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

Master of Computer and Information Sciences (MCIS)

2019

School of Engineering, Computer and Mathematical Sciences

Abstract

Automated driving technology is an inevitable trend in the future development of transportation, it is also one of the eminent achievements in the matter of artificial intelligence. Primarily deep learning produces a significant contribution to the progression of automatic driving. Deep learning not only promotes autonomous vehicles to sense and identify the surrounding environment, but also identifies and classifies various information regarding to vehicles. With the upgrades and improvement of deep learning technology, it can be promptly and readily learned and employed. A large number of pretraining networks and public datasets have provided convenience for training numerous traffic scenes. Nevertheless, automated driving technology is not flexible enough to understand scenes in complex traffic environments, with regard to traffic rules and transportation facilities in various countries. There is no algorithm so far designed for all traffic scenes.

In this thesis, our contributions are that we primarily deal with the issue of understanding of vehicle-related scene using deep learning. To the best of our knowledge, this is the first time that we utilize Auckland traffic environment as an analysis object for scene understanding. Moreover, automatic scene segmentation and object detection are coalesced for traffic scene understanding. The techniques based on deep learning dramatically decrease human manipulations. Furthermore, the datasets in this project provide a large amount of Auckland traffic data. Meanwhile, the performance of CNN processing is consolidated by combining with vehicle detection outcome.

Keywords: Traffic scene understanding, deep learning, automatic driving, image segmentation, object detection.

Table of Contents

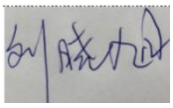
List of Figures

List of Tables

List of Algorithms

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 September 2019

Acknowledgment

First of all, I would like to appreciate my supervisor Wei Qi Yan. Dr. Yan provided me with extremely professional academic guidance, trained my efficient study habits and study skills during the research. Moreover, I have benefited from the theoretical lectures provided by my supervisor regularly.

Secondly, I must express my gratitude to the Auckland University of Technology for supplying the postgraduate courses. In the first year, I consolidated my basic knowledge through four courses and laid the foundation for my 120 points thesis. For this thesis project, the experience and help provided by the courses studied in the first half of the year are invaluable. Moreover, the learning facilities provided by the Auckland University of Technology, such as libraries and laboratories, also facilitate my study and research.

Finally, I would like to thank my family for their encouragement of my studies and provide financial support for my research and living in New Zealand.

Xiaoxu Liu

Auckland, New Zealand

September 2019

Chapter 1
Introduction

This chapter mainly includes five parts. Firstly, the overview and necessity as well as the significance of scene understanding using deep learning in recent years in the field of the vehicle-related scene will be construed. Sections 1.2 *and* 1.3 *will list the main research questions that will be discussed in this thesis and propose meaningful contributions in the field of deep learning. Section* 1.4 *will explicate the important significance and the final implementation of the function. Finally, the details regarding this thesis and the core content of each chapter will be depicted in Section* 1.5.

1.1 Background and Motivation

Intelligent surveillance is one of the significant research tendency concerning artificial intelligence. With the advancement of technology and usability of intelligent surveillance, the utilization is more convenient, resulting in its broad implementation in highways and vehicles. Intelligent surveillance has become a necessary technique safeguarding public and private safety and security, amalgamating computer science with engineering and multidiscipline, including computer vision, digital image processing, and computer graphics as well as computational intelligence (Yan, 2019).

In the achievements, the spinout of automatic vehicles has become an icon of the intelligent surveillance technology. The autonomous driving requires several features to achieve driving without human interference. One of these features is vehicle detection. The objective of this process is to assist the "Central Processing Unit" of the vehicle to discern what is around the vehicle, in an attempt to evaluate the circumstance to make the best decision for each situation in real-time (Mohamed, Hossam, Ahmed & Sherif, 2018).

In addition to shallow semantic understandings, automated vehicles also need to deal with various high-level semantic tasks. For example, the interrelationships between objects in a driving scene, and the understanding and prediction of events such as pedestrians, vehicles, scenes by self-driving vehicles. In addition, the understanding of sudden change in the scene caused by the behaviour of turning left and right, making a U-turn and downhill is also crucial for autonomous vehicles. Therefore, the vehicle-related scene understanding plays a vital role in autonomous vehicles.

The so-called scene understanding is the process of cognizing and inferring the scene based on spatial perception (Yong et al., 2019). In the vehicle-related scene understanding, the scene is the environment in which the vehicle is currently located. It includes the location, the person, the event, and the relationship between the three. Scene

understanding mainly includes object detection and recognition, semantic segmentation and the relationship between objects. For autonomous vehicles, scene understanding is the premise, because the vehicle can only make the vehicle's mobile control decision after autonomously perceiving the traffic road scene environment. It can accurate representation and in-depth understanding of the scene presents the vehicle an opportunity. Vehicle-related scene understanding also provides a reasonable understanding of the surroundings, enabling different tasks to be completed in an essential and safe means (Samui, Roy & Balas, 2017).

The scene information in the video is extraordinarily dense and has great discrepancy and complexity. Recently, due to the development of deep learning technology, the use of deep learning technology can significantly ameliorate the performance of video analysis, which is a method of using machine-pair perception scenes. One of the main reasons attributed to such representations in deep learning has been found is related to extensive usage of computer vision.

In addition, deep learning has an active merit of transfer learning that a myriad of pretraining networks and public datasets have provided benefits for training numerous traffic scenes. For vehicle-related scenes, to understand the objects, scenes, and events in the video, the deep learning neural network attempts to emulate the high-level abstraction in the data and encode it as a robust representation (Husain, et al, 2017).

At present, deep learning can understand the scene from five aspects. First, the model can determine the classification of each pixel through deep learning. Secondly, the deep neural network can also realize the region recognition in the scene with the boundary position as the focus of learning; more deeply, deep learning neural network can realize the classification of objects in the scene; Furthermore, it can even identify the current environment (streets, highways, mountain roads, beaches) based on the features in the image, and generate complete event descriptions (overtaking, braking, turning) based on a series of dynamic features.

In the object detection task, R-CNN, Fast R-CNN, Faster R-CNN, YOLO, and other deep learning networks have come out one after another, the object detection accuracy is getting higher and higher, and the speed is getting faster and faster. In image segmentation, the method based on deep learning has also achieved prodigious achievements. Unlike traditional machine learning models, deep learning neural networks can learn more in-depth semantic information in the scene. For example, Microsoft CaptionBot is a typical deep learning-based product that generates a description of the scene based on the information in the image. The description includes detailed object classification, the name of the famous person, and the positional relationship of the objects in the scene. Therefore, based on the high-level semantic learning ability and the high precision of the output, the deep learning method has unique advantages in the field of scene understanding.

Notably, deep learning is more ubiquitous and ameliorated than traditional methods. The applications of deep learning to image classification and object detection have resulted in significantly better results than traditional methods in many open competitions. The error rate of image classification for ILSVRC is refreshed every year as the model increasingly deepen. The error rate of image classification is also continuous decreasing and has been reduced to around 3.08% (Szegedy et al., 2016). On the same ImageNet dataset, human eye recognition error rate is about 5.1%. Models are all designed and implemented for image classification, but they all solve one of the most fundamental problems, namely, more powerful feature representation. The deep learning model is used to obtain more powerful feature representations. Therefore, deep learning can achieve higher precision than the traditional method in the scene understanding task (Hui, Kunfeng & Feiyue, 2017).

The end-to-end nature of deep learning is also one of the advantages of scene understanding. It achieves faster information processing speed than traditional methods under the premise of a particularly accurate understanding of the scene. For autonomous vehicles that demand to understand the information in real-time in complex traffic scene, the method of deep learning can more effectively satisfy the accuracy and real-time

requirements of scene understanding. Research investigators have implemented convolutional neural networks for learning automated vehicle controllers with end-to-end frames. Compared to controllers based on traditional display decomposition scene understanding (including lane marking detection, path planning and vehicle control), the deep learning end-to-end controller can optimize these steps simultaneously (Yang et al., 2019).

Automated driving technology using deep learning is not mature enough to understand scenes and objects in complex traffic scene, due to the diversities in traffic rules and transportation facilities in various countries. Currently, it is difficult to apply to all traffic scene using only a single algorithm (Y. Jin et al., 2017). The two essential branches of scene understanding give us inspiration: object detection identifies all objects of a predefined category on the image and positions them through a bounding box. Semantic segmentation operates at a finer scale; its purpose is to parse images and associated class labels with each pixel (Nikita et al., 2017). Although they are two similar tasks, few studies currently merged the two.

The primary research goal of this thesis is to understand the scene related to vehicles. This thesis splits the process of vehicle-related scene understanding into four parts: (1) collecting raw datasets, (2) generating the labelled datasets, (3) training neural network models, (4) establishing model evaluations, validations, and exploration.

Moreover, this thesis will also construe and explicate the core methods of vehicle-related scene understanding. In Chapter 2, the methods for vehicle detection and scene segmentation using various techniques are described in detail. Besides, this thesis will mainly implement vehicle-related scene understanding through deep learning techniques.

1.2 Research Question

Understanding of vehicle-related scenes is implemented through segmenting images of scenes and detecting vehicles within such traffic scenes. Following this research

hypothesis, this thesis mainly raises the following research questions:

Question:

"What deep learning techniques are suitable for vehicle-related scene understanding?"

We detail this generalized question to:

"What are the algorithms for vehicle-related scene understanding in deep learning? Which algorithm is suitable for our research?"

The fundamental aim of this thesis is to detect and segment vehicle-related scenes. Therefore, it is necessary to evaluate the performance of several techniques we apply to understand vehicle-related scenes. Based on the comparisons of algorithm performance, an appropriate method is selected for our scene understanding.

1.3 Contributions

In this project, image segmentation and vehicle detection for vehicle-related scene understanding through deep learning significantly reduce human workload. What is more, the datasets in this project provide a huge amount of Auckland traffic data. Simultaneously, adjustments and ameliorations have been implemented to the two networks for scene understanding.

1.4 Objectives of This Thesis

First of all, this thesis completes the understanding of vehicle-related scenes through deep learning from two aspects, which are vehicle detection and scene segmentation.

Secondly, so as to generate high-precision detectors, this thesis emphasizes the construction of neural networks. In the following, we will adjust the neural network architecture from follow aspects: (1) Experimentations with the same kind of networks generated based on different pre-training convolution models, (2) evaluations and

validations for the multiple models, (3) accuracy comparisons for the same detector with different networks layers.

Finally, this thesis will take advantage of multiple assessment methods for neural networks to evaluate the accuracy and stability of the model. In summary, it is essential to discover an optimum algorithm for scene understanding related to vehicles from various perspectives.

1.5 The Structure of This Thesis

The basic structure and content of this thesis are as follows:

In the second chapter, the previous literature will be reviewed and discussed, including recent methods and models employed to implement object detection and image segmentation. Therefore, Chapter 2 will review our networks from three perspectives: type, structure, and assessment methods of neural network models.

In the third chapter, the research methods of this thesis will be to interpret on details from the perspective of mathematical theory. Besides, this chapter will introduce image preprocessing, dataset acquisition and processing, principles of model evaluation methods, etc.

In Chapter 4, the experimental results and datasets are summarized in tabular and graphical ways. Moreover, a solution to the core problem advocated in this thesis will be proposed based on our experimental results.

In Chapter 5, we will discuss and study the experimental results, tables, and diagrams in Chapter 4.

Chapter 6 is related to a summary of the entire thesis and the plan for our future work.

Chapter 2
Literature Review

With comprehensive examination of the research question and reasonable reviews of the previous studies, the focus of this thesis is on vehicle-related scene understanding from videos, for instance, class-based segmentation of objects and object detection in digital images. In this chapter, we will review and summarize the achievements of research work in the case of scene understanding over the past few years.

.

2.1 Introduction

With the advancement of autonomous driving technology, the understanding of traffic scenes has become a significant research problem in the field of computer vision and also a hot topic in the matter of artificial intelligence. The current literature covers a wide range of solutions to address the understanding of traffic scenes from specific aspects. Jeong et al. (2019) identified the circumstance by measuring the position between the target object and the non-target object to complete the scene understanding. The performance of scene representation and understanding was improved by developing object location extractors to extract local object information from the scene (Lee and Yong, 2018). The model (Park et al. 2019) achieved the goal of vehicle-related scene understanding by fine-grained reasoning. The model performs object detection by labelling pixels belonging to the same target in the closed region as the same category. The detector takes advantage of the road lane as a reference to detect objects that may affect the driving situation and excludes objects that may not affect the driving condition from the recognition task.

For the reason why the characteristics of deep learning play an essential role in the research of scene understanding, high-quality scene understanding models are often achieved through deep learning. First, the layer-by-layer processing of deep learning enables the model to better express the information in the current traffic scene. By simulating the deep structure of the human brain and the gradual abstract cognitive process, deep learning model obtains higher-level expressions through a linear or non-linear combination based on the low-level expression. Therefore, the deep learning method enables the model to analyse complex traffic scenes like humans: it has a hierarchical information processing structure which is similar to the human brain, progressive abstract internal features, and sufficient model complexity. These characteristics of deep learning enable the model to understand the high-level semantics of traffic scenes (traffic event analysis, logical relationships of objects in traffic scenes).

Currently, deep learning can accurately segment lane trajectories to understand road conditions in the scene (Yao et al., 2017). It is also possible to predict overtaking, lane change and braking events by dynamically detecting the positional relationship between the two vehicles. (Wei et al., 2017)

Secondly, the end-to-end feature of deep learning has also made an extraordinary contribution to the development of scene understanding. As the population and the number of vehicles in New Zealand increase, the complexity of traffic scenes continues rising. Therefore, autonomous vehicles have higher requirements in terms of real-time performance. The model of traffic scene understanding based on deep learning utilizes end-to-end features to optimize all tasks (vehicle detection, pedestrian detection, path planning) simultaneously in a short period. For example, an agent with an end-to-end vehicle controller can detect obstacles in the scene and navigate them in real-time and accurately follow the curved lanes (LeCun et al., 2005). In the model designed by Bojarski et al. (2016), vehicle with end-to-end controllers can operate in lanes with and without lane markings.

Thirdly, deep learning algorithm has strong versatility. Faced with the many tasks involved in scene understanding, deep learning model does not require redesigning new algorithms for each task like traditional algorithms. Currently, each deep learning algorithm is suitable for a variety of scene understanding tasks. For example, Faster R-CNN model achieves outstanding results in tasks such as vehicle detection, pedestrian detection and lane detection.

Finally, the features learned by the deep learning model have active mobility. A mature autonomous vehicle must contain a large number of scene understanding tasks, and it takes a vast of time and experience to train each scene to understand the tasks from scratch. Deep learning model learns features from one task which can be used very well on other tasks. For example, the object features learned on ImageNet can also achieve outstanding results in the scene classification.

In the past few years, there has been a huge amount of research work related to image segmentation. Similarly, the technology of object detection has also achieved a great deal of meaningful breakthroughs. However, few studies have so far combined image segmentation and object recognition as well as deep learning together to understand the scenes. The core purpose of traditional algorithms is to design an approach for detecting traffic objects such as roads, vehicle, signs, traffic lights or pedestrians, independently in one environment (Melih & Celenk, 2017). Nevertheless, this thesis not only completed the detection of a single object but also implemented the combination of detection and semantic segmentation (Zhou, Lv, Jiang & Yu, 2019) to complete the scene understanding using deep learning.

Semantic segmentation is essential for scene understanding to classify each pixel in the image. Especially for the detection of road available areas, the identification of vehicles and pedestrians ahead, and understanding of traffic signs, the accuracy of detection and recognition of these objects directly determines the safety performance of autonomous vehicles. At the same time, image information can be applied to understand scenes in bad weather conditions, such as foggy, snowing, sand storming, which have a lower expenditure than object detection that relies on lidar. Simple object detection will not necessarily measure the relative positional information of the scene. For that reason, efficacious and precise image segmentation considerably assists the autonomous vehicle in sensing the surrounding driving environment (Robert, 2009).

In this thesis, the understanding of vehicle-related scenes is implemented in conjunction with a deep learning-based vehicle detector and a semantic segmenter. As compared to other machine learning methods, deep neural networks can improve accuracy by increasing the amount of training data and introducing sophisticated methods to ameliorate efficacy and accuracy. Additionally, the deep learning model eliminates feature extraction as an end-to-end model, because the data is passed directly to the input layer, and the network can acquire excellent performance. Finally, the underlying concepts and techniques for using deep learning in various spheres are commonly

transferable. Hence, deep learning techniques can be much readily adapted to various areas and applications.

2.2 Deep Learning

Deep learning is a technology that embodies the theory of artificial intelligence by establishing deep artificial neural networks with a hierarchical structure. It is also one of the most popular machine learning methods extensively applied to the area of computer vision, speech recognition, and natural language processing (Hongming et al., 2018). It is called “deep learning” because it was designed with deep ANN (Artificial Neural Networks) models having a hierarchy structure along with a huge amount of data to update the parameters and pursue training goals (Bengio, 2009). In the early days, a multilayer feedforward neural network with only one hidden layer consisting of enough neurons could approximate any complex continuous function with arbitrary precision (Hornik, 1989). However, because a single hidden layer of neural network requires a long computation time, the ability to express complex functions with insufficient samples and restricted computational units is exceedingly limited, the generalizing ability of the neural network with only one layer of the hidden layer on the complex classification problem is dramatically reduced.

In contrast, deep learning can achieve complex function approximation through deep nonlinear structures, characterize the distributed representation of input data. In the meantime, deep learning also demonstrates the power of learning from a small sample set. Hinton and Salakhutdinov (2006) demonstrated that deep artificial neural networks have stronger learning capabilities than single hidden layer of neural networks. More importantly, its learning characteristics have more meaningful data representations for eminent visualization or classification performance; Secondly, the “layered pre-training” mechanism in deep learning effectively overcomes the difficulty of training. Thus, deep learning with end-to-end learning characteristics and powerful learning capabilities is responsible for it commonly to be used in various fields.

In 2012, AlexNet, which is one of the largest CNNs, was trained in the ImageNet Large-Scale Visual Identity Challenge (ILSVRC)-2010 and ILSVRC-2012 Competition 2 using the ImageNet subset and gained the highest record on these datasets. This convolutional neural network consists of five convolutional layers and comprised of 60 million parameters and 650,000 neurons, some of them are followed by the max pooling layer and three fully connected layers, with the first 1000 softmax functions motivating a faster training. It also makes use of "pressure difference" in fully connected layers to mitigate overfitting (Hinton et al., 2012).

In 2013, the feedforward convolutional neural network was the first to conduct end-to-end training in supervised learning. Moreover, using large-scale multiscale raw pixels engenders exceptional achievement on standard scene parsing datasets (LeCun et al., 2013). The model exploits thoroughly supervised training of fully labelled images to learn relevant low-level and intermediate features without relying on engineering features (Farabet et al., 2013), which has made significant breakthroughs in scene labelling.

In addition, deep learning not only has a major impact on classification and labelling but also made significant progress in object identification as early as 2010. Training multi-level hierarchies through unsupervised learning is one of the effective methods for learning sparse convolution features. It trains the convolution operations by reducing the redundancy between feature vectors at adjacent locations as well as using the large image window sliding. These operations effectively improve the overall performance of the model. At the same time, they trained a capable of feedforward encoder to predict the quasi-sparse characteristics of the input (Kavukcuoglu, 2010).

Furthermore, the research problems such as machine translation and semantic mining can also take in from deep learning. For instance, the word2vector model developed in 2013 can better express grammatical information rather than merely learning the words itself like a traditional model (Tomas, 2013).

In the new field of deep learning, in addition to the impressive convolutional neural network, the emergence of CapsNet has also promoted the development of computer

vision. For the purpose of combining the relative relationship between objects, the CapsNet observes the probability of the vector as output, such as posture (position, size, direction), deformation, speed, etc. CapsNet mainly tackles the difficulty that the internal representation of convolutional neural networks does not take the crucial spatial hierarchy between simple and complex objects into consideration. CapsNet activates various features of a class of objects in units of capsules composed of several neurons. As the number of capsules with consistent output increases, the correct rate of detection increases. For the reason why CapsNets can identify objects according to the poses, instead of having to identify objects, they also have their spatial relationships as part of the object. As a result, CapsNet output appears to be more meaningful than a convolutional neural network in multiple cases (Hinton, 2017).

Figure 2.1 shows the workflow of a CapsNet. It roughly describes the working steps as the matrix multiplication of input vectors, scalar weight of input vectors, the sum of weighted input vectors, and vector-to-vector nonlinear transformation. The input vector of the capsules comes from the output of three capsules at lower level. The probability of corresponding objects is detected by using the lower capsules. The vector reflects internal states of the detected object. These vectors are then multiplied by using the corresponding weight matrix **W**, which reflects the spatial relationship between the objects at low and high levels. After multiplying the weight matrix, we get the predicted position $\hat{u}_j$ of the object. Unlike convolutional neural networks that apply backpropagation algorithms to update weights, CapsuleNets take advantage of dynamic routing mechanisms to determine which higher level capsules should be updated (Hinton, 2017).

In comparison with the linear weighted summation of fully connected neural networks, the weighted summation of CapsNets S_j adds a coupling coefficient c_{ij} (Jianhuang et al., 2018). It is expressed as

$$c_{ij} = soft\max(b_i) = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})} \tag{2.1}$$

Meanwhile,

$$b_{ij} = b_{ij} + \hat{u}_j \cdot v_j \tag{2.2}$$

where c_{ij} is coupling coefficients, b_{ij} is logarithmic prior probabilities that capsule i should be coupled to capsule j.

Another major innovation in CapsNet is the novel nonlinear activation function, which accepts a vector and scales the input vector to the unit length. Specifically (Hinton, 2017),

$$v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|} \tag{2.3}$$

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

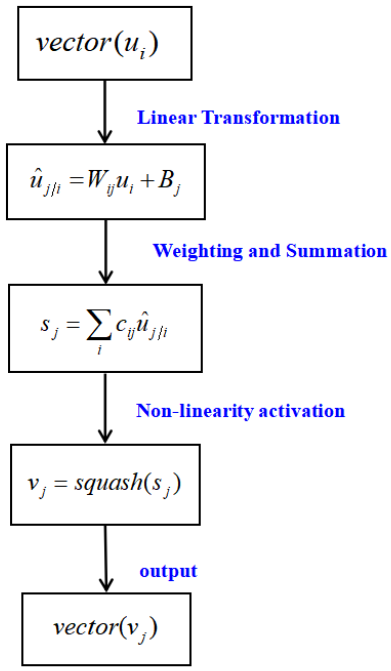


Figure 2.1 The workflow of a CapsNet

In summary, CapsNet can maintain the accuracy around 70% while the parameters are sparsing. Moreover, because CapsNets adopt activities that vary with viewpoints, the

object it learns is a directional spatial relationship, its prediction results are not affected by any viewpoint (Hinton, Krizhevsky & Wang, 2011).

2.3 Convolutional Neural Networks

In the 1960s, Hubel et al. (1962) demonstrated that the process of visual information transmission is that the retina transmits visual information to the receptive field, the multilayer receptive field transmits information to the cerebral cortex. Then, Fukushima (1980) suggested the unsupervised theoretical model “Neocognitron” based on the receptive field, which is a self-organizing multilayer neural network and the characteristics of each layer are from the convolution kernel with shared weights from the upper part. Moreover, its regional derivation function causes the response of each layer of the new generation to be activated by using the local sensory field of the upper layer. Therefore, object recognition using this model is not affected by any factors such as position, small shape changes, and scale. The function of regional derivation also lays an important foundation for today's convolutional neural networks (Yandong et al., 2016).

In 1998, the first convolutional neural network LeNet-5 (Lecun, 1998) was developed. It mainly consists of an input layer, a convolution layer, a downsampling layer, a fully connected layer, and an output layer. The convolutional nucleus of the convolutional layer acts as a receptive field in the biological neural network, stimulates the local information of the lower layer to a higher layer of the network. The concatenated convolution and downsampling layers convert an input image into a collection of feature maps. Finally, the fully connected layer classifies the feature maps (Yandong et al., 2016).

The input layer of the convolutional neural network can deal with multidimensional data. In other words, it can directly accept the original image data and time or spectral samples. Therefore, the convolutional neural network is able to effectively learn the corresponding features from a large number of samples, eschewing a complex feature extraction process (Yamashita et al., 2018).

Figure 2.2 is a typical component in the convolutional network, which contains three levels. In most of the existing successful models, such as ResNet and GoogleNet, this basic structure is followed. During the convolution stage, the convolutional layer of the model computes multiple convolutions in parallel to generate a set of linear activation responses; at the detection stage, each linear activation response is processed using a nonlinear activation function (Albawi et al., 2017), such as rectification linear activation function (ReLU) (Zoumpourlis et al., 2017). Finally, the pooling layer adjusts the output of this layer (Nagi et al., 2011).

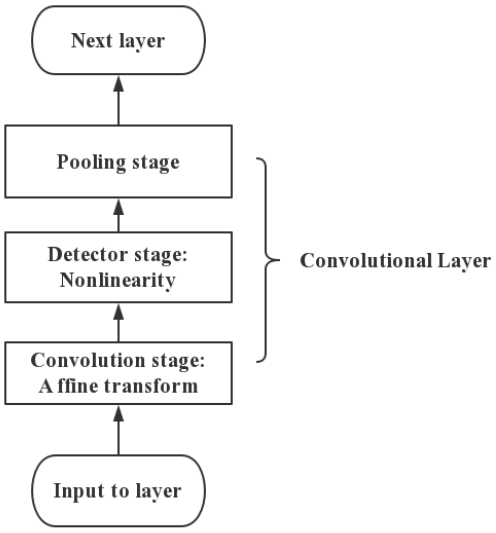


Figure 2.2 A typical layer in a convolutional network

The convolutional layer has played an role in computer vision, such as classification, detection, and segmentation. The models add a convolutional layer that ensures the network is to extract different features of the input. As the number of convolutional layers increases, the extracted features will become increasingly complex. The convolutional layer is comprised of the number of convolutional units, each of which is optimized by using a backpropagation algorithm (Dreyfus, 1990). The convolutional layer contains a plurality of convolution kernels consisting of weights and offset vectors. The size of the convolution kernel determines the size of the regions, in which each neuron in the convolutional layer is connected to a plurality of neurons in the adjacent layers. In the

process of convolution network, it will regularly multiply each input element by using the elements in the region and superimpose bias (Goodfellow, Bengio & Courville, 2016), the mathematical description of convolutional operations is

$$s(t) = \int x(a)\omega(t-a)da \tag{2.4}$$

where function $x(a)$ in this formula is usually called an input and the $w(t\text{-}a)$ is called a kernel function or a filter. The output $s(t)$ is called a feature map. For an image I, the input image and convolution kernel are assumed to be two-dimensional, the convolution operation is

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{2.5}$$

.

Generally, the size of the convolution kernel is substantially smaller than the size of the input view to achieve the characteristics of the sparse connectivity of the convolutional networks. This locally connects the characteristics and ensures that the learned filter has the most robust responses to local input features. Compared to the full connectivity, the sparse connectivity of convolutional networks has fewer parameters, this design dramatically reduces the cost in terms of time and computational complexity. Moreover, the parameter sharing characteristic of convolutional networks allows the network to learn a single set of parameters when conducting convolution operations because multiple functions adopt the same parameters in the same network. Therefore, convolution is vastly superior to traditional neural networks in storage requirements and statistical efficiency.

Moreover, in order to understand advanced features better, the convolutional layer utilizes activation functions (e.g., ReLU, Sigmoid, and hyperbolic tangent) in expressing sophisticated features. (Bengio, 1998) The representation (Goodfellow, Bengio & Courville, 2016) is

$$A^l_{i,j,k} = f(Z^l_{i,j,k}) \tag{2.6}$$

.

After feature extraction, the pooling layer performs feature selection and information filtering based on the input feature map. In order to reduce the input image information and leave only relevant information, the pooling function replaces the result of a single point in the feature map with the statistic for its neighbor region. Pooling can not only achieve downsampling but also achieve nonlinearity, expand the perception field and simplify network complexity. In applied pooling (e.g., average pooling, max pooling, and overlapping pooling), max pooling makes a significant contribution to image processing and video analytics. It achieves the role by selecting the maximum value of the image region as the combined value of this area. That is equivalent to retain the best match in this area.

Finally, the output layer of the convolutional network will export the results of object recognition or classification in the form of coordinates or pixels, sizes or classifications. For object detection, the output layer can output the object's centre coordinates, size, and classification. In image semantic segmentation, the output layer directly exports the classification result for each pixel (Younes et al., 2014).

Convolutional neural networks achieve convergence by iteratively performing feedforward and feedback operations (Zamir et al., 2017). The feedforward network extracts the high-level semantic information from the original data and abstracts it layer by layer through convolution, confluence and nonlinear activation of the layer stack. The error between the predicted value and the actual value is obtained by calculating the objective function. The feedback operation exploits a backpropagation algorithm (Rumelhart, Hinton & Williams, 1986) to minimize the error from the last layer to achieve the purpose of updating the parameters of each layer (Shrestha & Mahmood, 2019).

The weight sharing of the convolutional neural networks reduces the number of times of training, thereby decreases the computational complexity of the network. Meanwhile, in order to improve the generalization capability of the network, the local transformation has certain invariance through using the aggregation operation. Moreover, CNN as one of the models for deep learning has perfect end-to-end characteristics. CNN directly

inputs the original data into the network, the model has the ability to learn features autonomously during the training process. Therefore, the convolutional neural network saves computational resources and eschews inaccuracies and errors from training data.

2.4 Backpropagation Algorithm

The convolutional neural networks, like other deep learning, learn parameters by minimizing the value of the loss function. In order to optimize the extremely complex nonconvex function of convolutional neural networks, deep learning models typically exploit Stochastic Gradient Descent (SGD) and error backpropagation to update the parameters of the model (Newton, Pasupathy & Yousefian, 2018).

During the process of parameters updating, the gradient indicates the direction in which the error increases dramatically, the direction opposites the gradient direction should be taken as the one that minimizes the loss function. This process is iterated till the network response to the input data reaches a satisfactory target range. When the learning rate is η, the specific steps of backpropagation are (Xiushen, 2018)

Algorithm 2.1. Backpropagation Algorithm.

Input: Training set (x_n^l, y_n) with N training samples, n=1,...,N; epoch T

Output: w^i, i=1,...,L

1:for t=1,...T do

2: while $n \neq N$ do

3: Get x^i each layer by feedforward operation and calculating the final error z;

4: for i = L,...,1 do

5: (a) Calculate the derivative of the i-th layer error on the layer parameters $\frac{\partial z}{\partial \left(vec\left(w^i\right)^T\right)}$;

6: (b) Calculate the derivative of the i-th layer error on the input data of the layer

$$\frac{\partial z}{\partial\left(vec\left(x^i\right)^T\right)};$$

7: (c) update parameters: $w^i \leftarrow w^i - \eta \frac{\partial z}{\partial w^i}$;

8: end for

9: end while

10: end for

11: return w^i

2.5 Typical Convolution Model

2.5.1 AlexNet

In 2012, a deep convolutional neural network AlexNet was designed and won the 2012 ImageNet LSVRC champion. AlexNet is more significant than other convolutional networks, because not only it proves the efficacy of CNN in complex models, but also it makes use of GPU implementations to get through the training. Compared to CPU, GPU with thousands of cores for parallel computing is more likely to get results in an acceptable timeframe. AlexNet replaces traditional Sigmoid and Tanh (hyperbolic tangent) with ReLU (Rectified Linear Units) as its activation function for the first time. AlexNet replaces the traditional Sigmoid and Tanh with ReLU as its activation function for the first time.

In Figure 2.3, we see when the input is less than 0, the output is 0; when the input is greater than 0, the output is equal to the input. Compared with the nonlinear Sigmoid and Tanh function, ReLU is significantly reduced in both calculation and convergence speed, which is attributed to the fact that ReLU is linear and the derivative is always 1.0 (Krizhevsky, Sutskever, and Hinton, 2012). However, in order to suppress the infinity of ReLU response results, AlexNet takes advantage of activated neurons to suppress

adjacent neurons to achieve local inhibition. This method is called Local Response Normalization (LRN).

Denoted by $a^i_{x,y}$ the activity of a neuron computed by applying kernel i at position (x, y) and applied the ReLU nonlinearity, the response-normalized activity $b^i_{x,y}$ is given by using the expression, the formula is (Krizhevsky, Sutskever, and Hinton, 2012)

$$b^i_{x,y} = a^i_{x,y} \Big/ (k + \alpha \sum_{j=\max(0,\, i-n/2)}^{\min(N-1,\, i+n/2)} (a^j_{x,y})^2)^{\beta} \tag{2.7}$$

Moreover, AlexNet employs data augmentation and dropout to prevent overfitting during training. In other words, in the case of insufficient training data, AlexNet generates new data by performing a series of performatives on the original training dataset to enhance training data expeditiously. At the same time, dropout sets the neurons to 0 by using a defined probability in the neural network, suppressing its forward and backward propagation, while keeping the number of neurons in the input and output layers unchanged.

$$\text{ReLU}(x) = \begin{cases} x & x > 0 \\ 0 & x \leqq 0 \end{cases} \tag{2.8}$$

2.5.2 GoogLeNet

Since the birth of AlexNet in 2012, the ImageNet champions have been made with Convolutional Neural Networks, and the layers are getting deeper and deeper. Therefore, CNN became the core algorithm in the field of image recognition and classification. In 2014, GoogLeNet became the champion of the ImageNet Challenge (ILSVRC14). GoogLeNet has only 22 layers, but its performance is superior to AlexNet in many respects. This advantage owns to GoogLeNet having far fewer parameters than AlexNet, which guarantees better results with limited memory or computing resources. The

GoogLeNet team proposed the Inception network structure as the reason why it can show excellent results with fewer parameters.

The GoogLeNet team proposed the Inception network structure as it can engender meaningful results with fewer parameters. Inception emulates the characteristics of repeated accumulation of neurons of human brain. Moreover, the Inception network clusters sparse matrices into more dense sub-matrices. This operation further ameliorates the computational performance of the model in order to build a sparse and high performance network. In addition, Inception network structure refers to Network in Network (Min, Qiang & Shuicheng, 2013) to make the most of 1×1 convolution kernel for dimensionality reduction and network size limitation. The development of GoogLeNet has four stages from v1 to v4.

Inception v1 combines 1×1, 3×3, 5×5 convolution and 3×3 pooling operations, while expanding the network width, it also aggrandizes the adaptability of this model (Szegedy et al., 2015).

Inception v2 adds the BN layer based on Inception v1, which reduces the internal covariate shift to normalize the output of each layer to a $N(0, 1)$ Gaussian distribution. Moreover, it refers to VGG using two 3×3 convolution kernels to replace 5×5 in the inception module, which reduces the number of parameters and speeds up the calculation (Szegedy et al., 2015).

Subsequently, Szegedy et al. (2016) introduced Factorization Machine (FM) in Inception v3, which decomposed 7×7 into a 1×7 and a 7×1 convolution, 3×3 into 1×3 and 3×1 convolution. The purpose is to accelerate the calculation of the neural network training and increase the nonlinearity of the network.

In Inception v4, grid size of the Inception block was unified. Besides, the structure of ResNet (Szegedy, et al. 2016) can enormously assist the acceleration of network training and performance improvement.

2.5.3 VGGNet

VGG was proposed by the Visual Geometry Group of Oxford, which is a deeper convolutional neural network based on AlexNet. The main contribution is to demonstrate that the performance of the network will be increased as the network grows. VGG mainly adopts the method of increasing the convolution layer to deepen the network. It is found that as the number of network layers increases, not only the learning ability of the network is aggrandized, but also the classification capability is strengthened. When the depth is increased up to 16 layers, the recognition has been much improved, namely VGG-16 and VGG-19.

Compared to AlexNet, one advancement in VGG-16 is the replacement of larger convolution kernels in AlexNet with three consecutives 3×3 convolution kernels, this leads to deeper layers and fewer parameters than that of AlexNet, but it only needs a fewer number of iterations.

2.5.4 ResNet

ResNet (Deep residual network) was proposed in 2015 and won first place in the classification of the ImageNet competition. It mainly copes with the trouble that the accuracy of the training set decreases, which is caused by the gradient disappears as the network deepens. The core method of ResNet is to introduce an identity shortcut to skip one or more layers directly. The training error of deep networks is generally higher than that of shallow networks (He, Ren & Sun, 2016), but adding an identity mapping of multiple layers to a shallow network can obtain less training errors. This solution shows that the layers of the identity map are more suitable for training because residual learning requires less computations than others. Additionally, from a mathematical point of view, if the residual unit is expressed as

$$\begin{aligned} y_l &= h(x_l + F(x_l, W_l)) \\ x_{l+1} &= f(y_l) \end{aligned} \tag{2.9}$$

where x_l and x_{l+1} represent the input and output of the first residual unit, respectively; noted that each residual unit typically contains a structure of multiple layers. F is the residual function, representing the learned residual and h is the identity map, while f is the ReLU activation function. Thus, it can be derived that the learning characteristics from shallow (l) to deep (L) layers are

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i) \quad . \tag{2.10}$$

Using chain rules, the gradient of the reverse process can be found

$$\frac{\partial loss}{\partial x_l} = \frac{\partial loss}{\partial x_L} \cdot \frac{\partial x_L}{\partial x_l} = \frac{\partial loss}{\partial x_L} \cdot (1 + \frac{\partial}{\partial x_L} \sum_{i=l}^{L-1} F(x_i, W_i)) \tag{2.11}$$

The loss function of the first factor reaches the gradient of L, the parentheses indicates that the short-circuit mechanism can propagate the gradient without loss, and the other residual gradient needs to pass through the layer with weights, and the gradient is not directly transmitted.

2.6 Faster R-CNN

Faster R-CNN is an improved network based on Fast R-CNN and R-CNN. R-CNN and Fast R-CNN make use of ROI pooling layer after the convolutional layer, multitask loss function as a loss function facilitates the addition of bounding box regression directly to the CNN network. However, the method of generating the Region Proposal with Selective Search still takes more training time and test time. Faster R-CNN (Ren, et al, 2017) properly uses Region Proposal Network (RPN) in place of previous selective search to generate the region proposal and shares the convolution network with the object detection network. Accordingly, RPN decreases the number of proposals from approximately 2000 to 300. On top of that, the quality of proposal boxes has also improved in essence (Liu et al., 2017).

The Faster R-CNN working process is shown in Figure 2.5. When an entire image

is input to the Faster R-CNN for feature extraction, 300 proposals are generated in each image through the RPN. Subsequently, the proposals are mapped to the last layer of the convolutional feature map. Each RoI generates a fixed-size feature map through RoI pooling layer. Classification probability and bounding box regression are trained by softmax loss function and smooth L_1 loss function (Tang et al., 2018).

A sliding window of RPN is employed on the convolution feature map to generate a fully connected feature of length 256 or 512 dimensions. Moreover, Faster R-CNN also feeds these features into the classification and regression layers. The classification layer is used to discriminate that the proposal is foreground or background. The regression layer predicts the center position and width of the proposal. In other words, the position of the sliding window of the RPN provides general position information of the object, and the regression of the frame provides a more precise position of the frame (Ren et al., 2015).

Moreover, Faster R-CNN has two fully connected layers responsible for classification and regression, respectively. Similarly, it makes the most use of two loss functions for fine-tuning. For the classification task, if i is assumed to be the index of an anchor in a mini-batch; p_i is the probability that the algorithm output is that the object is the object; p_i^* is the ground-truth label of the anchor i; t_i is a four-element vector, which is bounded by the algorithm. The parameterized coordinates of the box t_i^* is the ground-truth box associated with a positive anchor, the representation is

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{2.12}$$

for the regression, the parameterizations of the four coordinates as following, where x, y denote the center coordinates of proposal box, respectively; w and h stand for the width and height of the proposal box. Variables x, $x_{a,}$ and x^* are for the predicted box, anchor box and ground-truth box, respectively. The bounding-box regression from an anchor box to a nearby ground-truth box (He, Girshick & Sun, 2017) is

$$t_x = (x - x_a)/w_a \qquad t_y = (y - y_a)/h_a \qquad t_w = \log(w/w_a) \qquad t_h = \log(h/h_a)$$

$$t_x^* = (x^* - x_a)/w_a \qquad t_y^* = (y^* - y_a)/h_a \qquad t_w^* = \log(w^*/w_a) \qquad t_h^* = \log(h^*/h_a) \tag{2.13}$$

2.7 SegNet

SegNet is an image segmentation model developed by the University of Cambridge that divides the area of objects in an image into pixels (e.g., cars, roads, pedestrians, etc.). SegNet consists of an encoder-decoder network with the VGG-16 convolutional layer as five encoders consisting of 13 convolutional layers and removing the fully connected layers (Simonyan & Zisserman, 2014). The decoder performs nonlinear upsampling of the input feature map using the maximum pool index accepted from the corresponding encoder (Badrinarayanan, Kendall & Cipolla, 2017). More specifically, the first decoder corresponds to the last encoder, the second decoder corresponds to the penultimate encoder, and so forth. The output of the decoder is fed into a multistage softmax classifier, which conducts classification by using pixels (Mittal, Hooda, & Sofat, 2018).

The encoder in SegNet increases the resolution of the feature map to the input size in a step-by-step manner, avoiding reduction of the spatial resolution of the feature map by applying batch normalization and ReLU operations on the feature map. The resultant information is lost. Moreover, SegNet acquires and stores the boundary information in the encoder map before upsampling. Therefore, due to translation invariance, maximum merging and upsampling can be applied for classification tasks for better robustness without compromising the accuracy of image segmentation.

Compared to several other models for segmentation, SegNet is much efficient because it only stores max pooling indices and uses the indices in the decoder network to obtain satisfactory performance.

For example, DeconvNet and its semi-supervised variant decoupled platform exploit nonlinear upsampling in the decoder network with the maximum position of the encoder feature map. These architectures are independent on SegNet. However, the encoder network includes a fully connected layer that occupies approximately 90% parameters of

the entire VGG-16 network. It makes the network training tremendously troublesome, so more auxiliary tools are needed, such as using zone proposals. In addition, this also significantly increases the time of reasoning (Noh, Hong & Han, 2015).

Instead of using pooled location index information, U-Net transmits the entire signature to the corresponding decoder, often at the expense of more memory (Ronneberger, Fischer & Brox, 2019). Besides, FCN disregards high-resolution feature maps for end-to-end, increases the possibility of the loss of edge information. In addition to the training-related issues, the way in which the encoder profile is multiplexed in the FCN, makes it significantly memory-intensive during test (Shelhamer, Long, & Darrell, 2017).

In general, since the primary motivation of SegNet is the application of scenario understanding, it is designed to ensure the efficiency of memory and computation time during prediction. Moreover, quantitative assessments show that SegNet is more efficient in terms of time and memory usage than other architectures.

2.8 Object Detection

Vehicle detection is a branch under object detection, the label of detection is the vehicle. In the process of detecting the target, the model not only needs to accurately find the position of the target object in the provided images but also controls the ROI boxes in appropriate sizes. Therefore, the main issues to be settled by using object detection in traffic scene understanding are where the object is and what the most suitable size of feature box is. However, solving these problems poses considerable challenges in actual operations. For example, weather and lighting conditions vary, the area in which an object appears is uncertain because there are multiple categories of objects. So far, thousands of researchers have overcome the above difficulties, research results are extremely praiseworthy.

Among them, many projects using traditional methods have achieved satisfactory performance in scene understanding. However, the traditional machine learning method based on the strategy of sliding window selection is not targeted. Not only does it have a high degree of time complexity, but also its features such as redundant windows and manual design have a negative impact on diversity changes. Moreover, the traditional object detection method is not universal. Therefore, different algorithms need to be customized for different objects. Compared with traditional machine learning, deep learning algorithms are much versatile for scene understanding because the acquired features are transferable. Similarly, engineering development, optimization, and maintenance costs are low. Therefore, deep learning has become the mainstream for object detection.

In the network models based on deep learning, accurate detection of targets of different sizes or small-scale targets is one of the main challenges. Even though the detection efficacy of Faster R-CNN is more accurate than R-CNN in the traditional sense, in the case where most of the vehicles in the traffic flow belong to large vehicles, the RPN is more inclined to adjust the bounding box of large vehicles. The MV-RCNN amalgamates two different detectors of the RPN and R-CNN structures as an appropriate solution for missing targets problem in projects related to multi-scale objects. The two detectors accommodate different sizes of vehicles with different parameters. The detector RPNS predicts smaller scale vehicles and the detector RPN-L predicts larger scale vehicles (Geng et al., 2018). Object detection based on Faster R-CNN in infrared images also uses two RPNs to detect large-scale targets and small-scale targets, respectively. At the same time, multiscale aggregation modules have been introduced into the main network to analyze the response of the model to different scale objects. In addition, the first four modules and ROI PSalign pool layer are exploited to construe more lavish object features at sensitive locations (Hao et al., 2019).

There is also a class of methods that capture the characteristics of the advanced feature map of deep learning model based on the small target detection regression and

cause the small target to be missed. Detection performance can also be ameliorated by increasing the feature response of different regions. A multiview object detection approach based on deep learning emulates the imaging magnification process observed by human eyes as it gradually approaches an object. Applying a region segmentation operation before object detection, each region of the image can be regarded as separated one. The view is magnified to generate more responses to feature map of the model (Tang et al., 2019).

For the sake of an improved goal of distinguishing multiple targets, in addition to the method of region segmentation, adjusting the offset is also an effective measure. A small object detection method based on SSD in a complex environment increases the density of the default box in horizontal direction by setting the offset, so as to efficiently eliminate the influence of the missing matching default box and make it easier to distinguish multiple targets. Secondary, this kind of models replace the standard 3×3 volume with a 5×1 convolution kernel to better detect smaller targets (Zhang et al., 2018).

In addition to deal with the difficulty of detecting too large and too small targets, the deformation issue of the target object has also been designed very well in the case of deep learning. For example, a modified Faster R-CNN model is used to detect images in a 360° camera. Even though a 360° camera can capture the entire scene in its entirety, the distorted images it captured often cause the object could not be accurately identified, which is a difficulty in object detection. To handle the distortion problem, data enhancement processing on the distorted data was designed to train various distorted data (Wang et al., 2019). Furthermore, the introduction of a multicore layer to apply different sized cores on different regions eliminates distortions. The acquisition of location information corresponding to the different distortions in the image by using object location in the 360° image also aids to identify the distorted vehicle images accurately.

In order to ensure the usability of the vehicle detection module in practical applications such as automatic driving, module inspection task must have better real-time

performance. Compared with convolutional networks for object detection, Faster R-CNN, which eliminates the selective region extraction method, is much capable for real-time applications. Some studies are based on the widespread use of faster R-CNNs and ZF-Net and VGG16 as the main pillars to improve inference time without loss of precision so as to achieve real-time target detection. The model not only has excellent real-time performance, but also amalgamates the auxiliary multifunction cascade area and the reduced area into a multifunctional auxiliary area so as to more accurately find the region of interest (ROI) and compensate for the loss of precision due to trimming (Shih et al., 2019).

Even though various sophisticated methods are adopted to achieve high accuracy and high real-time performance of the model, the prerequisite for achieving high performance by these methods still has sufficient training datasets. For the datasets with complex marking processes, it is challenging to make thousands or even tens of thousands of tagged images in a short period. A difficulty is tackled by modifying Fast R-CNN model for vehicle detection, conducting supervised pretraining of large auxiliary datasets, and fine-tuning small datasets to deal with the problem of highly descriptive training data (Hsu, Huang & Chuang, 2018).

2.9 Semantic Segmentation

Object detection can help us draw the bounds of entities, but human understanding of the scenes can detect each entity at a pixel-level and mark the exact boundaries. With the advent of autonomous vehicles, an in-depth understanding of the surrounding environment has become one of its main issues, so it is increasingly vital to segment entities accurately. As deep learning continues to heat up. In practice, it is proved that deep convolution neural networks have significant advantages in extracting image features. Therefore, in order to improve the efficacy of semantic segmentation algorithm, the application of convolutional neural networks to semantic image segmentation is a hot research topic (Chen et al., 2018).

Nonetheless, according to the results of previous work, image segmentation at the boundary of two objects is far less than the region from the boundary. So far, all the methods engender the best performance using pixel-level labels, which require expensive workforce and high time costs. For solving this problem, one of the solutions obtains more accurate segmentation results near the edge of the object through accurate contour detection. The main idea is to make good of object detection to classify region proposals and then apply saliency detection methods to classify these classification proposals (Shuhan et al., 2017). Even though this model has significantly improved accuracy, it is not an end-to-end framework. A CRF-RNN model takes in the advantages of CNN and CRF-based graphical models in a unified framework that passes loss reduction from its output to input during backpropagation-based training while learning CRF parameters to enable end-to-end training (Zheng et al., 2015). The same is the combination of CRF and neural network. The DCNN-CRF model improves the location of object boundaries by amalgamating DCNN and probability map models. This approach overcomes the effect of the combination of the max pooling and downsampling on object boundary location by combining the response of the final DCNN layer with a fully connected conditional random field (CRF) (Papandreou, 2018).

Another model D-RefineNet, which is a depth-assisted RefineNet, proposes a depth-assisted loss function that takes advantage of the depth value to sharply change at the boundary of the object to constrain the edge segmentation so as to refine the efficacy of edge segmentation. Moreover, the proposed network is capable of predicting classification results using only RGB images. Depth images are only employed for loss functions without increasing model size (Chang, Guo & Ji, 2018).

Regarding most algorithms that adopt object proposing, there are often challenges such as severe occlusion, object shape changes, and lack of category information. General approaches state these difficulties by using bottom-up regional grouping or segmentation. It exploits a similarity measure to determine whether two adjacent regions should be merged. Moreover, this merging algorithm performs an inference process of grouping two

regions into super regions so as to generate object proposal. Therefore, if the model determines that the two regions belong to the same class of objects, they assign a higher similarity score to adjacent regions and recursively merge the adjacent regions with the highest score. However, these methods often require careful adjustments or settings that limit their performance in complex environments.

A hierarchical region grouping method is applied to generate a segmentation proposal by learning a recurrent neural network (ReNN). The cross region and similarity measure -based learning process are included in the bottom-up region merge process for end-to-end training. At the same time, a structural loss function is defined, which compensates for incorrect merging candidates by measuring the similarity and objectivity of adjacent regions, optimizes cross-region similarity and object prediction in recursive iterations (Pinheiro, Collobert & Dollár, 2015).

On the other hand, in addition to cope with the segmentation problem of object edges, segmenting complex scenes is an issue that needs exigent attention in semantic segmentation. For example, in an urban environment, traffic conditions can be very complicated, the behavior of dynamic traffic participants is unpredictable, which makes them more challenging than highways or country roads.

In order to handle this issue, a breakthrough method of obtaining datasets was taken by using traditional monitoring devices (Deng et al. 2017). Instead, the fisheye camera was utilized to obtain a 180° front hemisphere view to cover the field of view. In order to deal with complex scenes in fisheye images, an Overlapping Pyramid Pool (OPP) module is proposed to explore local, global, and pyramid local region context information so as to increase the training by using additional data obtained from existing sources through changing the focal length of a fisheye camera. The dataset enhancement to improve the generalization performance of the network was under consideration.

Inspired by skip connection, SharpMask model includes a top-down refinement approach to increase the feedforward network for object segmentation, enabling lower

layers in the convolutional network to capture rich spatial information. At the same time, the upper layer encodes object-level knowledge but does not change factors such as posture and appearance. Although this method also utilizes the functions of all layers of the network, it does not output independent predictions at each layer. Instead, it first outputs a rough "mask code" in the feedforward transfer, and then uses the features of successive lower layers. This mask encoding is optimized in the top-down pass (Pinheiro et al., 2019).

However, because CNN loses image details during the convolution and pooling process, the size of feature map becomes smaller and smaller, which makes it impossible to point out the specific contour of the object and indicate which object each pixel belongs to, thus the object cannot be accurately segmented. Therefore, AlexNet (Krizhevsky, Sutskever & Hinton, 2012), VGG network (Simonyan & Zisserman, 2014) and Goog LeNet (Szegedy et al., 2014) are merged into a complete convolutional network and transferred the learned knowledge (Jonathan, et al., 2015), it represents the fine-tuning and split task (Donahue et al., 2014), it creates a skip structure that incorporates the semantic information of the deep coarse layer with the appearance information of the shallow fine layer to generate accurate segmentation.

Despite this, FCN-based semantic segmentation networks have predefined fixed-size perceptual fields. Therefore, objects that are substantially larger or smaller than the receptive field may be fragmented or mislabeled. An instance-wise prediction deals with object-scale changes by eliminating the limitations of a fixed-size receptive field in the FCN (Hyeonwoo, et al, 2015).

In the FCN network, a rough segmentation map is generated by using upper convolutional layer and jump connections, more jump connections are introduced in order to improve the efficacy. However, the extensive FCN network only duplicates the encoder, while the SegNet network replicates the max pooling index. It makes SegNet more efficient than FCN in memory usage.

U-Net is another semantic segmentation architecture that exploits an encoder-decoder layer like SegNet. The reason is in consequence of its U-shaped structure, in which the first few layers perform downsampling, and the latter layer performs upsampling. On the other hand, opposite from SegNet, U-Net never utilizes pooled indices and transfers the entire feature map to the corresponding decoding layer. The transmitted feature map is coupled to the upsampled feature map so as to generate a resulting feature map. Very few images can be used to train the network, in particular for the field of biomedical image segmentation where annotation data is expensive. The project for hand bone segmentation employs a lightweight U-Net architecture. Moreover, this model adopts multiscale convolutional operations. Multiple filters with different core sizes are applied in the model to resist hand-scale changes during child growth (Ding et al., 2019).

Compared to 2D input, 3D point cloud descriptors need to overcome more challenges. A semantic segmentation problem for large-scale 3D scenes is addressed by merging 2D images and 3D point clouds together. The model acquires 2D images and 3D point clouds simultaneously. Then, taken the two-dimensional image as input, the segmentation result is obtained through the training of the convolutional neural network. The coordinates of the obtained two-dimensional image are transformed into a three-dimensional point cloud (Zhang et al., 2019).

2.10 Data Augmentation

The main feature of the most advanced deep learning architecture is that there are thousands of trainable parameters so that these parameters are worked correctly and require a huge amount of data for training. However, the actual training data is not as much as expected. Therefore, optimizing the performance of this extremely complex model is challenging. One of the main challenges is that the number of training samples is insufficient, and the model is overfitting to the training dataset (Srivastava et al., 2014). For image analysis, data expansion can be an effective solution that can flip, translate or

rotate existing images to create more data and make neural networks better generalized (Stivaktakis, Tsagkatakis & Tsakalides, 2019). The application of data expansion can also improve the robustness of the model.

Among them, offline and online augmentations are two categories of data enrichment, which takes into consideration of both the expansion for small and large datasets. The online augmentation method includes rotations, translations, flipping, etc. after obtained the batch data. Linear enhancement methods are often used for large datasets, deep learning frameworks already support this data enhancement approach and can be optimized using GPU. For example, the multilabel land cover scene categorization project adopts an online method for data augmentation, where each training batch is dynamically enhanced at each round of iteration. Because CNN rarelies or never processes the same example twice, dynamic expansion eliminates the memory requirements associated with more massive static datasets and enhances the generalization capabilities of the network compared to offline alternatives (Stivaktakis, Tsagkatakis & Tsakalides, 2019).

Offline augmentation is typically employed for small datasets, which increases the size of the data by using a factor which equals to the number of conversions performed (Fawzi et al., 2016). For example, it can increase the number of datasets by flipping all the images. In the project of image segmentation for blood cells, in order to increase the size of the dataset, the original image is augmented by using random reflection, and the random translation along x-axis or y-axis direction. After the training dataset is augmented, the training image is increased by using a factor of five (Tran et al., 2018).

A segmentation method based on SegNet for accurately segmenting glandular structures in histological images of colon cancer, selecting random cropping to obtain a large number of training images means using a random method to cut the image (Tang, Li & Xu, 2018). An approach of deep learning-based decision fusion for action or gesture recognition in order to perform augmentation on depth data, additional training data is generated by adding rotations if the images were captured from different directions or

viewpoints of the depth camera (Dawar, Ostadabbas & Kehtarnavaz, 2019). This method increases the size of the dataset 102 times. However, for rotation-based data augmentation, the dimension of the images may not be preserved after the original image is rotated. At the same time, in scaling-based data augmentation, because image reduction reduces the image size, the model has to make assumptions outside the image boundaries. Therefore, increased data by using rotation and scaling may reduce the performance of convolutional neural networks (Peixinho et al., 2018).

In many cases, multiple transformation methods are amalgamated together to achieve more comprehensive expansion. A Joint Data Enhancement (JDA) scheme effectively takes in from a multiaxis weighted fusion algorithm, a background noise fusion algorithm, and a random clipping algorithm. Therefore, JDA not only makes it easier to construct samples that fit the actual driving environment but also deals with the problem of insufficient samples (Zhang et al., 2019).

An Image Transformation Pursuit algorithm (Paulin et al., 2014) takes advantage of a greedy strategy to select the best transform from a set of candidate transforms. While getting impressive results, the strategy must not only hardly set the parameters of the candidate transform but also involve several steps in the retraining of the classifier. Therefore, when the number of candidate transforms is large, it is highly probable that an extremely high time cost is incurred. In addition, the classifier has additional resistance to noise (Szegedy et al., 2014) and geometric transformation (Fawzi & Frossard, 2015) for various disturbances. Therefore, improving the model from the viewpoint of geometry is also a data enhancement method (Fawzi et al., 2016).

Chapter 3
Methodology

This chapter mainly describes the research methods of scene understanding by describing the details of image segmentation and vehicle detection. Moreover, this chapter also details the evaluation methods applied in this thesis.

3.1 Vehicle-related Scene Understanding

For the understanding of vehicle-related scene, a high-performance model is not only to classify and detect isolated single objects, but also to understand advanced semantic information in the scene. Therefore, we make full use of deep learning to simulate the characteristics of the human brain. The high-level semantics in complex traffic scenes are employed through layer-by-layer processing and a stepwise abstraction of the feature map. If we use a hierarchical system to emulate human cognition of the scene, the entire human cognitive process of information needs to be carried out through several different levels of cognition. At a lower level, humans can extract tactile, visual, and auditory information as the basic features, which is similar to the idea of extracting concepts from the scene and forming the basic layer of the ontology; at a higher level, human brain unifies these basic features and judges between these features. At the highest level, our human brain can obtain the implicit semantics of the information through multiple reasoning, which is similar to the semantic expression of the semantic layer to the event.

Figure 3.1 is a three-layer semantic expression from bottom to up in the understanding of traffic scenes using deep learning. The basic layer expresses all the basic concepts and related information in the traffic scene. The interaction layer describes all the interactions in traffic scene. The semantic layer is the semantic expression of the event in the traffic scene (Yun & Kai, 2015).

Therefore, in vehicle-related scene understanding, the model understanding of the interaction layer and semantic layer in the scene can effectively improve the accuracy and logic of the prediction. In the model, the positional relationship between objects in the scene is very helpful for understanding the high-level semantics. It is also one of the necessary steps to use deep learning to analyse traffic scenes. The end-to-end nature of deep learning allows the model to handle multiple tasks. The deep learning model can complete multiple progressive tasks, the results of the previous task are used as an aid to the latter task. In order to explore the vehicle-related scenes more deeply, we conduct the

positional relationship exposition of the objects in the scene in semantic segmentation.

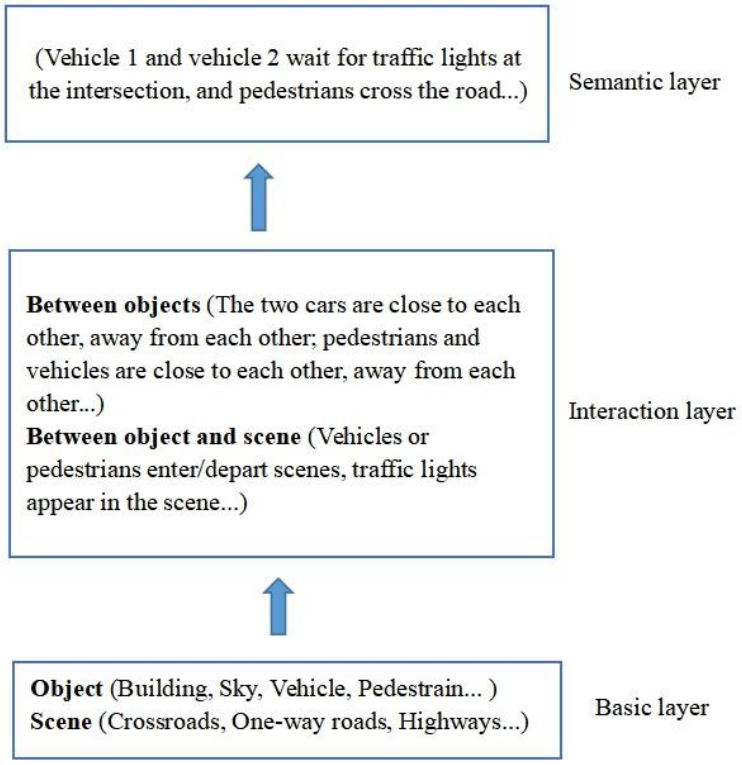


Figure 3.1 The layers of traffic scene understanding

Figure 3.2 utilizes logical relationships to correlate the classes in the scene understanding. According to prior knowledge, objects such as trees and buildings should normally appear on both sides of the road. Bus lane signs are usually drawn on the road. Most vehicles only travel on the road and do not appear on trees or in the sky. The sky is always above all objects and is the fact that it never changes.

The object relationship in the topological map plays a decisive role in scene understanding base on deep learning. According to the topological relationship between objects in the scene, the model can be used to clarify the objective logic in the traffic scene. Deep learning neural network, as a typical ANN model, can achieve more human-like cognition by learning the logical relationship between objects. Moreover, the object relationship topology map constrains the distribution range of the output results, reduces

the output of the unrealistic logic, thereby improves the accuracy of the output results.

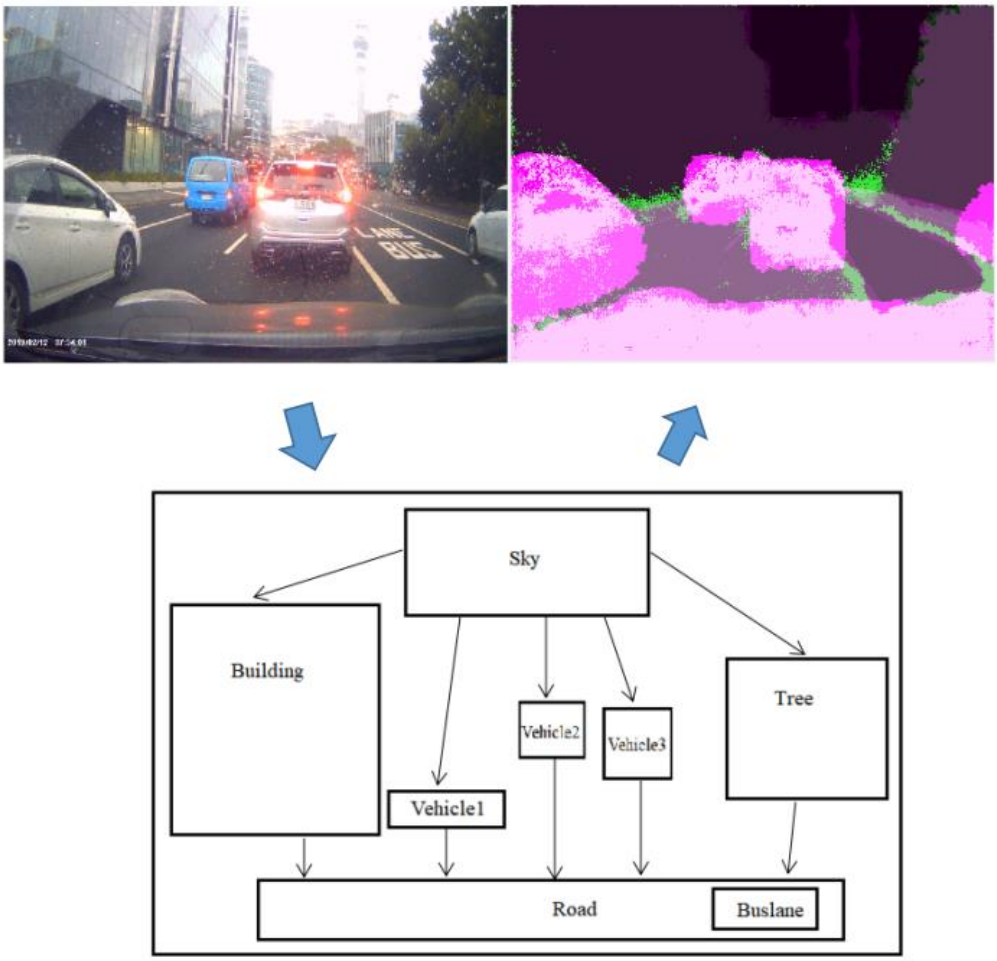


Figure 3.2 The topological relationships in the scene

3.2 Research Design

In order to deeply discover the contribution of deep learning in traffic scene understanding, we provide a detailed experimental method based on deep learning for semantic segmentation and vehicle detection tasks.

3.2.1 Collection and Processing of Data

This thesis exploits the 12-megapixel wide-angle camera of the iPhone 8 mobile to capture real streets and highway traffic scenes in Auckland. In order to obtain a clearer view, the camera is fixed at the middle front of a vehicle, its video recording mode is

switched on during driving time. The Auckland traffic scene around the vehicle is stored as .mp4 files.

During the data processing phase, .mp4 video files recorded in the Auckland traffic scene were converted to image files in .png format. This thesis applies MATLAB as a programming tool to split the video files into thousands of frames and store them on a local hard disk. Moreover, in order to improve the quality of the image data, we carefully select the most clear and distinct images from these files stored in the computer for labelling.

3.2.2 Research Design for Image Segmentation

The core problem of scene understanding is more and more indispensable taken account of increasing applications in realistic demands or relevant knowledge from the images. Therefore, semantic segmentation has become one of the pivots in vehicle-related scene understanding. It is also a high-level task that paves the way for scene understanding. So as to certainly describe the experimental procedure, this thesis provides a detailed flowchart for the experimental steps. The workflow for performing semantic segmentation has four steps shown in Figure 3.3.

The first step is to label data or obtain labeled data. In order to make the model learn rich features and achieve better robustness and accuracy, it is imperative to collect images of the relevant environment extensively. Consequently, the dataset plays a decisive role in the quality of a model. There are two basic ways to acquire datasets for training neural networks. One is to download the required labelled images on the Internet. Nowadays, there is plenty of complete datasets related to traffic scenes in the field of image processing for scholars, such as CamVid of Cambridge University, Caltech Dataset of California Institute of Technology and BDD100K of Berkeley. However, automated driving technology is not mature enough to understand scenes and objects in complex traffic environments due to the discrepancies in traffic rules and transportation facilities in various countries. No algorithm can be applied for all traffic scenes.

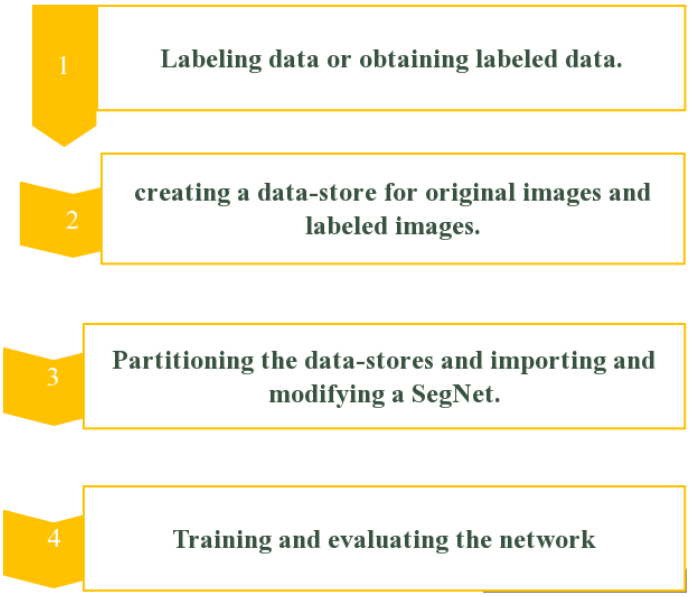


Figure 3.3 The flowchart of image segmentation

Therefore, this project focuses on the understanding of Auckland's traffic scenes. Our dataset provides a large number of Auckland highways as well as street vehicles, traffic signs, buildings, and other road information. We use an in-vehicle camera to obtain video files of Auckland traffic scenes. In order to facilitate the labeling, the video is split into 91 images by acquiring one frame every 10 seconds and using MATLAB to define pixel-level labels for project object classification, including sky, building, buslane, road, lane, tree, trafficsign, turnsign, vehicle, pedestran, and trafficlight as shown in Figure 3.4. In addition, for the sake of adaptivity to the image size requirements of the input layer of the SegNets, all images in the dataset are resized to 360×480. In order to enable the dataset and train an effective semantic segmentation network, we ensure the following points when creating the dataset：

- This dataset guarantees the sharpness of images, there is no image blur or distortion generated by motions.
- The dataset contains classification objects taken from various angles and locations. The same object does not appear in the same position in different images.

- The same classification object has different distances from the camera; i.e., the same classification objects in different images have different sizes.

However, when processing large amounts of data, it is usually not possible to load all of this information into memory. In order to manage large datasets, *ImageDatastore* and *PixelLabelDatastore* are utilized in this project to store raw data and labelled data, respectively. The data store contains the location of the used files and can only be read into memory when the file is manipulated.

After created SegNet, it is required to divide the dataset into two parts: the first part is the training dataset for training SegNet, the other is the testset, which is exploited to evaluate the accuracy of the network. At the same time, refining the network based on the training set and test set could ensure that the model is at the best performance.

Figure 3.4 Labelled video frames

Ideally, our dataset is required to have an equal number of pixels (resolution) across all categories. In contrast, in realistic highway and street scenes, the pixels and number of each class are unbalanced. Because most areas of the image are sky, buildings, and roads, objects such as road traffic sig and pedestrians are small, sky, buildings, and roads

in these scenes have more pixels than pedestrians, vehicles, and traffic signs (see Figure 3.5).

This project adjusts the class weights based on PixelCount, which is the number of pixels in class. The *ImagePixelCount* (*iPixelcount*) is the total number of pixels in images that have instances of classes. If *Pixelcount* and *iPixelcount*, shown in Figure 3.6, represent the number of pixels in each class and the total number of pixels, *CF* represents the class frequency. *CF* can be expressed as

$$CF=Pixelcount/iPixelcount \tag{3.1}$$

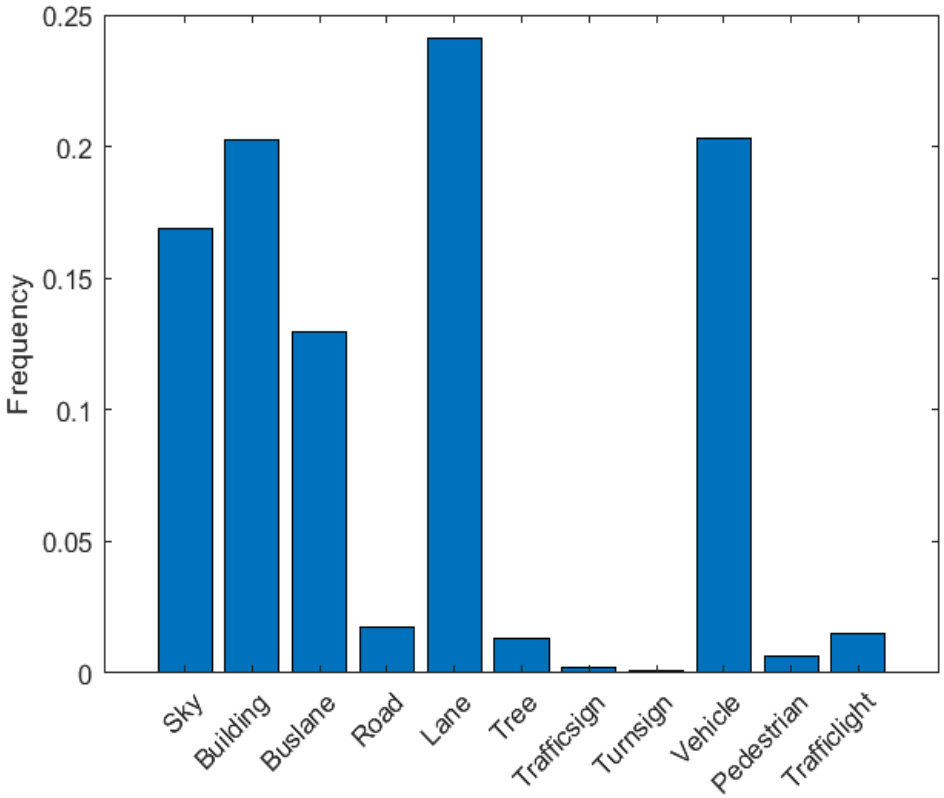


Figure 3.5 Frequency of the objects

Name	PixelCount	ImagePixelCount
'Sky'	1.2725e+07	7.465e+07
'Building'	1.5254e+07	7.6723e+07
'Buslane'	9.7786e+06	7.2576e+07
'Road'	1.3303e+06	2.6957e+07
'Lane'	1.8165e+07	7.6723e+07
'Tree'	9.9524e+05	7.0502e+07
'Trafficsign'	1.4261e+05	2.281e+07
'Turnsign'	66177	2.4883e+07
'Vehicle'	1.5341e+07	7.465e+07
'Pedestrian'	4.9813e+05	3.9398e+07
'Trafficlight'	1.1343e+06	5.3914e+07

Figure 3.6 *Pixelcount* and *IPixelcount* acquired

at the same time, if the class frequency array is represented as $X_1,.....,X_n$, it is organized in the order $X_{(1)},.....,X_{(n)}$. When n is an odd number, the median can be expressed as

$$Median_{odd} = X_{(n+1)/2} \tag{3.2}$$

When n is an even number, the median can be expressed

$$Median_{even} = \frac{X_{n/2} + X_{n/2+1}}{2} \tag{3.3}$$

In summary, the median frequency class weights

$$Classweight = Median(CF)/CF\,. \tag{3.4}$$

Furthermore, this project takes much time to label images manually. Due to the short period, only 91 images were included in the dataset. For the purpose of eschewing overfitting, we are usually required to geometrically transform the original image data, change the position of the image pixels and ensure that the features are unchanged to get

enough data. Consequently, data augmentation was used during training to expand the size of the training set. Through the method of data augmentation, the network can learn more features to improve the performance of the network. This project reflects each image at a 50% probability level. The translation is performed in units of pixels from the horizontal and vertical directions, the distance of translation is randomly selected from the continuous uniform distribution within the interval [-10, 10].

The structure of our neural network utilizes a combination of encoders and decoders to produce feature maps of images. The encoder of SegNet includes convolutional layer, batch normalization, ReLU activation, and max pooling. Max pooling is suitable for reducing the size of feature maps. Even though the object boundary in the image may be blurred during the operation of max pooling, the pooling is indeed the best way to reduce the size of the feature maps. For reducing the feature map size while retaining the complete boundary information, SegNet extracts the boundary information in the feature map before performing the downsampling. During the decoding process, the upsampling operation of the decoder preserves the size of the original input. The max pooling memory index stored in each encoder map is used for upsampling the feature map. The last decoder is connected to the Softmax classifier so as to classify each pixel in the image (Tran et al., 2018). This project created a SegNet network whose weight was initialized from the VGG-19 network. The additional layer required for semantic segmentation replaces the last pooling layer. The network structure is shown in Figure 3.7.

In addition, for acquired improved training results, it is decisive to adjust the parameters to suit for the corresponding dataset and network structure, setting hyperparameters for the network and training the network to acquire the best significant accuracy. The hyperparameters we utilize are for Figure 3.8 and Figure 3.9.

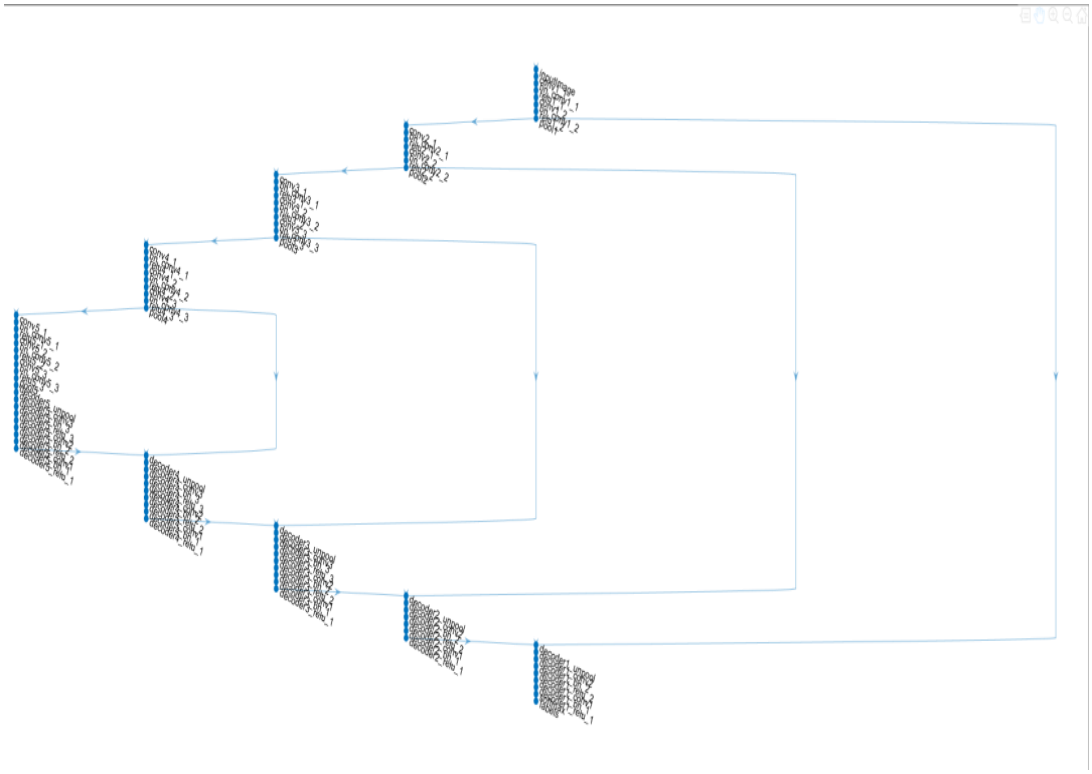


Figure 3.7 The architecture of our neural network

We make use of VGG19 as the encoder of the proposed model and the weight of the corresponding VGG19 as the initial weight of the proposed model. This model sets the input size of the input layer to 360 × 480 so as to ensure better learning of features in the dataset. In addition, the convolutional layer of the entire network proper uses a small size convolution kernel of 3×3 and a stride of one.

Compared to large convolution kernels, small convolution kernels stack more nonlinear layers, increase network depth to ensure more complex modes and lower costs. To ensure that the model can fully utilize and process the edge information of the input image and match other network parameters to keep the output and input size the same, we set the padding value of the convolution layer to 1. At the same time, the max pooling of 2×2 block and the stride of 2 are used, so that the width and height of each pooling layer are half of that of the previous layer.

109x1 Layer array with layers:

1	'inputImage'	Image Input	360x480x3 images with 'zerocenter' normalization
2	'conv1_1'	Convolution	64 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]
3	'bn_conv1_1'	Batch Normalization	Batch normalization with 64 channels
4	'relu1_1'	ReLU	ReLU
5	'conv1_2'	Convolution	64 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]
6	'bn_conv1_2'	Batch Normalization	Batch normalization with 64 channels
7	'relu1_2'	ReLU	ReLU
8	'pool1'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
9	'conv2_1'	Convolution	128 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]
10	'bn_conv2_1'	Batch Normalization	Batch normalization with 128 channels
11	'relu2_1'	ReLU	ReLU
12	'conv2_2'	Convolution	128 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]
13	'bn_conv2_2'	Batch Normalization	Batch normalization with 128 channels
14	'relu2_2'	ReLU	ReLU
15	'pool2'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
16	'conv3_1'	Convolution	256 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]
17	'bn_conv3_1'	Batch Normalization	Batch normalization with 256 channels
18	'relu3_1'	ReLU	ReLU
19	'conv3_2'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]
20	'bn_conv3_2'	Batch Normalization	Batch normalization with 256 channels
21	'relu3_2'	ReLU	ReLU
22	'conv3_3'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]
23	'bn_conv3_3'	Batch Normalization	Batch normalization with 256 channels
24	'relu3_3'	ReLU	ReLU
25	'conv3_4'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]
26	'bn_conv3_4'	Batch Normalization	Batch normalization with 256 channels
27	'relu3_4'	ReLU	ReLU
28	'pool3'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
29	'conv4_1'	Convolution	512 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]
30	'bn_conv4_1'	Batch Normalization	Batch normalization with 512 channels
31	'relu4_1'	ReLU	ReLU
32	'conv4_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
33	'bn_conv4_2'	Batch Normalization	Batch normalization with 512 channels
34	'relu4_2'	ReLU	ReLU
35	'conv4_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
36	'bn_conv4_3'	Batch Normalization	Batch normalization with 512 channels
37	'relu4_3'	ReLU	ReLU
38	'conv4_4'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
39	'bn_conv4_4'	Batch Normalization	Batch normalization with 512 channels
40	'relu4_4'	ReLU	ReLU
41	'pool4'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
42	'conv5_1'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
43	'bn_conv5_1'	Batch Normalization	Batch normalization with 512 channels
44	'relu5_1'	ReLU	ReLU
45	'conv5_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
46	'bn_conv5_2'	Batch Normalization	Batch normalization with 512 channels
47	'relu5_2'	ReLU	ReLU
48	'conv5_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
49	'bn_conv5_3'	Batch Normalization	Batch normalization with 512 channels
50	'relu5_3'	ReLU	ReLU
51	'conv5_4'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
52	'bn_conv5_4'	Batch Normalization	Batch normalization with 512 channels
53	'relu5_4'	ReLU	ReLU
54	'pool5'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
55	'decoder5_unpool'	Max Unpooling	Max Unpooling
56	'decoder5_conv4'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
57	'decoder5_bn_4'	Batch Normalization	Batch normalization with 512 channels
58	'decoder5_relu_4'	ReLU	ReLU
59	'decoder5_conv3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
60	'decoder5_bn_3'	Batch Normalization	Batch normalization with 512 channels
61	'decoder5_relu_3'	ReLU	ReLU

Figure 3.8 Parameters of Semantic Segmentation Model (1)

3.2.3 Research Design for Vehicle detection

Similar to the steps of image segmentation, the workflow for performing vehicle detection also follows four steps as shown in Figure 3.10.

To achieve vehicle detection in the Auckland traffic scene, the first step is to obtain the original Auckland traffic scene video through automobile data recorder and split the video into 337 single frame images, see Figure 3.11. At the same time, the image size is converted to 128×228 depending on the image size requirements specified by the input layer.

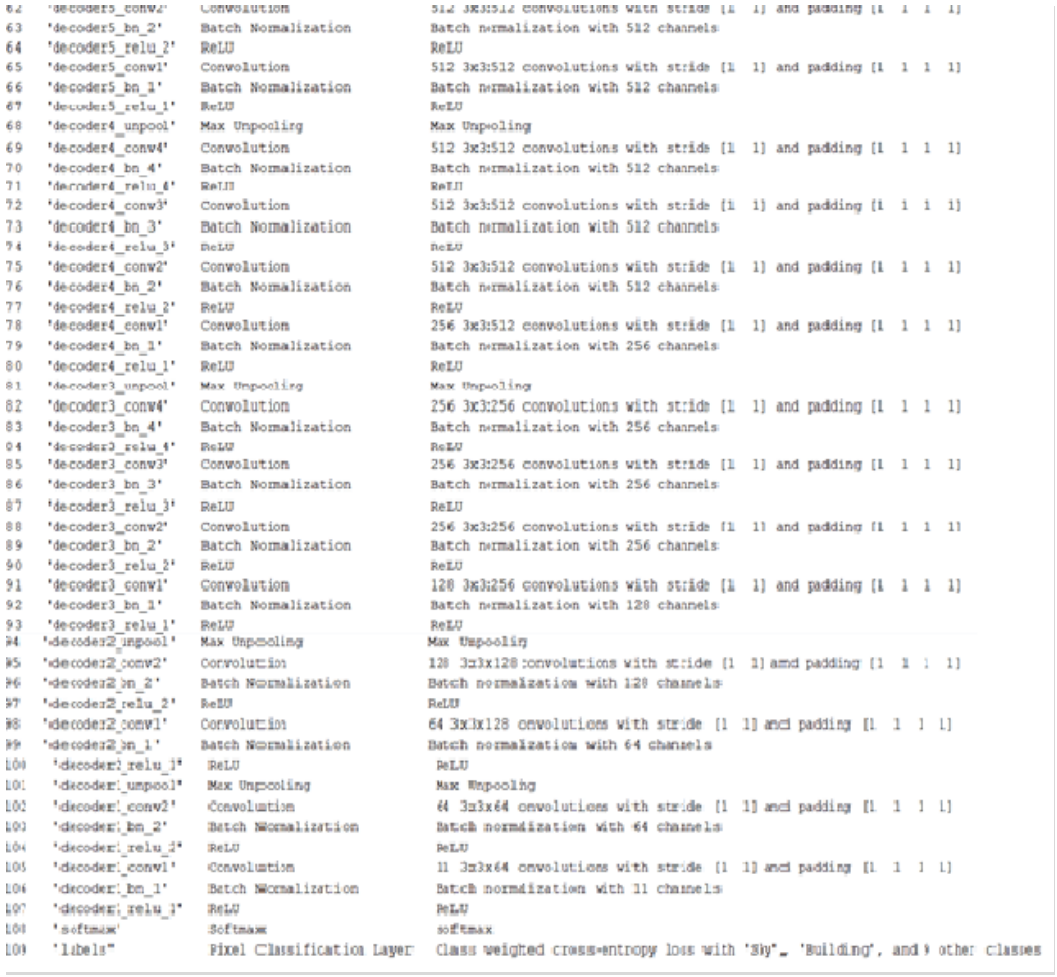


Figure 3.9 Parameters of Semantic Segmentation Model (2)

In the second step, we need to get the region of interest (ROI) and the corresponding coordinate parameters to indicate the location of vehicles in the image. In order to get this series of ground truth parameters, we use the MATLAB application to complete the task of labelling. It provides a convenient way to label a rectangular area of interest (ROI) by manually generating labelled ground truth data while manually labelling image frames from an image collection as shown in Figure 3.12.

The ground truth data contains image file name and object location of the rectangular ROI label. The rectangular ROI is a bounding box with the region [*x*, *width*]×[*y*, *height*] that specifies the position of an object in each image. The ground truth will be compared with the predicted data in the performance evaluation stage. It determines the effect of the model learning characteristics and directly affects the accuracy of the model evaluation.

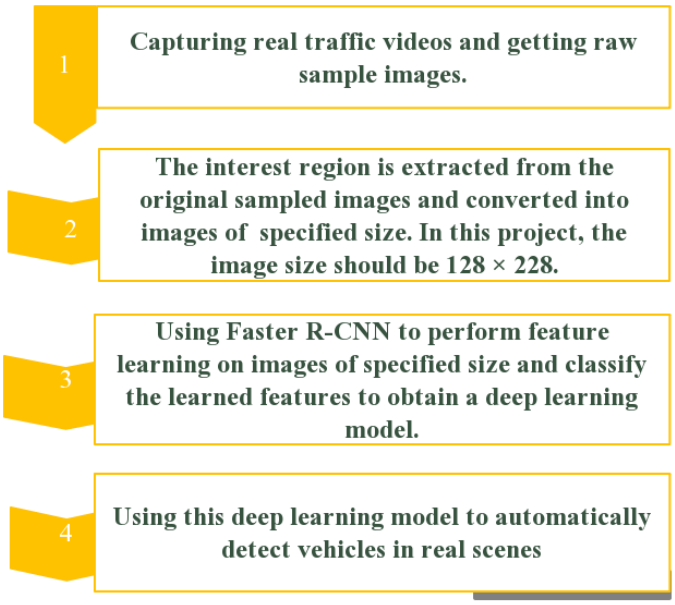


Figure 3.10 The flowchart of vehicle detection

Figure 3.11 The frames from a video

The dataset is split into a training set for the training detector and a test set for evaluating the detector. Among them, 60% of the data was employed for training, the remaining 40% was exploited for evaluation. The training dataset is transmitted to the

network to perform feature learning and classify the learned features to obtain a deep learning model.

We chose Faster R-CNN as our vehicle detector. Compared to the other two R-CNN based object detectors, R-CNN detector and the fast R-CNN detector, Faster R-CNN detector does not crop the proposal area from the image like the R-CNN detector and resizes it into CNN (Donahue, Darrell & Malik., 2014), but instead, the region proposes the corresponding CNN feature to process the entire image (Ross, 2015). Moreover, the calculation of the overlap region in the fast R-CNN detector is shared, so the fast R-CNN detector is more efficient than the R-CNN detector. Faster R-CNN discards the edge box algorithm and adopts the regional proposal. The network generates regional proposals directly in the network and applies anchor boxes for object detection (Ren et al., 2015). Overall, R-CNN detector trains object detectors for a shorter time, but the detection time is short. Faster R-CNN generates regional proposals faster in the network and better adapts to training data.

The network is trained in four steps. The first and second sections train regional proposals and detection network. The third and fourth sections combine the networks trained by the first two to create an entire vehicle detector.

Our object detection network consists of three parts. A feature extraction network is directly connected to the input layer, and two side-by-side sub-networks are received for the feature maps. One of the subnets connected to the feature extraction network is RPN, which is used to generate vehicle objects and backgrounds. The role of another subnet is to predict the actual class of the object generated by the RPN, such as vehicle and background. The feature extraction layer of our detector is shared by the RPN layer and the fully connected layer, consisting of the convolutional layer, ReLU layer, and max-pooling layer.

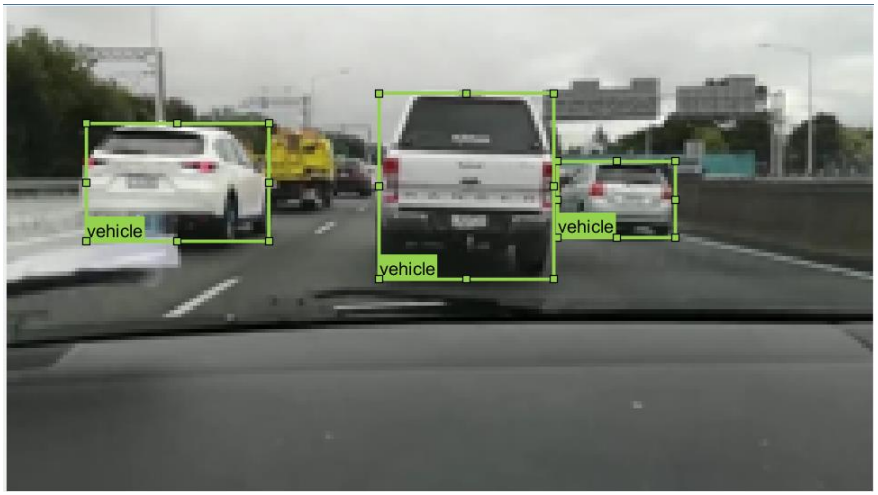


Figure 3.12 Feature extracting

The RPN network is adopted to generate region proposals. The softmax function judges whether the anchors belong to positive or negative bounding box to attain the final proposals. The RoI pooling layer combines the acquired feature maps and proposals. The synthesized feature map is then sent to the fully connected layer that performs category confirmation. The classification layer predicts the category based on the feature maps while obtaining the final precise location of the detected frame. In addition, the box regression layer in the network can refine bounding box locations. The region proposal layer as a part of RPN in the network is responsible for outputting the bounding box around the potential objects in the image. These outputs are further refined by additional layers within the network to produce the final object detection results, as shown in Figure 3.13. The parameters are shown in Figure 3.14.

The input size in the input layer is set to 32×32 for better convolution operations. All convolutional layers of this model use a 3×3 convolution kernel and set the step size to 1. Moreover, in the convolutional layer of Faster R-CNN, all the convolutions are subject to 1 padding expansion processing, resulting in an increase of 2 in length and width. It is this setup that does not change the size of the input and output matrices.

Similarly, the convolution kernel size and step size of the pooling layer in the model are both set to 2. Thus, the size of each matrix passing through the pooling layer becomes

one-half of the original. In other words, the convolutional layer and the ReLU layer maintain the size of the feature maps, the pooling layer reduces the size of the feature maps to 0.25 times.

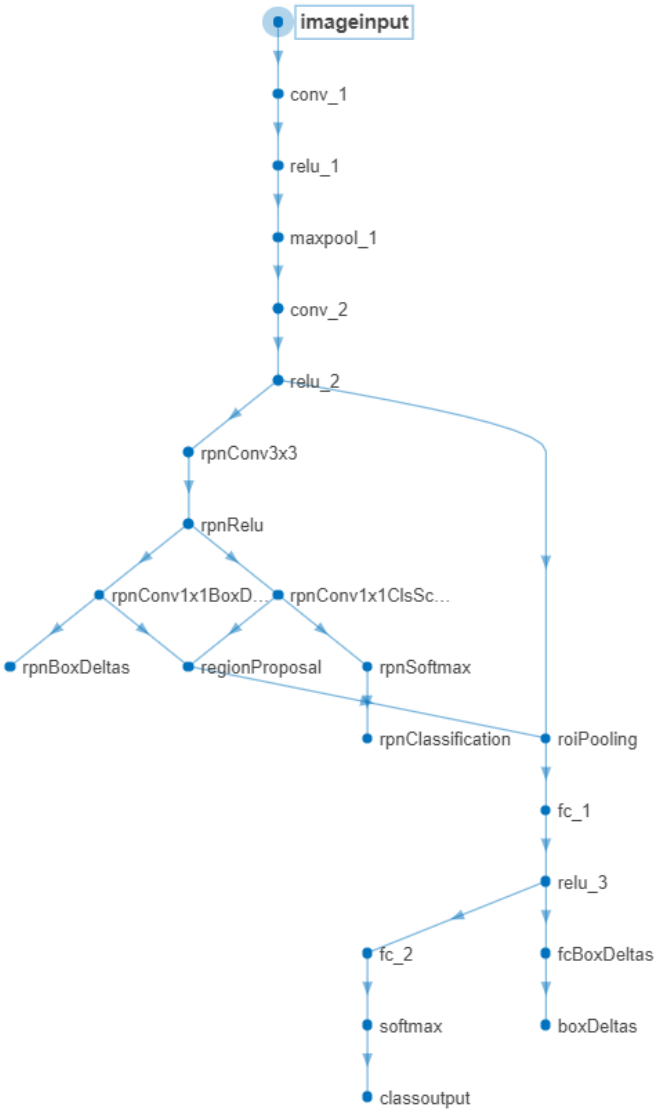


Figure 3.13 The network structure of vehicle detector

Moreover, this model converts the collected proposals in the RoI Pooling layer into 7×7 proposal feature maps and sends them to the classification layer. Feature maps are classified using the cross-entropy as a loss function in the classification layer.

In the end, we adopted this deep learning model to verify the test-set, further validate

the model, measure the accuracy and robustness of the model with average accuracy and miss rate.

3.3 Evaluation Methods

3.3.1 Evaluation Methods for Vehicle Detection

In order to construct a more accurate and robust detection model, the project considers the accuracy of the detection from different aspects. Precision, recall, miss rate, and false positive per image (FPPI) are used in the vehicle detection section to evaluate the usefulness of the detector. The precision and recall are adopted to calculate the average precision. It provides a single number to measure the detector's ability to classify objects and the ability to detect related objects. The log-average-miss rate evaluates that the rate of the detection results compared to ground truth, which is used to measure the performance of the object detector. For a multiclass detector, the log-average-miss rate is a vector of scores for each object class in the order specified by the ground truth.

In Table 3.1, p and p' express positive samples of actual and positive samples of predicted, respectively. Similarly, n and n' express negative samples of actual and negative samples of predicted, respectively. P is the number of real positive cases in the data and N is the number of real negative cases in the data.

- *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

ANALYSIS RESULT

	Name	Type	Activations
1	imageinput 32x32x3 images with 'zerocenter' normalization	Image Input	32×32×3
2	conv_1 32 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	32×32×32
3	relu_1 ReLU	ReLU	32×32×32
4	maxpool_1 3x3 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	15×15×32
5	conv_2 32 3x3x32 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	15×15×32
6	relu_2 ReLU	ReLU	15×15×32
7	rpnConv3x3 32 3x3x32 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	15×15×32
8	rpnRelu ReLU	ReLU	15×15×32
9	rpnConv1x1ClsScores 30 1x1x32 convolutions with stride [1 1] and padding [0 0 0 0]	Convolution	15×15×30
10	rpnSoftmax rpn softmax	RPN Softmax	225×15×2
11	rpnConv1x1BoxDeltas 60 1x1x32 convolutions with stride [1 1] and padding [0 0 0 0]	Convolution	15×15×60
12	regionProposal region proposal with 15 anchor boxes	Region Proposal	1×5
13	roiPooling ROI Max Pooling with pooled output size [7 7]	ROI Max Pooling	7×7×32
14	rpnBoxDeltas smooth-l1 loss	Box Regression Output	-
15	rpnClassification cross-entropy loss with 'object' and 'background' classes	RPN Classification Output	-
16	fc_1 64 fully connected layer	Fully Connected	1×1×64
17	relu_3 ReLU	ReLU	1×1×64
18	fc_2 2 fully connected layer	Fully Connected	1×1×2
19	softmax softmax	Softmax	1×1×2
20	classoutput crossentropyex with classes 'vehicle' and 'Background'	Classification Output	-
21	fcBoxDeltas 4 fully connected layer	Fully Connected	1×1×4
22	boxDeltas smooth-l1 loss	Box Regression Output	-

Figure 3.14 Parameters of vehicle detection

Precision refers to the number of correct positive predictions. The calculation should be presented as Eq. (3.1).

$$Precision = TP / (TP+FP). \tag{3.1}$$

The recall is also the true positive rate (TPR), which can be defined as the percentage of positive cases which is expressed as

$$Recall = TP /P = TP / (TP + FN). \tag{3.2}$$

Table 3.1 Form of classification criteria

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

The average precision (AP) can be understood as the area enclosed by using the precision-recall curve and the coordinate axis. If P and R are the precision a nd recall rate, respectively, AP should be

$$AP = \int_0^1 P(R)dR \quad . \tag{3.3}$$

Miss rate is false negative rate (FNR) that is the proportion of all negatives which yields positive test outcomes. It is calculated using Eq. (3.4)

$$MissRate = FNR = FN / (FN + TP). \tag{3.4}$$

False positive per image (FPPI) is defined as the average number that can be correctly retrieved in each graph which can be presented as

$$FPPI = FP / (P+N). \tag{3.5}$$

When $a_1, a_2, ..., a_9$ are positive values corresponding to the miss rates at nine evenly spaced FPPI points in log-space, between 10^{-2} and 10^0,

$$\log averagemissrate = (\prod_{i=1}^{n} a_i)^{\frac{1}{n}} = \exp[\frac{1}{n}\sum_{i=1}^{n} \ln a_i] \quad . \tag{3.6}$$

3.3.2 Evaluation Methods of Semantic Segmentation

In order to measure the value of this semantic segmentation model, Accuracy (*ACC*), Intersection over Union (IoU)，Global Accuracy and Weighted IoU are exploited in this project to evaluate the segmentation results of this model.

The precision represents the percentage of pixels that each class is correctly identified. According to the basic fact, the accuracy is ratio of the correctly classified pixels in the object to the total number of pixels in the class (Zhao et al., 2018). The mathematical expression of accuracy is

$$ACC = TP / (TP + FN). \tag{3.7}$$

For each class, IoU is the ratio of correctly classified pixels to the total number of ground truth and predicted pixels in that class (Zhao et al., 2018). Eq. (3.8) shows how IoU is calculated by using mathematical method.

$$IoU = TP / (TP + FP + FN). \tag{3.8}$$

Considered the entire dataset, *GlobalAccuracy* is the ratio of the correctly categorized pixels to the total number of pixels (Zhao et al., 2018), allowing fast and computationally inexpensive estimates of the percentage of correctly categorized pixels. It is defined as

$$GlobalAccuracy = \frac{\sum_i n_{ii}}{\sum_i t_i} . \tag{3.9}$$

where n_{ii} is the number of pixels of class i predicted to belong to class i and t_i is the total number of pixels of class i (Ghosh, Li & Chakareski, 2018).

Similarly, *weightedIoU* is one of the performance indicators based on the entire dataset. When the proportions of the categories in the dataset are not balanced, *WeightedIoU* can reduce the impact of errors in the class on the overall assessment score. *WeightedIoU* can be defined as average IoU of each class, which is weighted by the

number of pixels in that class (Jude et al., 2019). The expression is

$$WeightedIoU = \left(\sum_k t_k \right) \frac{\sum_i t_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}} \tag{3.10}$$

where n_{ii} is the number of pixels of class i predicted to belong to class i. n_{ij} is the number of pixels of class i predicted to belong to class j, t_i is the total number of pixels of class i (Ghosh, Li & Chakareski, 2018).

Chapter 4 Results

The main content of this chapter is to introduce the schema of the whole methods and the implementation of vehicle-related scene understanding. The experimental environment will be built in this chapter. In addition, this chapter will clarify the results of image segmentation and vehicle detection. At the same time, the findings and limitations of this paper will be pointed out at the end of this chapter.

4.1 Experimental Parameters and Environment

4.1.1 Experimental Parameters and Environment for Semantic Segmentation

One of the most challenging tasks for training a deep learning model is the number of parameters that need to be processed, including width, depth, connection mode of the network itself as well as the hyperparameter design and debugging of the loss function. In addition, there are learning rates, batch sample sizes, optimizer parameters, and others. The large number of parameters will have direct or indirect effects on the ultimate effective tolerance of the network model. Faced with so many parameters, if optimizing them one by one, the time and resources required are impractical. Therefore, it is more critical to perform reasonable debugging based on hyperparameters.

This project focuses on the understanding of the traffic environment around the vehicle through monitoring. It exploits the Intel Core i7-8550 CPU to train deep neural networks and 8.00G memory to complete file storage. Regarding software, this project preprocesses digital images and completes network refine based on MATLAB 2018b environment. In the process of training the neural network, critical parameters need to be set as shown in Table 4.1.

When the network weight is initialized, setting a suitable initial weight is very necessary for the network, which can make the loss function converge fast during the training process. However, if the network weights are randomly initialized, the initial weight of the network will not be guaranteed in an appropriate state for each initialization. If there is a problem with the initial weight setting, the loss function probably reaches a local minimum. Therefore, setting an appropriate initial weight is the key step to achieve a global optimal model. This project sets the momentum to 0.9 so as to resolve this issue. When the momentum becomes larger, more momentum is converted into potential energy. Therefore, the more potential energy will act on the local concave domain, make it into the global concave domain. In other words, if the previous round of momentum is as same

as the current negative gradient direction, then the magnitude of this decline will increase, thus accelerate the convergence.

Learning rate can be one of the crucial hyperparameters in model training. Generally, one or a group of excellent learning rates can both speed up the training of the model and get better or even better precision. Too large or too small learning rate will directly affect the convergence of the model. If the learning rate is too low, the training will progress very slowly because the weight of the network has only been adjusted very little. However, if the learning rate is too high, it may have undesired consequences on the loss function. In this project, the initial learning rate is set to 1×10^{-3}.

The max epoch is also the key to affect the overall performance of a model. Each time, a round of forward propagation and backpropagation is completed at the same time, the parameters of the network model are updated, and the value of the loss function is reduced. The most suitable max epoch in this project is 400. The appropriate max epoch is, when the error rates of training and testing are low, both error rates are within acceptable limits. If the network is trained continuously, there may exist overfitting.

If the learning rate directly affects the convergence of the model computations, the batch size affects the generalization of the model. Model performance is less sensitive to batch size than learning rate, but batch size becomes a very critical parameter when further improving model performance. Large batch size reduces training time and improves stability. However, as the batch size increases, the performance of the model will decrease. Therefore, it is vital to choose the right mini-batch size. In this project, mini-batch size is set to 8.

In addition, *CheckpointPath* is set to a temporary location. At the end of each training, the network checkpoint is automatically saved in the path. If training is interrupted due to a system failure or power outage, the saved checkpoint can recover some of the training data.

Table 4.1 Training Parameters

Parameters	*Description and Setting*
Momentum	The last parameter update contributed to the current iteration of the stochastic gradient descent with momentum. Set it to 0.9.
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 10^{-3}.
Max Epochs	The maximum number of epochs for training. The epoch representation model completes a training throughout the training set. It is set to 400.
Mini Batch Size	Size of the mini-batch to use for each training iteration. It is set to 8.
Checkpoint Path	Path for saving the checkpoint networks. Set it to *tempdir*.

During the training, we conducted a full observation of epoch, the mini-batch accuracy and minibatch loss. Figure 4.1 shows the data from epoch 1 to epoch 21. As the number of iterations increases, the mini-batch accuracy shows a steady upward trend, while the mini-batch loss slowly declines as the training progresses.

Command Window

```
Training on single CPU.
Initializing image normalization.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:55 |        9.01% |       2.5914 |          0.0010 |
|       1 |           2 |       00:01:38 |        9.09% |       2.6014 |          0.0010 |
|       2 |           4 |       00:03:13 |        8.91% |       2.5879 |          0.0010 |
|       2 |           6 |       00:04:33 |        8.91% |       2.5933 |          0.0010 |
|       3 |           8 |       00:05:56 |        8.99% |       2.5917 |          0.0010 |
|       4 |          10 |       00:07:22 |        8.82% |       2.5938 |          0.0010 |
|       4 |          12 |       00:08:42 |        9.52% |       2.5851 |          0.0010 |
|       5 |          14 |       00:10:03 |        9.31% |       2.5895 |          0.0010 |
|       6 |          16 |       00:11:26 |        9.34% |       2.5942 |          0.0010 |
|       6 |          18 |       00:12:54 |        9.71% |       2.5762 |          0.0010 |
|       7 |          20 |       00:14:19 |       10.19% |       2.5719 |          0.0010 |
|       8 |          22 |       00:15:41 |       10.11% |       2.5659 |          0.0010 |
|       8 |          24 |       00:17:03 |        9.97% |       2.5876 |          0.0010 |
|       9 |          26 |       00:18:37 |       10.12% |       2.5766 |          0.0010 |
|      10 |          28 |       00:20:00 |       10.24% |       2.5661 |          0.0010 |
|      10 |          30 |       00:21:20 |       10.97% |       2.5485 |          0.0010 |
|      11 |          32 |       00:22:45 |       10.57% |       2.5506 |          0.0010 |
|      12 |          34 |       00:24:07 |       10.91% |       2.5618 |          0.0010 |
|      12 |          36 |       00:25:25 |       11.04% |       2.5397 |          0.0010 |
|      13 |          38 |       00:26:49 |       11.61% |       2.5358 |          0.0010 |
|      14 |          40 |       00:28:12 |       11.94% |       2.5310 |          0.0010 |
|      14 |          42 |       00:29:31 |       12.02% |       2.5163 |          0.0010 |
|      15 |          44 |       00:30:53 |       12.60% |       2.4891 |          0.0010 |
|      16 |          46 |       00:32:15 |       12.59% |       2.5288 |          0.0010 |
|      16 |          48 |       00:33:38 |       12.71% |       2.4854 |          0.0010 |
|      17 |          50 |       00:35:02 |       13.10% |       2.4791 |          0.0010 |
|      18 |          52 |       00:36:25 |       13.60% |       2.4617 |          0.0010 |
|      18 |          54 |       00:37:44 |       13.92% |       2.4771 |          0.0010 |
|      19 |          56 |       00:39:05 |       13.89% |       2.4708 |          0.0010 |
|      20 |          58 |       00:40:28 |       14.05% |       2.4753 |          0.0010 |
|      20 |          60 |       00:41:48 |       14.79% |       2.4302 |          0.0010 |
|      21 |          62 |       00:43:11 |       14.73% |       2.4580 |          0.0010 |
fx
```

Figure 4.1 Partial training data

4.1.2 Experimental Parameters and Environment for Vehicle Detection

The network of vehicle detection is trained in the same experimental environment as Table 4.1. The training is divided into four stages; at the first stage, a Region Proposal Network (RPN) is trained; in the second stage, a Faster R-CNN Network is modelled

using the RPN; in the third stage, RPN is re-trained using weight sharing with Faster R-CNN; finally, training Faster R-CNN is used to update RPN. In these four phases, we all train the four parameters: *MaxEpochs*, *MiniBatchSize*, *InitialLearnRate*, and *CheckpointPath*.

In order to reduce the amount of noise in the gradient between batches, allowing stochastic gradient drops to be closer in the best direction can ensure better generalization, we set *MiniBatchSize* to 1, *MaxEpochs* is set to 10 for training, and *InitialLearnRate* is set to 1×10^{-3}.

4.2 Experimental Results

We use deep learning to achieve scene understanding. In order to verify the advantages of deep learning in scene understanding, we explored semantic segmentation and vehicle detection through experiments.

4.2.1 Results of Semantic Segmentation

This project experimented with a collection of 91 Auckland street view images and a corresponding set of 91 pixels labelled images. There are 60% of the datasets which are set as training set, the remaining 40% is set as the test set. That is to say, this experiment trains 55 raw images and 55 labelled images. At the same time, 36 raw images and 36 labelled images are tested. Furthermore, in order to enrich the original dataset and increase its size so as to improve the accuracy of the network, the original image is enhanced through random reflection, the random translation along to *x*-axis and *y*-axis. For memory saving and training time reducing, this experiment uniformly resizes images to 360×480×3. Then, the proposed neural network classifies and labels each pixel on the input image. Experiments show that as the processed dataset increases, the accuracy of the model is higher.

As shown in Figure 4.2, visually, the model in the semantic segmentation results is

satisfactory for the segmentation of buildings, sky, buslane, and vehicles.

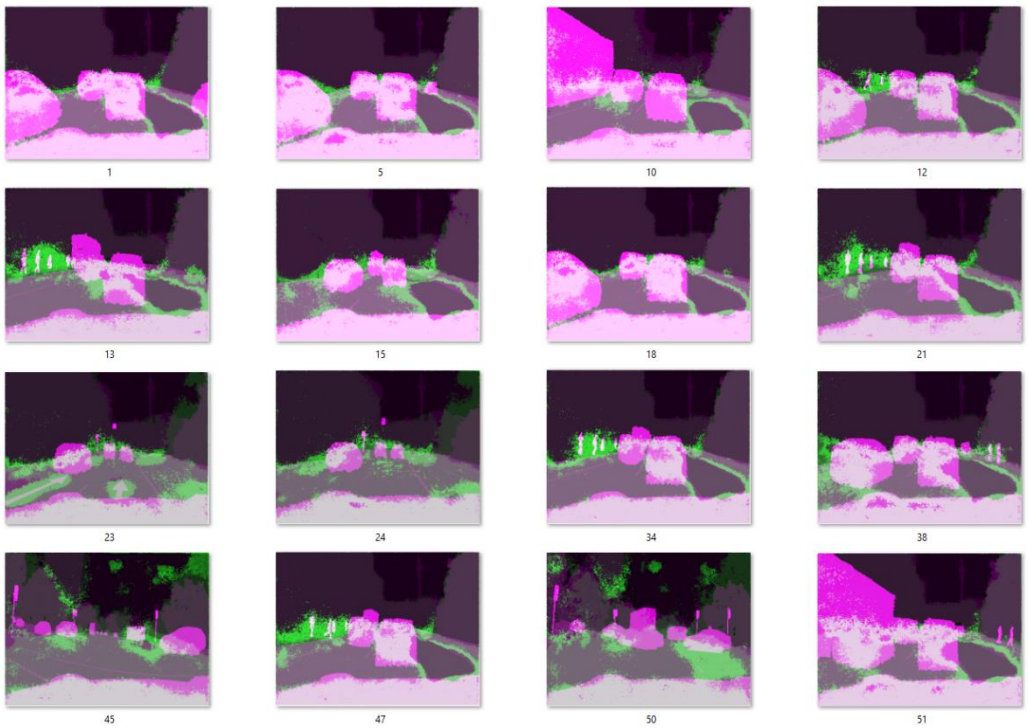


Figure 4.2 Segmentation Results

However, objects such as pedestrians, lanes, and trees are not accurate. The project uses IoU metrics to measure the amount of overlap for each class. Taking the first segmentation result in Figure 4.2 as an example, the IoU of each category in which a single image is segmented is shown in Figure 4.3. Among them, the building *IoU* is up to 87%, with the highest accuracy among the 11 categories. At the same time, the proposed model scored higher on roads, sky, buslane, and vehicles, with 75%, 73%, 71%, and 76% respectively. However, the segmentation scores for lane, tree, and turnsign are lower. The reason why we get these results is that the objects in the dataset lane and traffic sign are generally smaller, the probability of occurrence is smaller, and the distance is farther. In future, we will focus on collecting data including lane, tree, and turnsign images.

```
ans =

  11×2 table

      classes          iou
    _____________    ________

    "Sky"            0.73946
    "Building"        0.8788
    "Buslane"        0.71061
    "Road"           0.75492
    "Lane"           0.48905
    "Tree"          0.073477
    "Trafficsign"        NaN
    "Turnsign"       0.01617
    "Vehicle"        0.76446
    "Pedestrian"           0
    "Trafficlight"         0
```

Figure 4.3 IoU of Single Images Segmentation

Table 4.1 shows the semantic segmentation metrics used to assess the quality of the model on the test dataset. The *GlobalAccuracy* is used to make fast and computationally inexpensive estimates of the percentage of correctly classified pixels. At the same time, we also take advantage of weighted IoU metrics, the average IoU of each class, weighted by the number of pixels in the class. The weighted IoU metric reduces the impact of errors generated by using imbalances in the number of pixels in the class. The proposed model has 77% and 65% *GlobalAccuracy* and *WeightedIoU* on this dataset, respectively, which indicates that the model performs well on pixel classification. However, it is necessary to further aggrandize the performance of the model with the number of unbalanced pixels in the class.

Table 4.2 Segmentation Metrics of the Model

Evaluation methods	**Value**
Global Accuracy	77%
Weighted IoU	65%

This project uses class metrics for each class, including classification accuracy and the intersection over union (IoU), as shown in Table 4.2.

Table 4.3 The metrics For Each Class (I)

	Accuracy	**IoU**
Sky	91%	74%
Building	81%	71%
Buslane	90%	76%
Road	86%	61%
Lane	70%	50%
Vehicle	68%	60%

For each class, the accuracy is the ratio of the correctly categorized pixels to the total number of pixels in that class. As shown in Table 4.3, the accuracy of the Sky is the highest among all classes at 91%. Secondly, buslane, vehicle, road, and the building also achieved satisfied levels of accuracy, 90%, 68%, 86%, and 81% respectively. Meanwhile, the project exploits IoU to measure the overlap between the ground truth and the labelled image in the dataset and generate the resulting image from the proposed model. As shown in Table 4.3, the overlap rate of buildings, sky, and Buslane is high, 71%, 74%, and 76% respectively. In contrast, the accuracy and *IoU* of lane detection are worse than those of other categories, because the labelled area is small, and its frequency of occurrence is lower than other classes.

4.2.2 Results of Vehicle Detection

From the experimental results of vehicle detection, the detector can successfully detect

vehicles in different directions, different sizes, and different types of vehicles under normal conditions and even detect vehicles with only half of the body appearing in the imaging range, as shown in Figure 4.4.

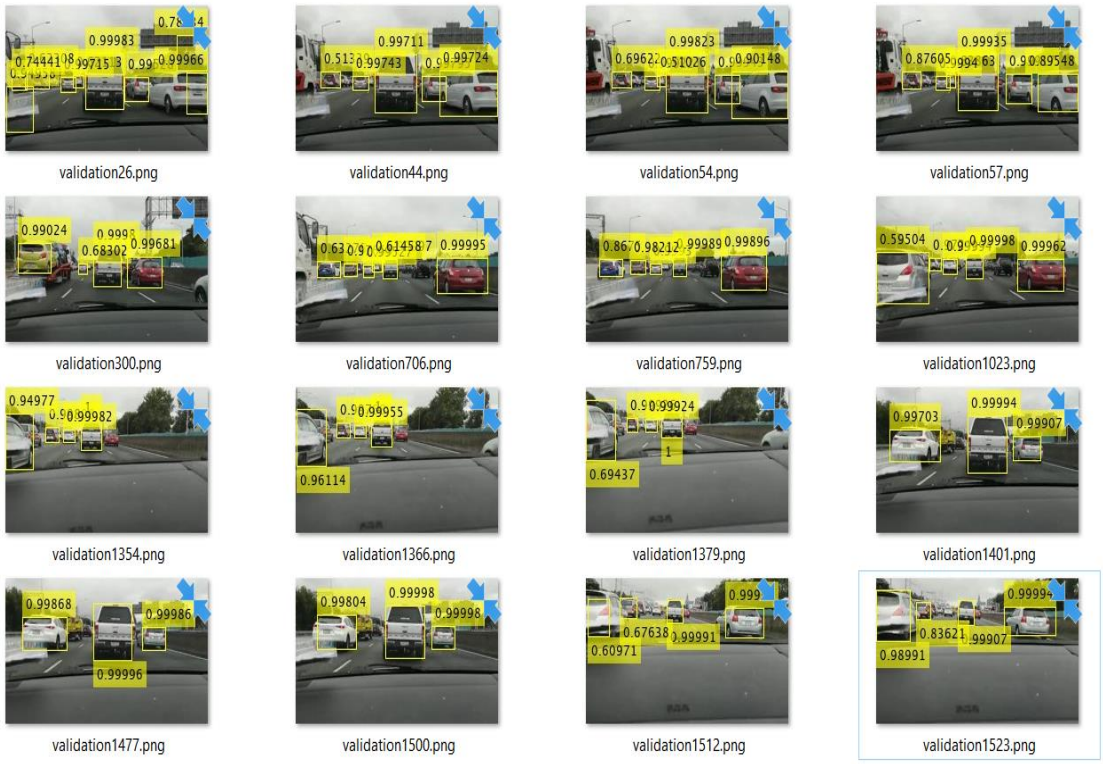


Figure 4.4 Vehicle detection results

However, when there is interference in the detection environment, our detection accuracy will be decreased. Figure 4.5 shows that when the road tax sticker appears on the windscreen, the detector incorrectly detects the road tax sticker as a vehicle. On the other hand, the detector can roughly detect the presence or absence of a vehicle in a particular region, but the position of the label boxes is offset.

This project takes advantage of average precision to measure the accuracy of the detector, which is expressed as a precision/recall curve in Figure 4.6. Both precision and recall are based on an understanding and measure of relevance. Ideally, the precision will be 1.0 at all recall levels. Each stage is iterated ten times after four stages of training. The

best training result of average precision for this project is 81%, which used a 22-layer Faster R-CNN network.

Figure 4.5 Errors and inaccurate results

Figure 4.7 states the log-average miss rate of the detection results compared to ground truth, which is adopted to measure the performance of the object detector in this project, miss rate decreases as false positive per image (FPPI) grows, and the value of the log average miss rate is 0.4.

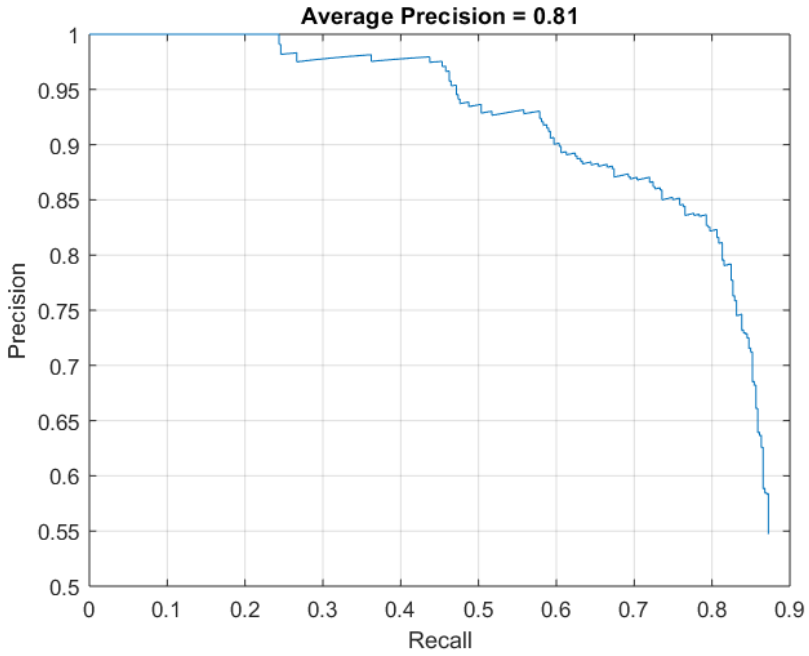


Figure 4.6 Precision/Recall Curve

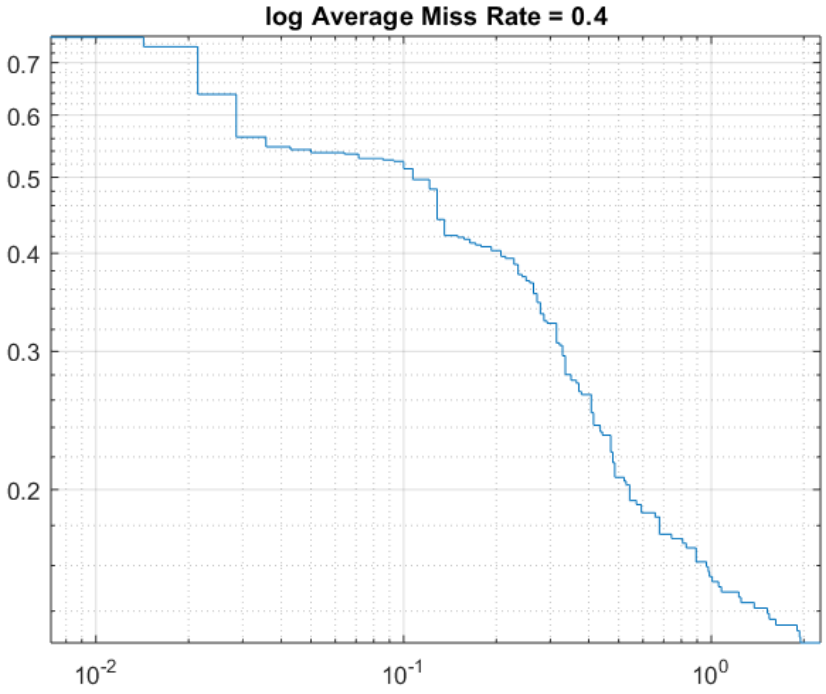


Figure 4.7 MissRate / Fppi Curve

Chapter 5
Analysis and Discussions

In this chapter, comparisons of the results of proposed models and other models are provided. Moreover, this project also compares the parameters involved in the model training and test.

5.1 Analysis

We have directly introduced the results of scene understanding. This chapter will compare our results with the results of other models. Simultaneously, our detailed analysis and discussion will be conducted.

5.1.1 Analysis of Vehicle Detection

In order to make the network more stable and more accurate, it is necessary to compare the average precision of the 19, 20, 21, 22, 23, and 24 layers networks respectively. By observing in Figure 5.1, the networks can converge when the number of layers is 19, 20, 21, 22, 23, and 24 layers, but the performance is various. When the number of network layers is 19, the *AP* in the test set is 70%, which can effectively extract features for vehicle detection; the test results of 20 and 21 layers are 69% and 72%. Although they can converge, it takes longer time and more rounds of iterations to learn the useful features during the training process; the 22-layer structure has more training parameters and complex structure. In the case of the limited training set, the convergence speed is also longer, but the model has made the satisfactory results. When the number of network layers is 22, *AP* reaches 81%, which is 9% higher than the result of the 21-layer-structure.

Although meaningful results were obtained when the number of network layers was 22, in order to increase the integrity and persuasiveness of the experiment, we tried 23 layers and 24 layers, the experimental results show that the APs were 73% and 67%, respectively. It is 8% and 14% lower than the 22-layer structure, which takes a long time because the number of network layers is deep. Although all of the layers can converge, the 22-layer structure is more accurate and takes a significantly shorter time than other structures. More importantly, it achieves a lower average miss rate. The 22-layer structure can ensure the convergence speed during accuracy.

Therefore, for different sample sets, there is no qualitative standard for the optimal number of layers in the network. A large number of experiments can be employed to

obtain a relatively appropriate number of layers. The size of the input samples, the complexity, and the depth of the network are specific.

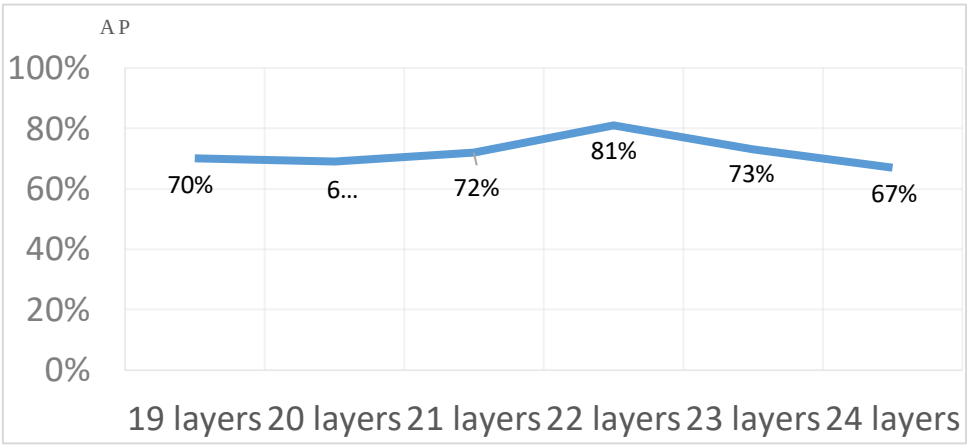


Figure 5.1 Average Precision Comparison with Difference Layers

To ensure the contribution and performance of this model, we introduce four other models as the references. They are AlexNet-FasterRCNN, ResNet18-FasterRCNN, ResNet18-YOLOv2, MF. Among them, AlexNet-FasterRCNN and ResNet18-FasterRCNN are the FasterRCNN models trained by AlexNet and ResNet as the base model, ResNet18-YOLOv2 is YOLOv2 model trained by using ResNet18 as the base model, MF is a model that is a FasterRCNN model trained by using two sets of the convolutional layer, ReLU layer, and a max pooling layer as the base model. Figure 5.2 shows the results of testing five models using the same image.

Among them, AlexNet-FasterRCNN successfully detected four vehicles, but the position of the rectangular box was offset. Moreover, the size control of the rectangular box by using AlexNet-FasterRCNN is not satisfactory enough. There is no significant difference between the detection results of ResNet18-FasterRCNN and AlexNet-FasterRCNN.

ResNet18-FasterRCNN also has been offered to detect four vehicles, and one vehicle was missed. There is also a problem of offset of the rectangular boxes. The result of ResNet18-YOLOv2 is weak. It only detected one vehicle and missed four, but the position

and size of ResNet18-YOLOv2 rectangular boxes are more accurate than the previous two models.

Compared with the first three models, the results of MF and our model are more precise. The size of the MF rectangular box is relatively suitable for the detection of vehicles, but the MF has not fully detected all the vehicles. Our model successfully detected all vehicles. The position and size of the rectangular frame were relatively accurate. Moreover, our model can even detect vehicles that are half-body occluded.

Further observation of ResNet18-YOLOv2 and ResNet18-FasterRCNN reveals when ResNet18 is also used as the base model, ResNet18-FasterRCNN has far better results than ResNet18-YOLOv2 in Figure 5.2. According to the experimental results, the network model of Faster RCNN may be more suitable for this project by using this dataset for vehicle detection.

5.1.2 Analysis of Semantic Segmentation

For obtaining a high-performance semantic segmentation model, it is necessary to select a suitable *maxepoch* during the training process. In this project, in order to observe the model performance change at each stage, we set *maxepoch* to 100, 200, 300, 400 and 500 in a fixed case with the batch size of 8 and learn the rate of 1×10^{-3}, and then record the Globel. The trends in accuracy and *WeightedIoU* are shown in Figure 5.3.

When the *maxepoch* is set to 100, *GlobalAccuracy* and *WeightedIoU* are 41% and 29%, respectively. When *maxepoch* is increased to 200, *GlobalAccuracy* and *WeightedIoU* are also increased to 41.5% and 31.3%. From the numerical value, the performance of the network is slightly improved. Even though the increase in epoch has improved *GlobalAccuracy* and *WeightedIoU* by 0.5% and 2.3%, respectively, in the case of *maxepoch* as 100 and 200, the network does not converge; that is, the less number of epoches does not make the network learning the features well.

Figure 5.2 Resultant comparison of four different model with the same image

When *maxepoch* is set to 300, *GlobalAccuracy* and *WeightedIoU* are 59.1% and 44%. Although it has not reached a satisfactory level, the network gradually converges in this state.

When *Maxepoch* was set to 400, *GlobalAccuracy* and *WeightedIoU* made significant improvements at 77.2% and 64.5% respectively. Compared to 300 epoches, 400 epoches *GlobalAccuracy* and *WeightedIoU* increased 18.1% and 20.5%, respectively. It shows that

the network has the ability to learn most of the features in the dataset, resulting in better convergence.

To further explore the impact of epoch on *GlobalAccuracy* and *WeightedIoU*, we also trained the network using 500 epochs. However, the results show that increasing the epoch from 400 to 500 does not help with network performance. In the case of 500 epoches, the training time increased 5 hours, but the results of *GlobalAccuracy* and *WeightedIoU* were not significantly different from the case having 400 epochs. The gradient of the loss function updates during the network training process is significantly reduced. In order to save time and eschew overfitting, this project chose 400 max epochs to train the network.

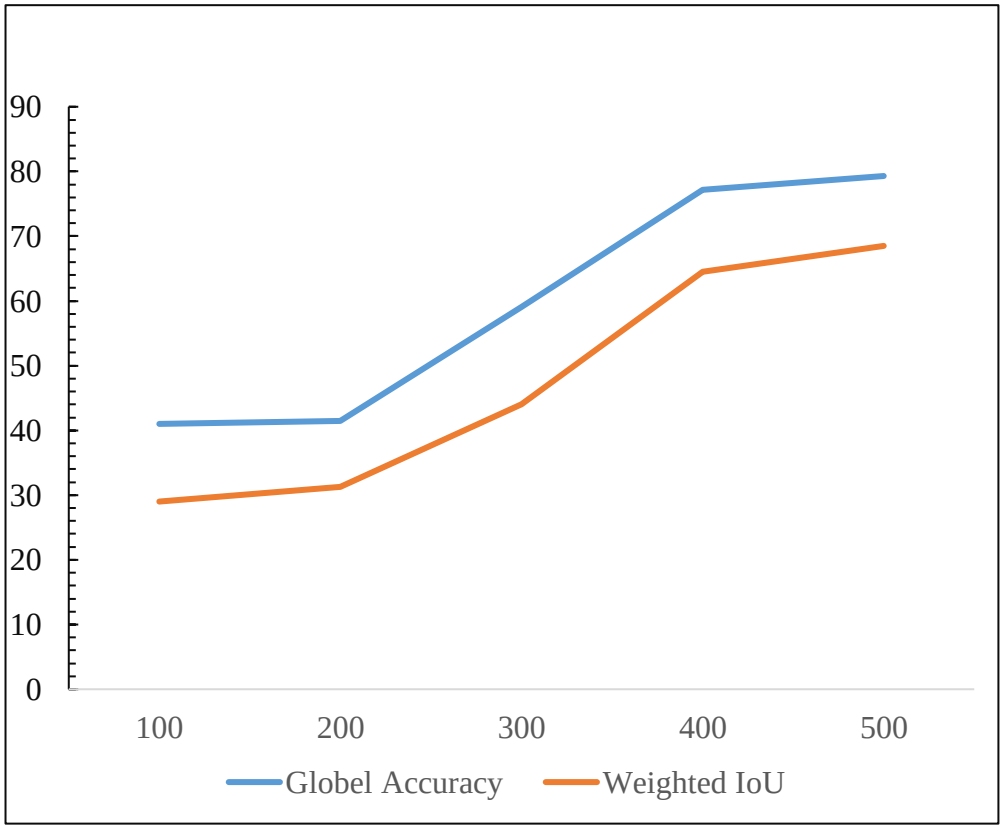
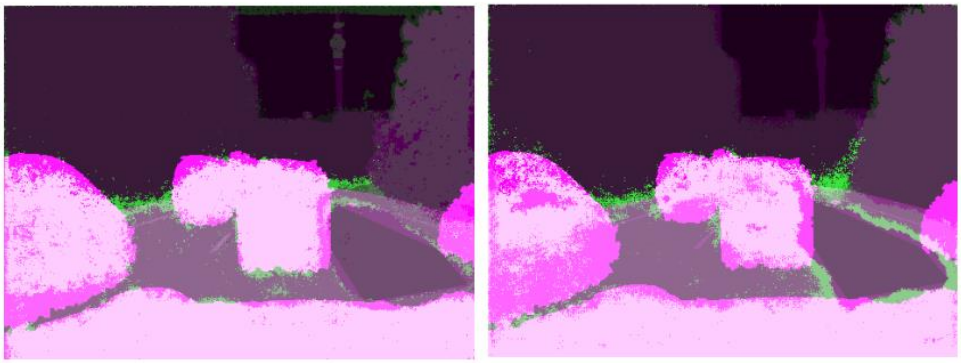


Figure 5.3 The effect of *maxepoch* on the performance of the model

Through observing the segmentation results of VGG16-SegNet in Figure 5.4(a) and VGG19-SegNet in Figure 5.4(b), these two segmentations in buslane, road, and vehicle

are obviously different. According to the *IoU* of each category of the same images in Table 5.1, the category with most substantial difference of IoU value is buslane, and the performance of VGG19-SegNet on buslane is 15% higher than VGG16-SegNet. At the same time, the VGG19-SegNet segmentation results in the sky are also better than VGG16-SegNet 7%. However, the segmentation effects of VGG19-SegNet are not dominant in all categories. In Table 5.1, the segmentation results of VGG19-SegNet on road, lane, and vehicle are obviously at a disadvantage, lower than VGG16-SegNet 9%, 8% and 10%, respectively.

The experimental result shows that VGG19-SegNet is generally better than VGG16-SegNet for larger size objects such as sky, building and buslane. However, for smaller objects such as vehicles, lane, VGG19-SegNet is far less than VGG16-SegNet. The reason is that VGG19-SegNet has three more convolution layers than VGG16-SegNet, which makes it more accurate for large-scale objects, but it causes over-fitting of small-scale objects.



(a) (b)

Figure 5.4 The result of two model in the same image

We use a single image as an example to construe and discuss the performance of VGG16-SegNet and VGG19-SegNet. The data from the perspective of the entire dataset is shown in Table 5.2, we adopted *GlobalAccuracy* and *WeightedIoU* to evaluate the indicators. The *GlobalAccuracy* results for VGG16-SegNet and VGG19-SegNet were 76%

and 77%, respectively. The *WeightedIoU* estimates were 64% and 65%, respectively. By comparison, VGG19-SegNet is 1% higher than VGG16-SegNet in the evaluation of the above two indicators. It shows that VGG19-SegNet can classify pixels more accurately. Moreover, VGG19-SegNet can reduce the impact of errors in small classes due to disproportionate sizes in the image on overall accuracy.

Table 5.1 Comparison of VGG16-SegNet and VGG19-SegNet in IoU

Class	VGG16-SegNet IoU	VGG19-SegNet IoU
Sky	74%	81%
Building	87%	89%
Buslane	71%	86%
Road	75%	66%
Lane	49%	41%
Vehicle	76%	66%

Table 5.2 Semantic Segmentation Metrics of VGG16-SegNet and VGG19-SegNet

	VGG16-SegNet	VGG19-SegNet
Global Accuracy	76%	77%
Weighted IoU	64%	65%

By comparing the values of *GlobalAccuracy* and *WeightedIoU*, VGG19-SegNet achieved much at the whole dataset level. However, it is also indispensable to explore and

judge the performance of the model for each category. In this project, *accuracy* and *IoU* in Table 5.3 are used to evaluate and compare the semantic segmentation results of VGG16-SegNet and VGG19-SegNet.

Table 5.3 The Class Metrics For Each Class (II)

	Accuracy		*IoU*	
	VGG16-SegNet	VGG19-SegNet	VGG16-SegNet	VGG19-SegNet
Sky	78%	91%	66%	74%
Building	84%	81%	72%	71%
Buslane	82%	90%	59%	76%
Road	91%	86%	59%	61%
Lane	68%	70%	50%	50%
Vehicle	72%	68%	67%	60%

The most apparent difference between the two models in accuracy is on sky. The accuracy of VGG16-SegNet and VGG19-SegNet in sky is 78% and 91% respectively. Compared to VGG16-SegNet, VGG19-SegNet segmentation accuracy increased 13%. Moreover, VGG19-SegNet's accuracy in the buslane category is 90%, and VGG16-SegNet is 82%, which also shows a salient advantage. The accuracy of VGG16-SegNet and VGG19-SegNet on sky is 78% and 91% respectively. Compared to VGG16-SegNet, VGG19-SegNet's segmentation accuracy increased 13%. More significantly, VGG19-

SegNet accuracy in the buslane category is 90%, and VGG16-SegNet is 82%, which also shows a prominent advantage. Conversely, the classification results of VGG19-SegNet in building, road and Vehicle are slightly lower than VGG16-SegNet, but the difference is not more than 4 %.

VGG19-SegNet also shows a tremendous outburst in specific categories when using IoU as the evaluation standard. As shown in Table 5.4, according to the results of Buslane class, compared with VGG16-SegNet with IoU value of 59%, the segmentation result of IoU is 76%, VGG19-SegNet is better. Furthermore, The IoU value of the Sky is also higher than VGG16-SegNet 8%. The overlapping of the VGG19-SegNet segmentation result and the expected results for Lane looks as same as VGG16-SegNet, both at 50%. In contrast, the IoU of Vehicle in VGG16-SegNet is 67% and only 60% on VGG19-SegNet.

5.2 Discussions

5.2.1 Discussions of vehicle detection

Object detection is one of the essential tasks in scene understanding. This project takes vehicle identification as an example, we complete the production of datasets and the construction of models. We chose Faster R-CNN as the main model and created a meaningful base model based on the size of the dataset and the size of the image. Moreover, we elaborated on model parameters and training hyperparameters in Section 3.2.2.

In Section 4.2.2, the final results of the experiment are described on details. The achievements and shortcomings of the experimental results are also construed in depth. The project also shows a comparison between the number of layers, and model types in Section 5.1 in order to verify the value of the proposed model. The results show that by comparing the network performance from the 19th to the 24th layers in the same environment, the 22-layer network is 81% higher than the other networks. Moreover, we

compare our model with Faster R-CNN models and YOLO models, including AlexNet-FasterRCNN, ResNet18-FasterRCNN, ResNet18-YOLOv2, and MF. Our model has significant advantages in detection performance. Moreover, through the results of ResNet18-FasterRCNN and ResNet18-YOLOv2, it is not difficult to find that Faster RCNN is more suitable for our project when the same pre-training model is used as the base model.

5.2.2 Discussions of Semantic Segmentation

This project takes semantic segmentation as a vital part of vehicle-related scene understanding. The operating environment and parameter settings of the semantic segmentation task in this project are described in Section 4.1, the experimental results are described in Section 4.2. Next, comparing this model with other models in Section 5.1.2, we will verify its usability.

In Section 3.1.1, this model employs VGG19 as the SegNet model of the pre-training network, namely, VGG19-SegNet. The experimental results in Section 4.2.1 consist of three parts: (1) Using *IoU* to verify the overlap of expected and predicted values for each category on a single image, (2) evaluating the model across the entire dataset by *GlobalAccuracy* and *WeightedIoU*, (3) *Accuracy* and *IoU* are exploited as evaluation criteria to measure the pros and cons of the results of each model in the entire dataset. The results show that VGG19-SegNet performed superior in sky, building, road, buslane, and vehicle, but the results of lane detection have yet to be improved. Then, this project introduced VGG16-SegNet with VGG16 as the pre-processing network as a comparison item in Section 5.1.2.

In contrast, VGG19-SegNet employs Global Accuracy and Weighted IoU to evaluate 1% higher than VGG16-SegNet. Among the results of measuring each category with Accuracy and IoU as the evaluation criteria, the best results of VGG19-SegNet were higher than 10% of VGG16-SegNet, and the worst result was less than 7% of VGG16-SegNet.

Chapter 6
Conclusion and Future Work

In this project, an in-depth explanation of the vehicle-related scene understanding is discussed. We elaborated on the research results and the innovation of research methods. In this chapter, we will present this argument at the scholar level. Additionally, we also integrate and organize the conclusions into the context, meanwhile point out the future work at the end of this thesis.

6.1 Conclusion

The goal of this project is to achieve an understanding of traffic scenes using deep learning, including semantic segmentation and vehicle detection. The availability of the neural network and datasets is demonstrated by adjusting the number and weight of convolutional layers of the network. The main contributions of this project are as follows.

We fulfil each phase of the project, such as dataset pre-processing, neural network design and training, model evaluation, resultant comparisons. This project provides a large number of original images and annotated images of the Auckland traffic environment for neural network training, including pixel annotation for image segmentation and rectangular frame annotation for vehicle identification. Furthermore, SegNet shows high-accuracy results with small datasets. Faster R-CNN adopts two sets of convolution units to achieve high-precision detection.

In the vehicle identification, the comparison shows that the 22-layer Faster R-CNN (including the two sets of convolution-ReLU-maxpooling) is better than the network with more or less than 22 layers. The test results were up to 81%, and the average error rate was as low as 0.4. In comparison with several other different types of models and models of different basic networks of the same category, this model is better in the control of the position and size of the bounding box. Moreover, the rate of detection is also not high. In the semantic segmentation, multiple evaluation indicators are used as the metric. We compare the network performance using VGG16 and VGG19 as the basic model, respectively. The results of IoU show that VGG16 is better at the segmentation of small-sized objects, while VGG19 prefers large-sized objects.

After measuring all the evaluation indicators in general, we choose VGG19 as the basic model of our segmentation model. Using multiple assessment methods as criteria, IoU was exploited as the evaluation criteria in the evaluation of each class. The segmentation of vehicles, roads, sky, buslane, and buildings has eminent performance. In

general, the proposed model has 76% and 63% GlobalAccuracy and WeightedIoU on this dataset, respectively, which indicates that the model performs well on pixel classification.

Through the exploration of vehicle detection and semantic segmentation, we find that deep learning has a great positive effect on scene understanding. It relies on a layered processing mechanism, from abstract to specific feature analysis and powerful feature transfer capabilities to improve the performance of the model in scene understanding.

6.2 Limitations

The proposed algorithms have been implemented successfully in this thesis for vehicle-related scene understanding. However, there are still some limitations that should be improved in the future.

6.2.1 Limitations of Semantic Segmentation

(1) In this project, our labelled images are not enough, it may affect our scene understanding, in future, we will label more data.

(2) The images of the semantic segmentation are unclear, this may be due to the cameras which are used for image capturing.

(3) In semantic segmentation, out segmentation classes contain only eleven categories. As a consequence, the classifications are not detailed. For example, the left turn sign, the right turn sign, and the straight sign are all covered in the turn sign. In the scene understanding, the unspecified classification may lead to uncertainty of decision making.

(4) The experimental results show that there are satisfactory precisions in the categories of the sky, buildings, vehicles, buslane and roads, but the segmentation results e.g., turnsign, pedestrian, tree, and lane, are not acceptable and require further

improvement. Although these categories are usually very small, traffic signs, pedestrians, and lanes are the focus of analysis in these scenes.

6.2.2 Limitations of Vehicle Detection

(1) Similar to semantic segmentation, the dataset of annotations is insufficient due to our limited footages.

(2) Due to the complexity of traffic scenes, even a tiny object can lead to traffic accidents. As a consequence, scene understanding has high requirements for image clarity. The image pixels in this project are responsible for the display of the distant vehicles, which is one of the reasons that affect the accuracy of segmentation and recognition.

(3) The detector can only detect if there is a vehicle in a particular region, but the type of vehicle cannot be identified. This may cause automatic vehicle to make false alarms because it has not sufficiently object characteristics.

(4) In this project, the ability of our detectors to discern the mixed objects is weak, which may affect the precision of our detection and segmentation.

6.3 Future Work

For the sake of understanding scene in different environments in the task of semantic segmentation, we will collect more datasets from different weather and locations in Auckland while using more data augmentation methods to expand the dataset. Furthermore, we will acquire more high-quality images as training dataset with more advanced equipment. It provides assistance for capturing more detailed features of the object during scene understanding.

In terms of function, more detailed classes will be added, such as isolation bands, rivers, overpasses, and more detailed traffic sign classifications. At the same time, further adjustment of weights and other parameters will improve the accuracy of extremely small

categories such as traffic signs, pedestrians, trees, and lanes.

For the task of vehicle detection, we will expand our dataset to include more vehicles in the Auckland traffic scenes. We will also apply the fine-tuning to the neural networks so as to receive larger images so that the network is able to learn features more accurately. It is also necessary to add a classification function to the model so that it can detect not only the vehicle but also the type of the vehicle.

Moreover, in order to make fully use of deep learning to implement the characteristics of multitasks on a model, in the future, the function of the model will be expanded so that the model detects vehicles as well as other classes of objects in the vehicle-related traffic scene. Finally, in order to detect the vehicle more accurately, the dataset with the interference object should be added more, the anti-interference ability of the vehicle detector is improved, the false detection rate of the detector will be further reduced.

References

Albawi, Saad, Abed Mohammed & Tareq (2017). Understanding of a Convolutional Neural Network. *International Conference on Engineering and Technology (ICET)*, pp. 1-6.

Alex Krizhevsky, Ilya Sutskever & Geoffrey E. Hinton (2012). ImageNet classification with deep convolutional neural networks. *Communications of the ACM* 60 (6): 84–90.

Arunmozhi, A. & Park, J. (2018). Comparison of HOG, LBP and Haar-Like Features for On-Road Vehicle Detection. *IEEE International Conference on Electro/Information Technology (EIT),* pp. 362-367.

Badrinarayanan, A., Kendall, A. & Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* (12), pp. 2481.

Bengio, Y., 2009. Learning deep architectures for AI. Foundations and trends in Machine Learning, 2(1), pp.1-127.

Chen, H., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 4(1), pp. 834.

Dawar, L., Ostadabbas, S., & Kehtarnavaz, N. (2019). Data Augmentation in Deep Learning-Based Fusion of Depth and Inertial Sensing for Action Recognition. *IEEE Sensors Letters,* 4(1), pp. 1.

Deng, L., Yang, M., Qian, Y., Wang, C. & Wang, B. (2017). CNN based semantic segmentation for urban traffic scenes using fisheye camera. *IEEE Intelligent*

Vehicles Symposium, pp. 231–236.

Ding, L., Zhao, K., Zhang, X., Wang, X., & Zhang, J. (2019). A Lightweight U-Net Architecture Multi-Scale Convolutional Network for Pediatric Hand Bone Segmentation in X-Ray Image. *IEEE Access, Access,* pp. 68436.

Dreyfus, Stuart E. (1990). Artificial neural networks, back propagation, and the Kelley-Bryson gradient procedure. *Journal of Guidance, Control, and Dynamics*. 13 (5): 926–928.

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. *Proceedings of the 31st International Conference on Machine Learning*, 32(1):647-655

Estrach, J.B., Szlam A. and LeCun, Y. (2014). Signal recovery from pooling representations. In International Conference on Machine Learning (Vol. 2, pp. 1585-1598).

Farabet, B., Couprie, C., Najman, L. & LeCun, Y. (2013). Learning Hierarchical Features for Scene Labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915-1929.

Fawzi, A. & Frossard, P. (2015). Manitest: Are classifiers really invariant? In British Machine Vision Conference (pp. 106.1-106.13).

Fawzi, A., Samulowitz, H., Turaga, D. & Frossard, P. (2016). Adaptive data augmentation for image classification. *IEEE International Conference on Image Processing (ICIP)*, pp. 3688-3692.

Felzenszwalb, P., McAllester, D. & Ramanan, D.(2008). A discriminatively trained, multiscale, deformable part model. *IEEE Conference on Computer Vision and Pattern Recognition Computer Vision and Pattern Recognition*, pp. 1.

Jeong, H., Choi, S., Jang, S. & Ha, Y. (2019). Driving Scene Understanding Using Hybrid Deep Neural Network. *IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 1-4.

Jian-Huang Lai et al., (2018). Pattern Recognition and Computer Vision. *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pp. 517-518

Jonathan Long, Evan Shelhamer & Trevor Darrell. (2015). Fully Convolutional Networks for Semantic Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431-3440.

Jude Hemanth, Madhulika Bhatia, & Oana Geman (2019). Data Visualization and Knowledge Engineering. Springer Nature Switzerland, vol.32, pp. 79-107

Fukushima, K. (1980). Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biological Cybernetics*, (36), pp.193-202.

Jin, Y., Li, J., Ma, D., Guo, X. & Yu, H.(2017). A Semi-Automatic Annotation Technology for Traffic Scene Image Labeling Based on Deep Learning Preprocessing. *IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, pp. 315-320.

Gazzah, R., Mhalla, A. and Essoukri. N. & Ben A.(2016). Vehicle detection on a video traffic scene: Review and new perspectives. *International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, pp. 448-454.

Geng, H., Guan, J., Pan, H. & Fu, H. (2018). Multiple Vehicle Detection with Different Scales in Urban Surveillance Video. *IEEE Fourth International Conference on Multimedia Big Data (BigMM)*, pp. 1-4.

Ghosh, T., Li, L., & Chakareski, J.(2018). Effective Deep Learning for Semantic Segmentation Based Bleeding Zone Detection in Capsule Endoscopy Images. *IEEE International Conference on Image Processing (ICIP)*, pp. 3034-3038.

Gindele, Brechtel, Dillmann. (2010) A probabilistic model for estimating driver behaviours and vehicle trajectories in traffic environments. *International Conference on Intelligent Transportation Systems*, pp. 1625-1631.

Girshick, Ross. (2015). Fast-RCNN. *IEEE International Conference on Computer Vision (ICCV)*, pp. 1440-1448.

Girshick, R., Donahue, J., Darrell, T. & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580-587.

Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep learning. *The MIT Press*. pp.44-56

Hao Qu, Lilian Zhang, Xuesong Wu, Xiaofeng He, Xiaoping Hu, and Xudong Wen. (2019). Multiscale Object Detection in Infrared Streetscape Images Based on Deep Learning and Instance Level Data Augmentation. *Applied Sciences*, pp.553-565.

He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778.

Hinton, E. & Salakhutdinov, R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science,* vol. 313, pp. 504.

Hinton G.E., Krizhevsky A., & Wang S.D. (2011). Transforming Auto-Encoders. *International Conference on Artificial Neural Networks (ICANN)*, pp. 44-51

Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, Thomas Blaschke, Chen, H., Engkvist, O., Wang, Y., Olivecrona, M., & Blaschke, T. (2019). The rise of deep learning in drug discovery. *Drug Discovery Today*, 23(6), 1241–1250.

Hornik K, Stinchcombe M, White H. (2019). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366.

Hsu, R., Huang, C. & Chuang, C. (2018). Vehicle detection using simplified Fast R-CNN. *International Workshop on Advanced Image Technology (IWAIT)*, pp. 1-3.

Hubel D H, Wiesel T N. (2019). Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160(1), 106–154.

Husain, Farzad, Dellen, Babette & Torras, Carme. (2017). Scene Understanding Using Deep Learning. *Academic Press*, pp. 373-382.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. (2015). Learning Deconvolution Network for Semantic Segmentation. *IEEE International Conference on Computer Vision (ICCV)*, pp. 1520-1528.

Ioffe, Sergey & Szegedy, Christian. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *International Conference on Machine Learning*, pp. 448-456

Kavukcuoglu, Koray & Sermanet, Pierre & Boureau, Y-Lan & Gregor, Karol & Mathieu, Michaël & Lecun, Yann. (2010). *Learning Convolutional Feature Hierarchies for Visual Recognition. Neural Information Processing Systems,* pp.1090-1098.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90.

Lai, Z., Chou, Y. & Schumann, T. (2017) Vehicle detection for forward collision warning system based on a cascade classifier using adaBoost algorithm. *IEEE*

7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin), pp. 47-48.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), pp.2278-2324.

Lee, A. W. C. & Yong, S. (2018). Localized Object Information from Detected Objects Based on Deep Learning in Video Scene. *IEEE Conference on Systems, Process and Control (ICSPC)*, pp. 100-105.

Lin, G., Milan, A., Shen, C., & Reid, I.D. (2016). RefineNet: Multi-path Refinement Networks for High-Resolution Semantic Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.5168-5177.

Li, B., Shi, Y., Qi, Z. & Chen, Z. (2018). A Survey on Semantic Segmentation. *IEEE International Conference on Data Mining Workshops (ICDMW),* pp. 1233-1240.

Li Yandong, Hao Zongbo & Lei Hang. (2016). Survey of convolutional neural network. Journal of Computer Applications, 36(9), pp.2508-2515.

Liu, B., Zhao, W., & Sun, Q. (2017). Study of object detection based on Faster R-CNN. *Chinese Automation Congress (CAC)*, pp. 6233-6236.

Long, J., Shelhamer, E. & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431-3440.

Melih, A. & Celenk, M. (2017) Road scene content analysis for driver assistance and autonomous driving. IEEE Trans. Intell. Transp. Syst., vol. 18, no. 12, pp. 3398–3407.

Mikolov, T., Corrado, G., Chen, K. & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on*

Learning Representations, pp.1-12.

Min Lin, Qiang Chen, & Shuicheng Yan. (2014). Network in network. *In ICLR* (Vol. 3, pp.10)

Mittal, A., Hooda, R. & Sofat, S. (2018). Wireless Personal Communications. Springer US, vol. 101, pp. 511-529

Mohamed, Hossam, Ahmed & Sherif. (2018). A Deep Learning Approach for Vehicle Detection. *International Conference on Computer Engineering and Systems (ICCES)*, pp. 98-102.

Nagi. J. et al., (2011). Max-pooling convolutional neural networks for vision-based hand gesture recognition. *IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, pp. 342-347.

Newton, A., Pasupathy, R. & Yousefian, F. (2018). Recent Trends in Stochastics Gradient Descent for Machine Learning and Big Data. *Winter Simulation Conference (WSC)*, pp. 366-380.

Nikita Dvornik, Konstantin Shmelkov, Julien Mairal, Cordelia Schmid.(2017). BlitzNet: A Real-Time Deep Network for Scene Understanding. *The IEEE International Conference on Computer Vision (ICCV)*, pp. 4154-4162

Noh, H., Hong, S., & Han, B. (2015). Learning Deconvolution Network for Semantic Segmentation. *IEEE International Conference on Computer Vision (ICCV)*, pp. 1520-1528.

Oeljeklaus, M., Hoffmann, F. & Bertram, T. (2017). A combined recognition and segmentation model for urban traffic scene understanding. *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1-6.

Park, H., Jang, S., Jeong, H. & Ha, Y. (2019). Roadway Image Preprocessing for Deep

Learning-Based Driving Scene Understanding. *IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 1-4.

Paulin, K., Revaud, J., Harchaoui, Z., Perronnin, F., & Schmid, C. (2014). Transformation pursuit for image classification. *IEEE Conference on ComputerVision and Pattern Recognition (CVPR)*, pp. 3646-3653.

Pedro O. Pinheiro, Ronan Collobert, & Piotr Dollár. (2015). Learning to segment object candidates. *Advances in Neural Information Processing Systems*, pp. 1990-1998.

Peixinho, A. Z., Benato, B. C., Nonato, L. G. & Falcão, A. X. (2018). Delaunay Triangulation Data Augmentation Guided by Visual Analytics for Deep Learning. *SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 384-391.

Pinheiro, P. O., Lin, T. Y., Collobert, R., & Dollár, P. (2017). SharpMask: Learning to refine object segments. European Conference on Computer Vision (pp. 75-91).

Ren, S., He, K., Girshick, R. & Sun, J. (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149.

Robert, K. (2019). Video-based traffic monitoring at day and night vehicle features detection tracking. *IEEE Conference on Intelligent Transportation Systems*, pp. 1-6.

Ronneberger, O., Fischer, P. & Brox. T. (2019) U-Net: Convolutional Networks for Biomedical Image Segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention,* pp.234-241

Samui, P., Roy, S. S. & Balas, V. E. (2017). Handbook of neural computation. *Academic Press*, pp. 12-34

Sara Sabour, Nicholas Frosst, & Geoffrey E. Hinton (2017). Dynamic routing between capsules. *International Conference on Neural Information Processing Systems*, pp. 3859-3869.

Shelhamer, E., Long, J., & Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 4, pp. 640-651.

Shen, X., Hertzmann, A., Jia, J., Paris, S., Price, B., Shechtman, E., & Sachs, I. (2016) Automatic Portrait Segmentation for Image Stylization. Computer Graphics Forum, 35(2), 93–102.

Shih, J., Chiu, C. & Pu, Y. (2019). Real-time Object Detection via Pruning and a Concatenated Multi-feature Assisted Region Proposal Network. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1398-1402.

Shrestha, A. & Mahmood, A. (2019). Review of Deep Learning Algorithms and Architectures. *IEEE Access*, vol. 7, pp. 53040-53065.

Shuhan, B., Ben, W., Jindong, L.& Xuelong, H. (2017). Semantic image segmentation using region-based object detector. *IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, pp. 505-510.

Srivastava, J. et al., (2014). Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929-1958.

Stivaktakis, R., Tsagkatakis, G., & Tsakalides, P. (2019). Deep Learning for Multilabel Land Cover Scene Categorization Using Data Augmentation. IEEE Geoscience and Remote Sensing Letters, vol. 16, no. 7, pp. 1031-1035.

Szegedy, A., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A.(2014). Going deeper with convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1-9

Szegedy, A., Vanhoucke,V., Ioffe, S., Shlens, J. & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818-2826.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A. (2015). Going deeper with convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1-9.

Szegedy, C., Ioffe, S., & Vanhoucke, V. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *AAAI Conference on Artificial Intelligence*, pp. 4278-4284

Tang, Cong, Yongshun Ling, Xing Yang, Wei Jin, & Chao Zheng. (2019). Multi-View Object Detection Based on Deep Learning. *Applied Sciences-Basel*, 8(9), pp.1423.

Tang, J., Li, j., & Xu, X. (2018). Segnet-based gland segmentation from colon cancer histology images. *Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 1078-1082.

Tang W., Zou D., Yang S., & Shi J. (2018). DSL: Automatic Liver Segmentation with Faster R-CNN and DeepLab. *International Conference on Artificial Neural Networks*, pp 137-147

Tran, S., Kwon, O., Kwon, K., Lee, S. & Kang, K. (2018). Blood Cell Images Segmentation using Deep Learning Semantic Segmentation. *IEEE International Conference on Electronics and Communication Engineering (ICECE)*, pp. 13-16.

Valliappan, C., Kumar, A., Mannem, R., Karthik, G., & Ghosh, P. K. (2019). An Improved Air Tissue Boundary Segmentation Technique for Real Time Magnetic Resonance Imaging Video Using SegNet. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5921-5925.

Wang, J & Lai, S. (2019). Object Detection in Curved Space for 360-Degree Camera. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3642-3646.

Wang, J., Wang, L., Lu, H., Zhang, P. & Ruan, X. (2019). Salient Object Detection with Recurrent Fully Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1734-1746.

Wang, H., Yu, Y., Cai, Y., Chen, X., Chen, L. & Liu, Q. (2019). A Comparative Study of State-of-the-Art Deep Learning Algorithms for Vehicle Detection. *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 2, pp. 82-95.

Wei, Y., Tian, Q., Guo, J., Huang, W., & Cao, J. (2017). Multi-vehicle detection algorithm through combining Harr and HOG features. *Mathematics and Computers in Simulation*, pp. 130–145.

Williams, J., Shawe-Taylor, R., Zemel, S., & Culotta, A. (2015). Scene analysis by mid-level attribute learning using 2D LSTM networks and an application to web-image tagging. *Patten Recognition Letters*, pp. 23–29.

Xie, S. & Tu, Z. (2015). Holistically-Nested Edge Detection. *IEEE International Conference on Computer Vision (ICCV*), pp. 1395-1403.

Xiushen Wei. (2018). Analytical Deep Learning: Convolutional Neural Network Principles and Visual Practice. *Electronic Industry Press,* pp. 29

Yamashita, R., Nishio, M., Do, R.K.G. et al. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, pp. 611.

Yan, W. Q. (2019). Introduction to intelligent surveillance: surveillance data capture, transmission, and analytics. *Springer*, pp. 1-6.

Yang, S., Wang, W., Liu, C., & Deng, W. (2019). Scene Understanding in Deep

Learning-Based End-to-End Controllers for Autonomous Vehicles. *in IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 53-63

Yao, W., Zeng,Q., Lin, Y., Guillemard, F., Geronimi, S., Aioun, F. (2017). On-road vehicle trajectory collection and scene-based lane change analysis. *IEEE Transactions on Intelligent Transportation Systems*, vol.18, no.1, pp.206-220.

Yu, Y. & Cao, Kai. (2015). A Method for Semantic Representation of Dynamic Events in Traffic Scenes. *Information and Control*, 44(1): 83-90.

Zamir A. R. et al., (2017). Feedback Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1808-1817.

Zhang, R., Tian, L., Li, C. & Li, H. (2018). A SSD-based Crowded Pedestrian Detection Method. *International Conference on Control, Automation and Information Sciences (ICCAIS)*, pp. 222-226.

Zhang, R., Li, M., & Wang. L. (2019). Fusion of Images and Point Clouds for the Semantic Segmentation of Large-Scale 3D Scenes Based on Deep Learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, pp.85–96.

Zhang, Y., Li, J., Guo, Y., Xu, C., Bao, C., & Song, Y. (2019). Vehicle Driving Behavior Recognition Based on Multi-View Convolutional Neural Network with Joint Data Augmentation. *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4223-4234.

Zhao, W., Fu, Y., Wei, X., & Wang, H.(2018). An Improved Image Semantic Segmentation Method Based on Superpixels and Conditional Random Fields. *Appl. Sci*, 8(5), p.837.

Zheng, S. et al., (2015). Conditional Random Fields as Recurrent Neural Networks. *IEEE International Conference on Computer Vision (ICCV)*, pp. 1529-1537.

Zhou, Wujie, Lv, Sijia, Qiuping, Jiang, Yu & Lu. (2019). Deep Road Scene Understanding. *IEEE Signal Processing Letters*, vol. 26, no. 4, pp. 587-591.

Zitnick, C. Lawrence, & P. Dollar. (2014). Edge boxes: Locating object proposals from edges. *ECCV European Conference*, Part V, pp. 391.

Zoumpourlis, G., Doumanoglou, A., Vretos, N., & Daras, P. (2017). Non-linear Convolution Filters for CNN-Based Learning. *IEEE International Conference on Computer Vision (ICCV)*, pp.4771-4779.