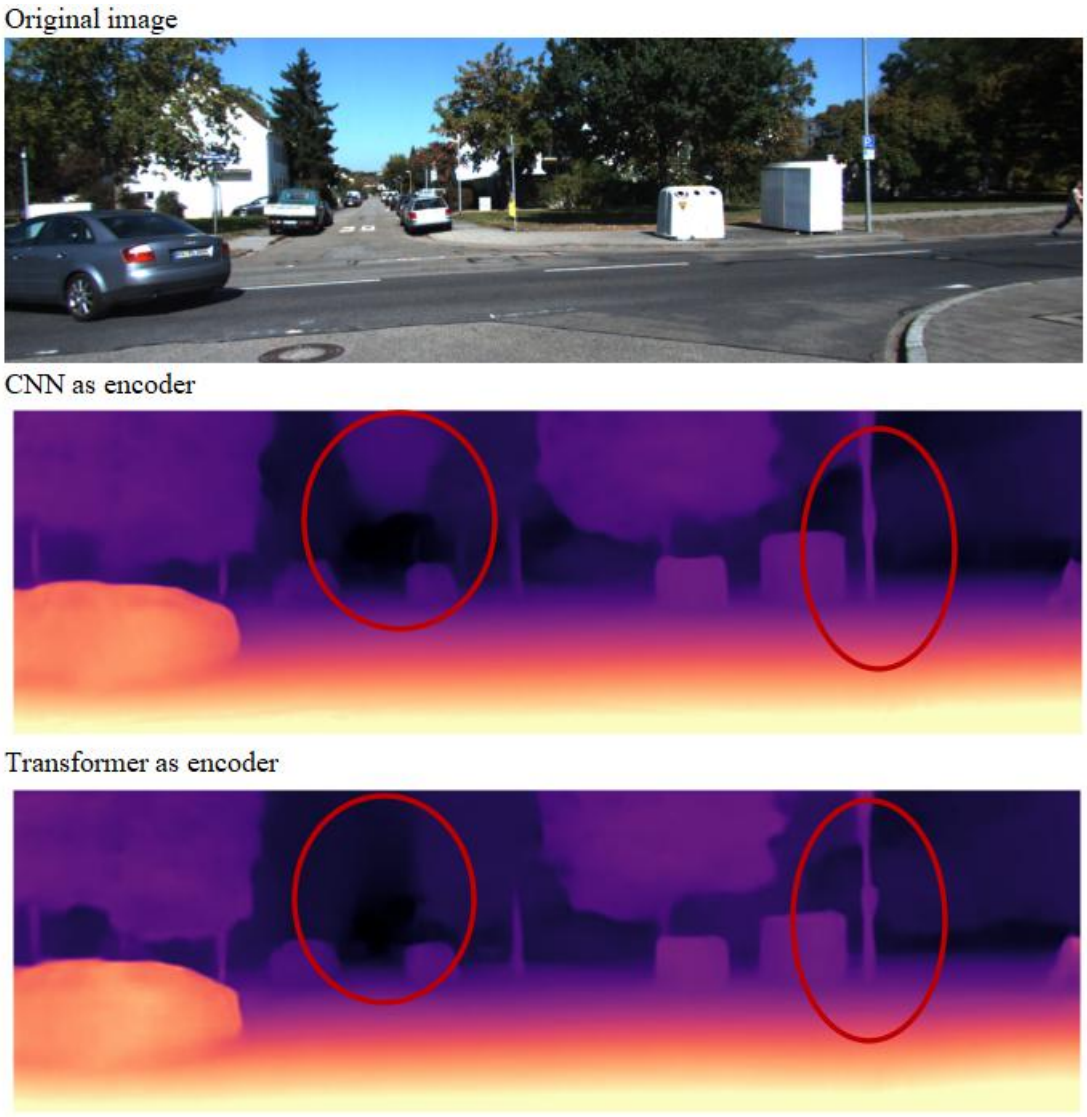


Figure 5.12 Comparison of generated depth images by using CNN and Transformer as

the encoder of model (group C)

5.4 Analysis of Scene Segmentation

In this part, we aim to showcase the efficacy of our scene segmentation model by conducting a comparison of how image resolution affects the model's performance. Additionally, we assess the impact of different models on our Auckland traffic scene dataset.

Table 5.11 Comparing the effect of different image resolutions on the effectiveness of semantic segmentation models

Models	Image resolution	mIOU
SegNet	512×512	58.73%
	1024×1024	60.54%
UNet	512×512	69.25%
	1024×1024	**74.18%**
Ours	512×512	71.83%
	1024×1024	**74.61%**

In Table 5.11, we compare the sensitivity of the different models to the resolution of the images. We have chosen two of the most classical models in semantic segmentation, SegNet and UNet, to compare with our model. The results show that upscaling the resolution from 512×512 to 1024×1024 improves our model mIOU by 2.78%. Meanwhile, it improves by 1.81% and 4.93% on SegNet and UNet, respectively. This shows that appropriately increasing the resolution can most effectively improve the model performance on UNet. However, SegNet is not sensitive to changes in image resolution.

We also took the same number of samples from the Cityscape dataset to train our

model and found the best combination of its optimal batch size and epoch through multiple experiments.

Table 5.12 Comparing the effect of different epochs and batch size on the effectiveness of semantic segmentation models

BatchSize\Epoch	20000 epochs	25000 epochs	30000 epochs	35000 epochs	40000 epochs	45000 epochs
Batch size =16	60.37%	62.91%	64.87%	63.40%	68.93%	69.11%
Batch size =8	67.20%	60.01%	63.48%	74.71%	**75.81%**	75.12%
Batch size =4	64.17%	64.09%	67.79%	72.15%	74.33%	75.27%

We have experimented from 20,000 epochs to 45,000 epochs with 4 batch size, 8 batch size and 16 batch size. Finally, the results shown in Table 5.12 demonstrate 40000 epochs with 8 batch size produced the best performance on the Cityscape dataset. On the Cityscape dataset, our model achieves the best performance of 75.81%. Compared to using our own labelled Auckland traffic dataset, the training results of the public dataset are more accurate. This may be because the labelling of public datasets is more accurate and detailed, and the level of detail of labelling directly affects the training effect.

Chapter 6 Conclusion and Future Work

This chapter delves into a comprehensive exploration of vehicle-related scene understanding, providing detailed explanations of research findings and innovative research methodologies. In this chapter, we present these arguments at a scholarly level. Moreover, we integrate and organize the conclusions within the context, while also identifying potential areas for future work at the conclusion of this thesis.

6.1 Conclusion

The aim of this thesis is to employ deep learning for comprehensive traffic scene understanding, encompassing both 2D and 3D aspects, including scene segmentation, vehicle tracking, depth estimation, and distance estimation. The neural network's effectiveness and the dataset's utility were demonstrated through hyperparameter tuning and experimentation with various model variants. The primary contributions of this thesis are listed as follows:

Every stage of the research, encompassing dataset pre-processing, neural network design and training, model evaluation, and result comparison, was accomplished successfully. The KITTI dataset and the Auckland traffic scenes dataset were utilized as training and testing sets throughout the research. The Auckland traffic scenes dataset was meticulously collected and annotated by ourselves, providing a substantial number of raw images along with precise annotations of the Auckland traffic environment. The dataset includes pixel-level annotations for scene segmentation and bounding box annotations for vehicle tracking. This valuable resource enables effective neural network training and evaluation for various aspects of traffic scene understanding. Moreover, comprehensive comparative validations conducted on the Auckland traffic dataset demonstrate outstanding performance for both scene segmentation models and vehicle tracking models. Additionally, our models exhibit impressive results in scene depth estimation and distance estimation tasks, leveraging the KITTI public dataset. These findings collectively highlight the effectiveness and robustness of our proposed approaches in the domain of traffic scene understanding.

By comparing the different input sizes, flops, parameters of YOLOv5 and four

different YOLOv5 variants (YOLO-fastest, ShuffleV2-YOLOv5, YOLOv4-Tiny, YOLOv3-Tiny), and their performance in vehicle tracking tasks, we found that although the small and lightweight model reduces computation and processing time, it loses accuracy and degrades the performance of the model to a large extent. Experiments show that YOLOv5 outperforms the smallest YOLO-fastest by 13% on mAP@0.5. We also improved YOLOv5 network by adding various attention modules (CA, SE and CBAM) and combined Swin Transformer encoder to these improved YOLOv5 networks and compare the impact of the Transformer and attention models on the vehicle tracking task. By comparing, we added the CBAM and Swin Transformer encoder to achieve 38.9% MOTA and 76.7% MOTP. The experiment demonstrated the Swin Transformer is powerful for helping to improve the combination of attention modules with YOLOv5 network, especially to YOLOv5-CBAM. We also combined SiamRPN with the Hungarian algorithm to achieve multi-target tracking.

In the depth estimation task, we introduced a significant improvement by combination of Transformer encoder and CNN decoder as the architecture. Our comparison between Transformer-based encoder and CNN-based encoder revealed that Transformer-based model performs better in preserving scene details. The experiments showcased that using Transformer as the encoder recieve the RMSE of 0.688 and Sq_rel of 0.745 in the depth estimation model, making it more accurate when evaluated on the KITTI dataset.

For distance estimation, our primary focus was to develop an advanced deep learning-based model tailored for low-cost monocular cameras, with the aim of optimizing hardware costs. By accurately detecting vehicles and applying extended distance estimation vector, our model obtains the bounding box coordinates, enabling precise distance calculations. Through experimentation, we discovered that combining YOLOv7-CBAM-Transformer with the extended distance estimation vector yielded the best results, achieving a remarkable 0.456 improvement in RMSE compared to the

original YOLOv7 model. In our analysis of the results, we observed that the YOLOv7-CBAM-Transformer outperforms the original YOLOv7-CBAM model in distance measurement. It exhibited enhanced accuracy and reduced occurrences of false detections and missed detections. This highlights the significant impact of the Transformer, which contributed to improving the model's feature understanding and its adaptability to diverse distances of vehicles in the scenes.

Combining the VS routeing method with CapsNet, we perform semantic segmentation based on posture and appearance features in scene segmentation. In our research, we employed a capsule network equipped with a diverse set of matrix to achieve exceptional semantic segmentation performance. This was accomplished through our curated dataset of Auckland traffic scenes, which we collected and meticulously annotated. By employing CapsNets and adopting the VS routing method, we observed that the model achieved faster convergence compared to the dynamic routing mechanism. During the experiment, our model demonstrated higher IoUs and better segmentation results compared to U-Net and SegNet. Specifically, our model achieved an IoU of 74.61% on our custom dataset. This proposed technique aims to effectively segment various visual elements in the driving environment, thereby enhancing the safety of autonomous vehicles.

6.2 Future Work

6.2.1 Future Work of Scene Segmentation

(1) To improve scene understanding, we plan to collect more diverse samples in various weather and lighting conditions from multiple locations.

(2) We intend to enhance our training dataset by acquiring a larger number of high-quality images using dedicated camera equipment. This will enable us to capture more specific object characteristics for better scene understanding.

(3) Our dataset will be expanded to include additional classifications such as streetlamps, lane lines, and traffic signals, providing a more comprehensive scene representation.

(4) Fine-tuning the weights and other parameters of our proposed model will be conducted to enhance the precision of smaller classes, such as cars, to achieve more accurate detections.

(5) In order to capture temporal information and understand the behaviours of surrounding vehicles (e.g., lane changes, turns), we plan to incorporate the LSTM module. This addition will help in handling occlusions and enriching the overall scene understanding.

6.2.2 Future Work of Vehicle Tracking

(1) We plan to expand our dataset by including a larger number of diverse automobile instances in the Auckland traffic scenes.

(2) Fine-tuning the neural network will enable us to handle larger input images, allowing the model to learn more precise features.

(3) We aim to add a classification function to the model so that it can not only detect vehicles but also identify their types.

(4) In future research, we intend to enhance the model's functionality by leveraging multitasking capabilities, enabling it to recognize and track other classes of objects in vehicle-related traffic scenes, in addition to detecting vehicles.

(5) To improve vehicle tracking precision, we will expand the dataset to include interference objects, enhance the model's anti-interference capabilities, and reduce the false detection rate of vehicle tracking.

6.2.3 Future Work of Depth Estimation

(1) In our experiments, we utilized Swin Transformer-based encoder to enhance depth estimation accuracy. For future work, we should explore and test various other type of Transformer as encoder, aiming to further reduce the estimation error.

(2) Incorporating time series feature analysis to develop a video-based depth estimation model should be considered. Additionally, efforts to streamline the model's complexity or computational load should be made to achieve real-time estimation and improve fps (frames per second).

(3) To ensure the model's stability and robustness, we should incorporate additional evaluation methods and loss functions, which will provide a comprehensive assessment of its performance and generalization capabilities.

6.2.4 Future Work of Distance Estimation

(1) Our experiment successfully enhanced accuracy by integrating attention modules and the Transformer encoder. For future work, we will explore additional methods to further improve model performance, such as incorporating small object detection layers or combining them with LSTM for enhanced capabilities.

(2) To ensure the model's versatility and generalization, we plan to conduct experiments and tests on various public datasets. Expanding testing to include datasets like cityscapes will help us assess the model's performance under diverse traffic scene conditions.

(3) In future research, we will consider integrating time series feature analysis to develop a video-based distance estimation model. Additionally, we will work on optimizing the model's complexity and computational effort to achieve real-time estimation and improve fps (frames per second).

Abbreviations

VS routing	Vector-Space Routing
CBAM	Convolutional Block Attention Module
CNN	Convolutional Neutal Network
CapsNet	Capsule Neural Network
SE	Squeeze-and-Excitation
CA	Coordinate Attention
LSTM	Long Short-Term Memory
MLP	Multilayer Perceptron
MSA	Multi-head Self-Attention
W_MSA	Window Multi-head Self-Attention
SW_MSA	Shifted Window Multi-head Self-Attention
CAM	Channel Attention Module
SAM	Spatial Attention Module
MOTA	Multitarget Tracking Accuracy
MOTP	Multiobject Tracking Precision
FP	False Positives
FN	False Negatives
RMSE	Root Mean Squared Error

References

Abdallah, M. S., Han, D. S., & Kim, H. (2022). Multi-vehicle tracking using heterogeneous neural networks for appearance and motion features. Intelligent Transportation Systems Research, 20(3), pp.720-733.

Alexey, B., ChienYao, W., & Mark, L. (2020). YOLOv4: Optimal speed and accuracy of object detection. Image and Video Processing, arXiv:2004.10934.

Alfred Daniel, J. et al. (2023). Fully convolutional neural networks for LIDAR–camera fusion for pedestrian detection in autonomous vehicle. Multimedia Tools and Applications, pp.1-24.

Alhashim, I., & Wonka, P. (2018). High quality monocular depth estimation via transfer learning. arXiv preprint arXiv:1812.11941.

Aliew, F. (2022). An approach for precise distance measuring using ultrasonic sensors. Engineering Proceedings, 24(1), pp.8.

Alvarado, S. T., Borja, M. G. B., & Torres, K. B. (2022). Object distance estimation from a binocular vision system for robotic applications using artificial neural networks. Control, Mechatronics and Automation (ICCMA), pp. 19-23.

An, N., Yan, W. (2021). Multitarget tracking using Siamese neural networks. ACM Transactions on Multimedia Computing, Communications and Applications, 17(2s), pp.1-16

Arabi S., Sharma A., Reyes M., Hamann C., Peek-Asa, C. (2022). Farm vehicle following distance estimation using deep learning and monocular camera images. Sensors, 22(7), pp.2736

Ammar, A., Koubaa, A., Ahmed, M., Saad, A., & Benjdira, B. (2021). Vehicle detection from aerial images using deep learning: A comparative study. Electronics, 10(7), pp.820.

Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. (2016). Fully convolutional Siamese networks for object tracking. Computer Vision, Part II, 14, pp.850-865

Bhamidi, S. B. S., & El-Sharkawy, M. (2020). 3-level residual capsule network for complex datasets. Circuits & Systems, pp.1-4.

Bochkovskiy, A., Wang, C. Y, & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv 2004.10934

Bremer, J., Maj, M., Nordbø, Ø., & Kommisrud, E. (2023). Deep learning–based automated measurements of the scrotal circumference of Norwegian Red bulls from 3D images. Smart Agricultural Technology, 3, pp.100133.

Bucher, M., et al. (2021). Handling new target classes in semantic segmentation with domain adaptation. Computer Vision and Image Understanding, 212, pp.103258.

Buhmann, J.; Kuhnel, H. (1992). Unsupervised and supervised data clustering with competitive neural networks. Neural Networks. IJCNN, Vol. 4, pp. 796–801.

Cai, Y., Ding, Y., Zhang, H., Xiu, J., & Liu, Z. (2020). Geo-location algorithm for building targets in oblique remote sensing images based on deep learning and height estimation. Remote Sensing, 12(15), pp.2427.

Carpenter, G.A. & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self organizing neural network. Computer. 21 (3), pp.77–88.

Chang, M., Guo, F., & Ji, R. (2018). Depth-assisted RefineNet for indoor semantic

segmentation. Pattern Recognition, pp. 2680-2685

Chen, C. et al. (2023). A review of hyperspectral image super-resolution based on deep learning. Remote Sensing, 15(11), pp.2853.

Chen, L., Zheng, J., Cao, J., Pan, L., & Zhang, R. (2020). Intelligent traffic sign recognition method based on capsule network. Computer Applications, 40(4), pp.1045.

Chen, T., Kornblith, S., Swersky, K., Mohammad N., & Geoffrey E.H. (2020). Big self-supervised models are strong semi-supervised learners. Advances in Neural Information Processing Systems, 33, pp. 22243-22255.

Chen, Y., et al. (2022). Mobile-former: Bridging MobileNet and Transformer. Computer Vision and Pattern Recognition, pp. 123-149

Chienyao, W., Alexey, B., Mark, L. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. Computer Vision and Pattern Recognition, arXiv:2207.02696

Ci, W., Xu, T., Lin, R., & Lu, S. (2022). A novel method for unexpected obstacle detection in the traffic environment based on computer vision. Applied Sciences, 12(18), pp.8937.

Comesaña, A., Bouza, R., & José, B. (2016). An application of Hebbian learning in the design process decision-making. Intelligent Manufacturing. 27 (3), pp.487–506.

Chuyi, L. et, al. (2022). YOLOv6: A single-stage object detection framework for industrial applications. Computer Vision and Pattern Recognition, arXiv:2209.02976.

Cui, W., Yan, W. (2016) A scheme for face recognition in complex environments. *International Journal of Digital Crime and Forensics* (IJDCF) 8 (1), 26-36.

Deng, L., Yang, M., Qian, Y., Wang, C. & Wang, B. (2017). CNN based semantic segmentation for urban traffic scenes using fisheye camera. Intelligent Vehicles Symposium, pp. 231–236.

Dimililer, K., Ever, Y. K., & Mustafa, S. M. (2019). Vehicle detection and tracking using machine learning techniques. Theory and Application of Soft Computing, pp. 373-381

Doborjeh, M., Liu, X., Shen, Y. et al. (2023) Prediction of Tinnitus treatment outcomes based on EEG Sensors and TFI score using deep learning, Sensors.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E. & Darrell, T. (2014). DeCAF: A deep convolutional activation feature for generic visual recognition. Machine Learning, 32(1), pp.647-655

Dosovitskiy, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Duarte, J. M., & Berton, L. (2023). A review of semi-supervised learning for text classification. Artificial Intelligence Review, pp.1-69.

Elhani, D., Megherbi, A. C., Zitouni, A., Dornaika, F., Sbaa, S., & Taleb-Ahmed, A. (2023). Optimizing convolutional neural networks architecture using a modified particle swarm optimization for image classification. Expert Systems with Applications, 229, pp.120411.

Fukushima, H., Farzad, D., Babette & Torras, C. (2017). Scene Understanding Using Deep Learning. Academic Press, pp. 373-382.

Gai, Y., He, W., & Zhou, Z. (2021, November). Pedestrian target tracking based on DeepSORT with YOLOv5. Computer Engineering and Intelligent Control, pp.1-5

Gao, X., Nguyen, M., Yan, W. (2023) A high-accuracy deformable model for human face mask detection. PSIVT

Garg, R, Vijay, K., Gustavo, C., & Ian, R. (2016). Unsupervised CNN for single view depth estimation: Geometry to the rescue. ComputerVision, Part VIII 14, pp. 740-756.

Gowdra, N., Sinha, R., MacDonell, S., Yan, W. (2021) Maximum Categorical Cross Entropy (MCCE): A noise-robust alternative loss function to mitigate racial bias in Convolutional Neural Networks (CNNs) by reducing overfitting. Pattern Recognition.

Ghafari, M., et al. (2021). Complementary performances of convolutional and capsule neural networks on classifying microfluidic images of dividing yeast cells. PloS One, 16(3), pp.6988.

Gu, Q., Yang, J., Kong, L., Yan, W., Klette, R. (2017) Embedded and real-time vehicle detection system for challenging on-road scenes. Optical Engineering, 56 (6), 063102.

Gu, Q., Yang, J., Yan, W., Klette, R. (2017) Integrated multi-scale event verification in an augmented foreground motion space. Pacific-Rim Symposium on Image and Video Technology (pp.488-500)

Gu, Q., Yang, J., Yan, W., Li, Y., Klette, R. (2017) Local Fast R-CNN flow for object-centric event recognition in complex traffic scenes. Pacific-Rim Symposium on Image and Video Technology (pp.439-452)

Guo, J., Wang, J., Wang, H., Xiao, B., He, Z., & Li, L. (2023). Research on road scene understanding of autonomous vehicles based on multi-task learning. Sensors, 23(13), pp.6238.

Guo, J., et al. (2022). CMT: Convolutional neural networks meet vision transformers. Computer Vision and Pattern Recognition, pp.314-328

Gou, L., Zou, L., Li, N., Hofmann, M., Shekar, A. K., Wendt, A., & Ren, L. (2020). VATLD: A visual analytics system to assess, understand and improve traffic light detection. Visualization and computer graphics, 27(2), pp.261-271.

Guo, Z., Huang, Y., Hu, X., Wei, H., & Zhao, B. (2021). A survey on deep learning based approaches for scene understanding in autonomous driving. Electronics, 10(4), pp.471.

Gupta, A., Anpalagan, A., Guan, L., & Khwaja, A. S. (2021). Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. Array, 10, pp. 100057.

Hao, Z. (2020). The method of recognizing traffic signs based on the improved capsule network. Computer Engineering and Intelligent Control, pp. 22-26

Hassaballah, M., Kenk, M. A., Muhammad, K., & Minaee, S. (2020). Vehicle detection and tracking in adverse weather using a deep learning framework. Intelligent Transportation Systems, 22(7), pp.4230-4242.

He, Z., Yang, Q., Zhao, X., Zhang, S., & Tan, J. (2020). Spatiotemporal visual odometry using ground plane in dynamic indoor environment. Optik, 220, pp.165165.

Herrera, A., Beck, A., Bell, D., Miller, P., Wu, Q., Yan, W. (2008) Behavior analysis and prediction in image sequences using rough sets. International Machine Vision and Image Processing Conference (pp.71-76)

Hinton, G., Krizhevsky, A., & Wang, S.D. (2011). Transforming auto-encoders. Artificial Neural Networks, Part I 21, pp. 44-51

Hinton, E. & Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. Science, 313(5786), pp. 504.

Hinton, G., Sejnowski, T. (1999). Unsupervised Learning: Foundations of Neural Computation. AI Magazine, 22(2), pp.101-101.

Hou, Q., Zhou, D., & Feng, J. (2021). Coordinate attention for efficient mobile network design. Computer Vision and Pattern Recognition, pp.13713-13722.

Hu, A., Cotter, F., Mohan, N., Gurau, C., & Kendall, A. (2020). Probabilistic future prediction for video scene understanding. Proceedings, Part XVI 16, pp. 767-785

Hu, J., Shen, L., Sun, G. (2018). Squeeze-and-excitation networks. Computer Vision and Pattern Recognition, pp.7132-7141

Huang, K. C., Huang, Y. K., & Hsu, W. H. (2021). Multi-stream attention learning for monocular vehicle velocity and inter-vehicle distance estimation. arXiv:2110.11608.

Huang, Z., Lv, C., Xing, Y., & Wu, J. (2020). Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding. Sensors, 21(10), pp.11781-11790.

Ignatious, H. A et al., (2023). Analyzing factors influencing situation awareness in autonomous vehicles—A survey. Sensors, 23(8), pp.4075.

Jabri, A., Owens, A., & Efros, A. (2020). Space-time correspondence as a contrastive random walk. NeurIPS, 33, pp. 19545-19560

Jabri, S., Saidallah, M., El Alaoui, A. E. B., & El Fergougui, A. (2018). Moving vehicle detection using Haar-like, LBP and a machine learning AdaBoost algorithm. Image Processing, Applications and Systems, pp. 121-124

Jeong, Y., Kim, S., & Yi, K. (2020). Surround vehicle motion prediction using LSTM-RNN for motion planning of autonomous vehicles at multi-lane turn

intersections. Intelligent Transportation Systems, 1, pp.2-14.

Ji, P., et al. (2021). Monoindoor: Towards good practice of self-supervised monocular depth estimation for indoor environments. Computer Vision, pp. 12787-12796

Jia, B., & Huang, Q. (2020). DE-CapsNet: A diverse enhanced capsule network with disperse dynamic routing. Applied Sciences, 10(3), pp.884.

Jia, W., et al. (2022). Feature dimensionality reduction: A review. Complex & Intelligent Systems, 8(3), pp.2663-2693.

Junayed, M. S., & Islam, M. B. (2022). Automated physical distance estimation and crowd monitoring through surveillance video. SN Computer Science, 4(1), pp.67.

Karimanzira, D., Pfützenreuter, T., & Renkewitz, H. (2021). Deep learning for long and short range object detection in underwater environment. Adv Robot Automn 5(1), pp.1-10

Kasabov, N. (2018) Time-Space, Spiking Neural Networks and Brain-Inspired Artificial Intelligence. Springer, pp. 457-473.

Kuhn, H. (2012). The Hungarian method for the assignment problem. Naval Research Logistic Quarterly, 2(1-2), pp.83-97

Kumar, C. R. (2021). A comparative study on machine learning algorithms using HOG features for vehicle tracking and detection. Computer and Mathematics Education, 12(7), pp.1676-1679.

Kong, X., Chen, Q., Gu, G., Ren, K., Qian, W., & Liu, Z. (2019). Particle filter-based vehicle tracking via HOG features after image stabilisation in intelligent drive system. IET Intelligent Transport Systems, 13(6), pp.942-949.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., & Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. 3D Vision, pp. 239-248.

Laña, I., Lobo, JL., Capecci, E., Del Ser, J., & Kasabov, N. (2019). Adaptive long-term traffic state estimation with evolving spiking neural networks. Emerging Technologies, 101, pp. 126-144.

Le, R., Nguyen, M., Yan, W. (2020) Machine learning with synthetic data – A new way to learn and classify the pictorial augmented reality markers in real-time. International Conference on Image and Vision Computing New Zealand.

Le, R., Nguyen, M., Yan, W. (2021) Training a convolutional neural network for transportation sign detection using synthetic dataset. International Conference on Image and Vision Computing New Zealand.

Le, R., Nguyen, M., Yan, W., Nguyen, H. (2021) Augmented reality and machine learning incorporation using YOLOv3 and ARKit. Applied Sciences

Lee, D. (2013). Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. Challenges in Representation Learning, Vol. 3, No. 2, pp. 896

Lee, Y., et al. (2022). MPViT: Multi-path vision transformer for dense prediction. Computer Vision and Pattern Recognition, pp. 211-230

Li, B., Yan, J., Wu, W., Zhu, Z., & Hu, X. (2018). High performance visual tracking with siamese region proposal network. CVPR, pp. 8971-8980.

Li, B., et al. (2021). StructDepth: Leveraging the structural regularities for self-supervised indoor depth estimation. Computer Vision, pp. 12663-12673

Li, H., Xiong, P., Fan, H., & Sun, J. (2019). DfANet: Deep feature aggregation for real-time

semantic segmentation. Computer vision and pattern recognition, pp. 9522-9531.

Li, F., Zhang, Y., Yan, W., Klette, R. (2016) Adaptive and compressive target tracking based on feature point matching. International Conference on Pattern Recognition (ICPR), (pp.2734-2739).

Li, P., Nguyen, M., Yan, W. (2018) Rotation correction for license plate recognition. International Conference on Control, Automation and Robotics.

Li, Y., Ming, Y., Zhang, Z., Yan, W., Wang, K. (2021) An adaptive ant colony algorithm for autonomous vehicles global path planning. International Conference on Computer Supported Cooperative Work in Design.

Li, W., Qu, Z., Song, H., Wang, P., & Xue, B. (2020). The traffic scene understanding and prediction based on image captioning. IEEE Access, 9, pp.1420-1427.

Li, X., Wang, W., Hu, X., & Yang, J. (2019). Selective kernel networks. Computer vision and pattern recognition, pp. 510-519.

Li, X., Zhao, Z., & Wang, Q. (2021). ABSSNet: Attention-based spatial segmentation network for traffic scene understanding. Cybernetics, 52(9), pp.9352-9362.

Li, Y., Li, W., Zhao, Z., & Fan, J. (2022). DRI-MVSNet: A depth residual inference network for multi-view stereo images. Plos One, 17(3), pp.721.

Li, Z., Wang, X., Liu, X., & Jiang, J. (2022). Binsformer: Revisiting adaptive bins for monocular depth estimation. arXiv:2204.00987.

Li, Z., Zhou, A., & Shen, Y. (2020). An end-to-end trainable multi-column CNN for scene recognition in extremely changing environment. Sensors, 20(6), pp.1556.

Lian, G., Wang, Y., Qin, H., & Chen, G. (2022). Towards unified on-road object detection and depth estimation from a single image. Machine Learning and Cybernetics,

pp.1-11.

Liu, X. (2019). Vehicle-related Scene Understanding Using Deep Learning. Master's Thesis, Auckland University of Technology, New Zealand

Liu, X., & Nguyen, M., Yan, W. (2019). Vehicle-related scene understanding using deep learn. Asian Conference on Pattern Recognition, 5, pp. 61-73

Liu, X., Yan, W., & Kasabov, N. (2020). Vehicle-related scene segmentation using CapsNets. IEEE IVCNZ, pp. 1-6

Liu, X., & Yan, W. (2021). Traffic-light sign recognition using Capsule network. Springer Multimedia Tools and Applications, 80, pp. 15161-15171.

Liu, X. &Yan, W. (2022). Depth estimation of traffic scenes from image sequence using deep learning, PSIVT.

Liu, X., Yan, W. (2022) Vehicle-related distance estimation using customized YOLOv7. International Conference on Image and Vision Computing New Zealand (IVCNZ)

Liu, X., Yan, W. Kasabov, N. (2023) Moving vehicle tracking and scene understanding: A hybrid approach. Multimedia Tools and Applications.

Liu, X., Yan, W. (2024) Vehicle detection and distance estimation using improved YOLOv7 model. Deep Learning, Reinforcement Learning and the Rise of Intelligent Systems, IGI Global.

Liu, Y., Nand, P., Hossain, A., Nguyen, M., Yan, W. (2023) Sign language recognition from digital videos using feature pyramid network with detection Transformer. Multimedia Tools and Applications.

Liu, Y., Shao, Z., & Hoffmann, N. (2021). Global attention mechanism: Retain information to enhance channel-spatial interactions. arXiv:2112.05561.

Liu, Z., Yan, W., Yang, B. (2018) Image denoising based on a CNN model. International Conference on Control, Automation and Robotics.

Liu, Z, et al. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. ICCV, pp.37-49

Lin, J. T., Dai, D., & Van Gool, L. (2020). Depth estimation from monocular images and sparse radar data. Intelligent Robots and Systems, pp. 10233-10240.

Lu, J., Nguyen, M., Yan, W. (2021) Sign language recognition from digital videos using deep learning methods. International Symposium on Geometry and Vision.

Lobo, JL., Laña, I., Capecci, E., Del Ser J, Bilbao, MN., Kasabov, N. (2018). Evolving spiking neural networks for online learning over drifting data streams. Neural Networks, 108, pp. 1-19.

Lu, K., Zhao, F., Xu, X., & Zhang, Y. (2023). An object detection algorithm combining self-attention and YOLOv4 in traffic scene. PLoS One, 18(5), pp.654.

Lu, Y., & Lu, G. (2021). An alternative of lidar in nighttime: Unsupervised depth estimation based on single thermal image. Applications of Computer Vision, pp. 3833-3843

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., & Kim, T. K. (2021). Multiple object tracking: A literature review. Artificial Intelligence, 293, pp.103448

Luo, Z., Nguyen, M., Yan, W. (2022) Kayak and sailboat detection based on the improved YOLO with Transformer. ACM ICCCV.

Luo, Z., Nguyen, M., Yan, W. (2021) Sailboat detection based on automated search attention mechanism and deep learning models. International Conference on Image and Vision Computing New Zealand.

Ma, C., Huang, J. B., Yang, X., & Yang, M. H. (2020). Hierarchical convolutional features

for visual tracking. Computer Vision, pp.3074-3082

Ma, J., Liu, W., Miller, P., Yan, W. (2009) Event composition with imperfect information for bus surveillance. IEEE International Conference on Advanced Video and Signal Based Surveillance.

Mamun, A. A., Ping, E. P., Hossen, J., Tahabilder, A., & Jahan, B. (2022). A comprehensive review on lane marking detection using deep neural networks. Sensors, 22(19), pp.7682.

Masoumian, A., Rashwan, H. A., Cristiano, J., Asif, M. S., & Puig, D. (2022). Monocular depth estimation using deep learning: A review. Sensors, 22(14), pp.5353.

Mehtab, S., Yan, W. (2021) FlexiNet: Fast and accurate vehicle detection for autonomous vehicles-2D vehicle detection using deep neural network. International Conference on Control and Computer Vision.

Mehtab, S., Yan, W. (2022) Flexible neural network for fast and accurate road scene perception. Multimedia Tools and Applications.

Mehtab, S. Yan, W., Narayanan, A. (2022) 3D vehicle detection using cheap LiDAR and camera sensors. International Conference on Image and Vision Computing New Zealand.

Mehta, S., & Mohammad, R. (2021). MobileViT: Light-weight, general-purpose, and mobile friendly vision transformer. arXiv:2110.02178.

Ming, Y., Meng, X., Fan, C., & Yu, H. (2021). Deep learning for monocular depth estimation: A review. Neurocomputing, 438, pp.14-33.

Ming, Y., Li, Y., Zhang, Z., Yan, W. (2021) A survey of path planning algorithms for autonomous vehicles. International Journal of Commercial Vehicles.

Müller, J., & Dietmayer, K. (2018). Detecting traffic lights by single shot detection. Intelligent Transportation Systems, pp.266-273

Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press. pp.384-391.

Nandi, D., Saif, A. F. M., Paul, P., Zubair, K. M., & Shubho, S. A. (2018). Traffic sign detection based on color segmentation of obscure image candidates: A comprehensive study. Modern Education and Computer Science 10(6), pp.35-46

Nartey, O.T., Yang, G., Wu, J. & Asare, S.K. (2019). Semi-supervised learning for fine grained classification with self-training. IEEE Access, 8, pp.2109-2121.

Nguyen, M., Yan, W. (2023) From faces to traffic lights: A multi-scale approach for emotional state representation. IEEE International Conference on Smart City.

Özcan, M., Aliew, F., & Görgün, H. (2020). Accurate and precise distance estimation for noisy IR sensor readings contaminated by outliers. Measurement, 156, pp.107633.

Pan, C., Yan, W. (2018) A learning-based positive feedback in salient object detection. International Conference on Image and Vision Computing New Zealand.

Pan, C., Yan, W. (2020) Object detection based on saturation of visual perception. Multimedia Tools and Applications, 79 (27-28), 19925-19944.

Pan, C., Liu, J., Yan, W., Zhou, Y. (2021) Salient object detection based on visual perceptual saturation and two-stream hybrid networks. IEEE Transactions on Image Processing.

Pang, H., Xuan, Q., Xie, M., Liu, C., & Li, Z. (2020). Target tracking based on Siamese convolution neural networks. Computer, Information and Telecommunication Systems, pp. 1-5.

Patrick, M. K., et al. (2022). Capsule networks–a survey. Journal of King Saud University-computer and information sciences, 34(1), pp.1295-1310.

Pan, X., et al. (2022). On the integration of self-attention and convolution. IEEE CVPR, pp.201-212

Parker, P. R., Abe, E. T., Beatie, N. T., Leonard, E. S., Martins, D. M., Sharp, S. L., ... & Niell, C. M. (2022). Distance estimation from monocular cues in an ethological visuomotor task. Elife, 11, pp.74708.

Peng, J., et al. (2020). Implementation of the structural SIMilarity (SSIM) index as a quantitative evaluation tool for dose distribution error detection. Medical Physics, 47(4), pp.1907-1919.

Peng, X. et al., (2020). Optical remote sensing image change detection based on attention mechanism and image difference. Geoscience and Remote Sensing, 59(9), pp.7296-7307.

Peng, Z., et al. (2021). Conformer: Local features coupling global representations for visual recognition. Computer Vision. arXiv:2005.08100.

Pinheiro, P., Collobert, R., & Dollár, P. (2015). Learning to segment object candidates. Advances in Neural Information Processing Systems, 28, pp. 1990-1998.

Pinheiro, P. O., Lin, T. Y., Collobert, R., & Dollár, P. (2017). SharpMask: Learning to refine object segments. IEEE ICCV, Part I 14, pp. 75-91.

Poggi, M., Tosi, F., Batsos, K., Mordohai, P., & Mattoccia, S. (2021). On the synergies between machine learning and binocular stereo for depth estimation from images: a survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(9), pp.5314-5334.

Prykhodchenko, R., & Skruch, P. (2022). Road scene classification based on street-level images and spatial data. Array, 15, pp.100195.

Qi, J., Nguyen, M., Yan, W. (2022) Small visual object detection in smart waste classification using Transformers with deep learning. International Conference on Image and Vision Computing New Zealand (IVCNZ).

Qi, J., Nguyen, M., Yan, W. (2022) Waste classification from digital images using ConvNeXt. Pacific-Rim Symposium on Image and Video Technology (PSIVT).

Qi, J., Nguyen, M., Yan, W. (2023) CISO: Co-iteration semi-supervised learning for visual object detection. Multimedia Tools and Applications.

Qi, J., Nguyen, M., Yan, W. (2024) NUNI-Waste: Novel semi-supervised semantic segmentation for waste classification with non-uniform data augmentation. Multimedia Tools and Applications.

Qi, Y., Zhang, S., Qin, L., Yao, H., Huang, Q., Lim, J., & Yang, M. H. (2021). Hedged deep tracking. Computer Vision and Pattern Recognition, pp.4303-4311

Qin, Z., Yan, W. (2021) Traffic-sign recognition using deep learning. International Symposium on Geometry and Vision.

Qu, Z., & Shao, Y. (2020). Recognition method and evaluation of traffic signs based on Capsule network. In CICTP, pp. 376-388.

Rao, Y., Cheng, Y., Xue, J., Pu, J., Wang, Q., Jin, R., & Wang, Q. (2020). FPSiamRPN: Feature pyramid Siamese network with region proposal network for target tracking. IEEE Access, 8, pp.176158-176169.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. ArXiv, abs/1804.02767.

Saffari, M., & Khodayar, M. (2023). Low-rank sparse generative adversarial unsupervised domain adaptation for multi-target traffic scene semantic segmentation. Industrial Informatics, pp.1-13

Sarker, I.H. (2021). Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN COMPUT, 2(6), pp.420

Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. SN Computer Science, 2(3), pp.160.

Shen, H., Kankanhalli, M., Srinivasan, S., Yan, W. (2004) Mosaic-based view enlargement for moving objects in motion pictures. IEEE ICME'04.

Shen, Y., Yan, W. (2019) Blind spot monitoring using deep learning. International Conference on Image and Vision Computing New Zealand.

Shi, M., et al. (2023). A hierarchically sampling global sparse transformer in data stream mining for lightweight image restoration. Advances in Signal, 2023(1), pp.1-18.

Shi, P., et al. (2023). Object detection based on Swin Deformable Transformer-BiPAFPN-YOLOX. Computational Intelligence and Neuroscience, pp.23-37.

Shi, R., Ruiyang, L., Niu, L., & Zhou, R. (2022). Sparse CapsNet with explicit regularizer. Pattern Recognition, 124, pp.108486.

Shu, D. W., Jang, W., Yoo, H., Shin, H. C., & Kwon, J. (2022). Deep-plane sweep generative adversarial network for consistent multi-view depth estimation. Machine Vision and Applications, 33, pp.1-10.

Sun, S., Wang, Y., & Piao, Y. (2021). A Real-time multi-target tracking method based on deep learning. Physics, Vol. 1920, No. 1, pp.12112.

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Colin, A., Raffel, Ekin D., Kurakin, A., & Li, C. (2020). Fixmatch: Simplifying semi-supervised learning with consistency and confidence. Advances in Neural Information Processing Systems, 33, pp.596-608.

Song, W., Jiao, L., Liu, F., Liu, X., Li, L., Yang, S., ... & Zhang, W. (2022). A joint Siamese attention-aware network for vehicle object tracking in satellite videos. IEEE Transactions on Geoscience and Remote Sensing, 60, pp.1-17.

Song, W., Wang, Y., Huang, D., & Tjondronegoro, D. (2018). A rapid scene depth estimation model based on underwater light attenuation prior for underwater image restoration. Pacific Rim Conference on Multimedia, Part I 19, pp. 678-688.

Steuer, N. A. K., & Schwenker, F. (2021). Next-generation neural networks: Capsule networks with routing-by-agreement for text classification. IEEE Access, 9, pp.125269-125299.

Swaraja, K., Akshitha, V., Pranav, K., Vyshnavi, B., Akhil, V. S., Meenakshi, K., ... & Duggineni, C. (2021). Monocular depth estimation using transfer learning-An overview. In E3S Web of Conferences, Vol. 309, pp. 01069.

Tak, S et al., (2021). Development of AI-based vehicle detection and tracking system for C-ITS application. Advanced Transportation, pp.1-15.

Taye, M. M. (2023). Theoretical understanding of convolutional neural network: concepts, architectures, applications, future directions. Computation, 11(3), pp.52.

Vijayanarasimhan, S., et al. (2017). SFM-Net: Learning of structure and motion from video. arXiv:1704.07804.

Vakili, E., et al. (2020). Single-camera vehicle speed measurement using the geometry of the imaging system. Mult. Tools Apps. 79, pp.19307–19327

Vallayil, M., Nand, P., Yan, W., Allende-Cid, H. (2023) Explainability of automated fact verification systems: A comprehensive review. Applied Science, 13(23) 1260

Wang, C., Peng, G., & De Baets, B. (2021). A distance-constrained semantic autoencoder for zero-shot remote sensing scene classification. Selected Topics in Applied Earth Observations and Remote Sensing, 14, pp.12545-12556.

Wang, J., Yan, W., Kankanhalli, M., Jain, R., Reinders, M. (2003) Adaptive monitoring for video surveillance. International Conference on Information, Communications and Signal Processing.

Wang, J., Kankanhalli, M., Yan, W., Jain, R. (2003) Experiential sampling for video surveillance. ACM SIGMM International Workshop on Video surveillance (pp.77-86).

Wang, J., Yan, W. (2016) BP-neural network for plate number recognition. International Journal of Digital Crime and Forensics (IJDCF) 8 (3), 34-45.

Wang, J., Ngueyn, M., Yan, W. (2017) A framework of event-driven traffic ticketing system. International Journal of Digital Crime and Forensics (IJDCF) 9 (1), 39-50.

Wang, J., Bacic, B., Yan, W. (2018) An effective method for plate number recognition. Multimedia Tools and Applications, 77 (2), 1679-1692.

Wang, K., & Liu, M. (2022). YOLOv3-MT: A YOLOv3 using multi-target tracking for vehicle visual detection. Applied Intelligence, 52(2), pp.2070-2091.

Wang, L., Zhang, J., Wang, O., Lin, Z., & Lu, H. (2020). Sdc-depth: Semantic divide-and-conquer network for monocular depth estimation. IEEE CVPR, pp.541-550.

Wang, X., & Yan, W. Q. (2023). Human identification based on Gait manifold. Applied Intelligence, 53(5), pp.6062-6073.

Wang, Y., Li, G., Zhang, R., Chen, X., & Li, T. H. (2022). Geometric-aware calibration

mechanism for self-supervised depth estimation. Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles, pp. 335-342.

Woo, S., Park, J., Lee, J. Y., & Kweon, I. S. (2019). CBAM: Convolutional block attention module. Computer Vision, pp. 3-19

Wu, Y., et al. (2023). A novel Siamese network object tracking algorithm based on tensor space mapping and memory-learning mechanism. Visual Communication and Image Representation, 91, pp.103742.

Xiao, B., Nguyen, M., & Yan, W. Q. (2023). Fruit ripeness identification using transformers. Applied Intelligence, 53(19), pp.1-12.

Xiao, D., et al. (2023). Siamese block attention network for online update object tracking. Applied Intelligence, 53(3), pp.3459-3471.

Xie, L., et al. (2020). Scene recognition: A comprehensive survey. Pattern Recognition, 102, pp.107205.

Xing, J., Yan, W. (2021) Traffic sign recognition using guided image filtering. International Symposium on Geometry and Vision.

Xing, J., Nguyen, M., Yan, W. (2022) The improved framework of traffic sign recognition by using guided image filtering. Springer Nature Computer Science.

Xing, J., Nguyen, M., Yan, W. (2022) Traffic sign recognition from digital images by using deep learning. Pacific-Rim Symposium on Image and Video Technology.

Xu, Y., Zhang, J., & Brownjohn, J. (2021). An accurate and distraction-free vision-based structural displacement measurement method integrating Siamese network based tracker and correlation-based template matching. Measurement, 179, pp.109506.

Yan, W. Q. (2019). Introduction to Intelligent Surveillance: Surveillance Data Capture, Transmission, and Analytics. Springer, pp. 1-6.

Yan, W. (2021). Computational Methods for Deep Learning: Theoretic, Practice and Applications. Springer

Yan, Z., et al. (2015). Bodypart recognition using multi-stage deep learning. Medical Imaging, 24, pp. 449-461.

Yang, K., et al. (2020). SiamAtt: Siamese attention network for visual tracking. Knowledge-Based Systems, 203, pp.106079.

Yao, Z., & Wang, L. (2023). Object localization and edge refinement network for salient object detection. Expert Systems with Applications, 213, pp.118973.

Yeh, C., Huang, Y., Lin, C., & Chang, C. (2020). Transfer2Depth: Dual attention network with transfer learning for monocular depth estimation. IEEE Access 8, pp.86081-86090.

Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., & Sang, N. (2021). BiseNet v2: Bilateral network with guided aggregation for real-time semantic segmentation. Computer Vision, 129, pp.3051-3068.

Zalevsky, Z. et al., (2021). Light detection and ranging (Lidar): Introduction. JOSA A, 38(11), pp.LID1-LID2.

Zhang, C., Ding, W., Peng, G., Fu, F., & Wang, W. (2020). Street view text recognition with deep learning for urban scene understanding in intelligent transportation systems. Intelligent Transportation Systems, 22(7), pp.4727-4743.

Zhang, D. (2023). STA-YOLOv7: Swin-Transformer-enabled YOLOv7 for road damage detection. Computer Science and Application. 13. pp.1157-1165.

Zhang, J., Huang, B., Ye, Z., Kuang, L. D., & Ning, X. (2021). Siamese anchor-free object tracking with multiscale spatial attentions. Scientific Reports, 11(1), pp.22908

Zhang, J., Yang, K., Constantinescu, A., Peng, K., Müller, K., & Stiefelhagen, R. (2022). Trans4Trans: Efficient transformer for transparent object and semantic scene segmentation in real-world navigation assistance. Intelligent Transportation Systems, 23(10), pp.19173-19186.

Zhang, X., et al. (2023). Unsupervised monocular depth and camera pose estimation with multiple masks and geometric consistency constraints. Sensors, 23(11), pp.5329.

Zhang, Y et al (2019). An unsupervised parameter learning model for RVFL neural network. Neural Networks, 112, pp.85-97.

Zhao, C., Sun, Q., Zhang, C., Tang, Y., & Qian, F. (2020). Monocular depth estimation based on deep learning: An overview. Science China Technological Sciences, 63(9), pp.1612-1627.

Zhao, C., Zhang, Y., Poggi, M., Tosi, F., Guo, X., Zhu, Z., ... & Mattoccia, S. (2022). MonoVit: Self-supervised monocular depth estimation with a vision transformer. 3D Vision, pp. 668-678.

Zheng, K., Yan, Q., Nand, P. (2017) Video dynamics detection using deep neural networks. IEEE Transactions on Emerging Topics in Computational Intelligence.

Zheng, Q., et al. (2022). Progressively real-time video salient object detection via cascaded fully convolutional networks with motion attention. Neurocomputing, 467, pp.467

Zheng, Y., Liu, P., Qian, L., Qin, S., Liu, X., Ma, Y., & Cheng, G. (2022). Recognition and depth estimation of ships based on binocular stereo vision. Journal of Marine Science and Engineering, 10(8), pp.1153.

Zhou, K., Wang, K., & Yang, K. (2020). PADENet: An efficient and robust panoramic monocular depth estimation network for outdoor scenes. Intelligent Transportation Systems, pp.3849

Zhou, L., Yan, W., Shu, Y., Yu, J. (2018) CVSS: A cloud-based visual surveillance system. International Journal of Digital Crime and Forensics (IJDCF) 10 (1), 79-91.

Zhu, W., Peng, B., Yan, W. (2014) Dual knowledge distillation on multiview pseudo labels for unsupervised person re-identification. IEEE Transactions on Multimedia.

Zhu, Y., Yan, W. (2022) Image-based storytelling using deep learning. ACM ICCCV.

Zhu, Y., & Yan, W. Q. (2022). Traffic sign recognition based on deep learning. Multimedia Tools and Applications, 81(13), pp.17779-17791.