MINISTRY OF EDUCATION AND TRAINING

DUY TAN UNIVERSITY

ADAPTIVE LEARNING SOLUTION BASED ON DEEP LEARNING FOR TRAFFIC OBJECT RECOGNITION

DOCTOR OF PHILOSOPHY OF COMPUTER SCIENCE

Da Nang, 2022

MINISTRY OF EDUCATION AND TRAINING
DUY TAN UNIVERSITY

ADAPTIVE LEARNING SOLUTION BASED ON DEEP LEARNING FOR TRAFFIC OBJECT RECOGNITION

Major: Computer Science

Code: 9480101

Da Nang, 2022

COMMITMENT

To the best of my knowledge, I hereby certify that all the content in the thesis entitled "Adaptive learning solution based on deep learning for traffic object recognition" is my own research. The figures and results of the thesis are honest, fully quoted and have not been previously published by another.

The author's signature

ACKNOWLEDGEMENTS

First of all, I would like to express my endless thanks to my instructors. Their kindly support and advices went through the completion process of my PhD thesis. Their companion encouraged me to improve my work. Their instructions and motivation helped me to grow as a research scientist.

I would also like to thank my council reviewers, members and independent scientists for giving me contribution and brilliant comments to my thesis.

I would like to express my sincere thanks to the Board of Trustees and Board of Rector of Duy Tan University, the teachers and officers of Duy Tan University's Graduate School, for helping me in the process of learning and researching at University.

I also acknowledge my thankfulness to the Board of Directors of the Quang Binh provincial Department of Information and Communications for kind assistances and support in my work and learning so that I can achieve the results today.

Many thanks come to the research group’s members for their participation in the published works and allowing me to use the research results for this thesis.

Finally, my deeply thanks come to my loved people and friends who were always beside me to help me when I need for the last time. A special thanks to my family where I got the most assistances and motivation for the whole of my life.

In spite of the fact that many efforts are made during the working process, the thesis may remain shortcomings due to limited time and research conditions. All valuable comments and suggestions for the thesis completion will be highly appreciated.

The author

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Explanation
AI	Artificial Intelligence
HAC	Human Action Recognition
ML	Machine Learning
DL	Deep Learning
AL	Adaptive Learning
CNN	Convolution Neural Network
NN	Neural Network
DNN	Deep Neural Network
ANN	Artificial Neural Network
SVM	Support Vector Machine
RF	Random Forest
ACF	Aggregate Channel Features
ITS	Intelligent Transportation Systems
ROI	Region of interest
SaaS	Software-as-a-Service
ADAS	Advanced Driver Assistance Systems
HOG	Histograms of Oriented Gradients
AV	Auto Vehicle

INTRODUCTION

1. Introduction

Artificial intelligence (AI) is intelligence demonstrated by an artificial system. Artificial intelligence is everywhere today such as office applications, automatic answering systems, intelligent traffic management, smart home management, etc. Since the Computer hardware systems became increasingly capable, artificial intelligence has made great progress, applied more widely in all fields of life and society.

Artificial intelligence focuses on developing algorithms and applications that support human in decision making or self- decision making in the process of data identifying and acquiring. Object detection, Object action recognition and Human action recognition are one of the research targeted directions such as security surveillance systems, security, manual remote control systems, blind assist systems, sports data analysis systems, automated robots, self-driving cars [1, 2, 3, 4, 5], and so on. There have been many studies proposing many different solutions to artificial intelligence development such as heuristic algorithm, evolution algorithm, Support Vector Machine algorithm, Hidden Markov Model algorithm, expert method, neural network method, [6, 7, 8], etc. Traditional solutions, yet all require human intervention and huge amounts of data to analyze and store but low accuracy and limited identification cases.

To overcome those shortcomings, machine learning with focusing on Deep Learning Method (Deep Learning) is now being applied in artificial intelligence in terms of object detection and action recognition.

Deep Learning has been a hotly debated AI topic. As a small category of machine learning, Deep Learning focuses on solving issues related to artificial neural networks in order to upgrade technologies such as voice recognition, image recognition and natural language processing. In just a few years, Deep Learning has promoted progress in a variety of fields which are used to be very difficult to

artificial intelligence researchers such as Object Perception, Machine Translation, voice recognition, etc.

However, despite of the fact that issues related to AI were solved, Deep Learning has still remained limitations that need to be settled.

- Firstly, to create a system capable of identifying a variety of objects, a huge amount of input data is required by Deep Learning to enable computers to learn. This process takes time with assistance of an extremely large processor which can be only processed by a large server system.

- Secondly, Deep Learning is still unable to recognize complex things like common social contacts. It, also, has trouble with detecting similar things because of having no technology good enough helping artificial intelligence to draw those recognition logically. Besides, integration of abstract knowledge into machine learning systems seem to be the challenging issues, such as information about what object is, what it is used for, how people use it, so on. In other words, machine learning has not acquired the usual knowledge like human yet.

The question is “How can a machine learning system learn the knowledge, select and update appropriate knowledge and then build a binding, stringed data set like human by itself?”. Research on Adaptive Learning [9, 10, 11, 12, 13, 14] can be a solution to improve Deep Learning' limitations, exploring issues that Deep Leaning has not been able to do.

A comprehensive Adaptive Learning model will make an auto robot system being capable of self-learning and self-intelligence that emulate the way the human brain work. Under the device’s operation, the intelligence of the system will increase over time. Accordingly, appropriate data will be automatically selected by the system with its retraining of the model and replacing of the old model

The proposed Adaptive Learning model could be promisingly applied in many different Auto Robot systems. Yet, in this research of a doctoral thesis, studying and experiment will be conducted on self-driving vehicles to simulate an

operation process of an auto robot. Recognition objects of self-driving vehicles include objects in traffic such as other vehicles (motorcycles, cars, trucks, passenger cars, etc.), pedestrians, traffic signs, roadbed, roadside, etc.

2. Research goal

The thesis goal is to study on artificial intelligence, the methods and algorithms that have been applied, evaluating the limitations of the current methods in order to propose improved solutions, enhancing the efficiency and accuracy of AI applied in object detection.

- Study, analyze and evaluate traditional methods: Support Vector Machine, Hidden Markov Model, Neural network, and so on.

- Study and evaluate the application of Deep Learning in classification and object detection in traffic (Pedestrians, traffic vehicles, traffic signs, etc.).

- Propose solutions to enhance the performance capacity of the Deep Learning model based on Adaptive Learning approach with conducting experiments of Adaptive learning and hyperparameters on self-driving vehicle (ADAS).

- Develop data sets for training and recognizing objects in traffic.

3. Research method

- Method of information collection: Collecting overview materials of basic foundation algorithm and AI, documents and articles about Deep Learning, Adaptive Learning and object detection. The experimental data were collected from real-time traffic cameras and from the videos on the internet.

-Comparison method: Summary and comparison between the obtained documents to provide an overview of the methods, advantages and disadvantages of those methods as well.

- Analysis method: Analyze the algorithms, their operation and characteristics. The effectiveness of the algorithms applied to specific cases is evaluated and analyzed to get the best results.

- Expert method: Consult from AI experts to complete the area need to be studied.

- Experimental method: Installing and testing algorithms applied to each method for a better understanding. From this, the advantages and disadvantages of each method are then evaluated and verified.

- Conduct experiments on Google's machine learning open-source system (TensorFlow), MathWorks (Matlab) to have comparison with the results of research experiments.

- Collect and establish real empirical data sets (Objects in traffic: pedestrians, vehicles, traffic signs, etc..) which are used for training and testing the proposed algorithms. Data sets of images are collected from actual photos on road or from videos on the internet.

- Install research results on the system to prove experiment.

4. Research subject and scope

- *Research subject*

+ Deep Learning method

+ Adaptive Learning method

- *Research scope*

+ Some machine learning methods.

+ Deep Learning method and Adaptive Learning method

+ Propose solutions to enhance on-road object detection quality of self-driving car system.

+ Study and propose Adaptive Learning solution which is applied in on-road object detection.

+ Create data and experiment, analyze results.

5. The structure of the thesis

Chapter and title	Relevant scientific publications	Relevant scientific Contribution
Chapter 1: Overview of artificial intelligence An overview of artificial intelligence and traditional algorithms includes decision tree, random forest, Support vector machine, and Artificial neural network. Domestic and international research on on-road object detection and Adaptive Learning solution for self-driving vehicle systems.		None
Chapter 2: Identifying objects by Deep Learning Proposes solution to on-road object detection by Deep Learning: pedestrians, vehicles	**PP 1.1** **PP 1.2** **PP 1.3**	- Deep Learning in pedestrian action prediction - Deep Learning in vehicle classification
Chapter 3: Developing Adaptive Learning techniques in object recognition Basing on the research results stated in Chapter 2, the Adaptive Learning solution of self-driving vehicle system data is continuously proposed. The proposed model is capable of self-learning and self-intelligence without any human intervention	**PP 1.4**	Adaptive learning techniques in vehicle, traffic sign recognition and advanced driver assistance systems
Chapter 4: Optimization of hyperparameter set in Adaptive Learning Basing on the proposed model mentioned in Chapter 3, the Adaptive Learning solution of algorithms and parameters is continuously studied, improving	**PP 1.5**	Adaptive learning through optimization of the training hyperparameter set based on a new dataset related to traffic sign and vehicle recognition

Chapter and title	Relevant scientific publications	Relevant scientific Contribution
efficiency and on-road object detection accuracy.		

CHAPTER 1. OVERVIEW OF ARTIFICIAL INTELLIGENCE

In this chapter, we investigate overview of artificial intelligence and traditional algorithms includes decision tree, random forest, Support vector machine, and Artificial neural network. Domestic and international research on on-road object detection and Adaptive Learning solution for self-driving vehicle systems.

1.1 Overview of artificial intelligence

1.1.1. Definition of artificial intelligence

There have been many different definitions of artificial intelligence, or AI in the world, specifically:

• By popular, artificial intelligence is is intelligence demonstrated by any artificial system. The term is often used to refer to computers with unspecified purpose and the science of theories and applications of artificial intelligence.

• According to Bellman, artificial intelligence is the automation of activities that we associate with human thinking, activities such as decision-making, problem solving, learning, etc.

• Rich and Knight: “Artificial intelligence is the study of how to make computers do things at which, at the moment, people are better".

Historically, each definition has its own right, but for simplicity we can get the idea of artificial intelligence as a computer science. It was built on a solid theoretical foundation and can be applied to automation of the intelligent behavior by computers. It makes computers acquire the human intelligence such as thinking, decision-making, problem solving, learning and self-adapting.

1.1.2 History of artificial intelligence

The history of artificial intelligence [15, 16, 17] has gone over many different stages of development, as shown in Figure 1.1.

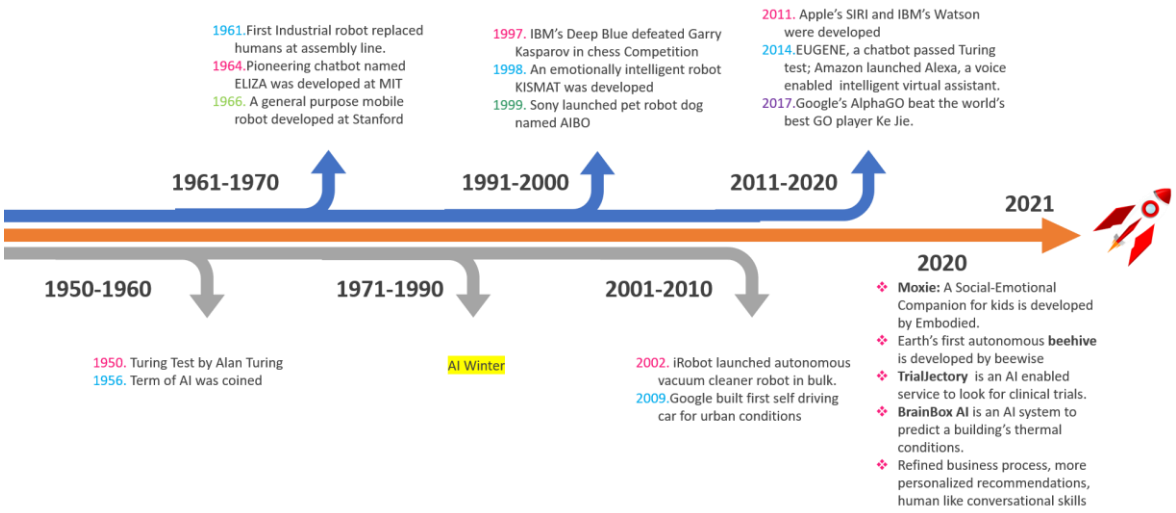


Figure 1.1 History of artificial intelligence (Source: https://connectjaya.com/)

1.2. Machine learning and identification techniques

As an AI subfield, machine learning uses algorithms that enable computers to learn from data to perform tasks instead of being explicitly programmed [18].

1.2.1 Machine learning applications

1.2.1.1 Image processing

Image processing problem solve issues of analyzing information from images or performing some transformations. Some examples are:

- Image tagging, like Facebook, an algorithm that automatically detects your face and your friends' photos. Basically, this algorithm learns from photos you've tagged yourself before.

- Optical Character Recognition (OCR) is an electronic conversion of the typed, handwritten or printed text images into machine-encoded text. The algorithm must learn to recognize what the snapshot of a character is.

- Self-driving cars, part of the mechanism used here is image processing. A machine learning algorithm enables self-driving cars to detect road edges, signs or obstacles by looking at each video frame from the camera.

1.2.1.2 Text analysis

Text analysis is a work of transforming or classifying free texts. The texts here can be Facebook posts, emails, chats, documents, etc. Some common examples are:

- Spam filtering is one of the most popular spam text classification applications. Text classification, here, is to identify the subject definition of a text. The spam filter can also “learn” what each user views as spam based on the user identifying email message and its subject.

- Sentiment Analysis learns how to classify an expression as positive**,** negative, or neutral

- Information Extraction is the process of extracting information from textual sources, learn how to useful information, address, a person's name, a keyword, etc. for ex.

1.2.1.3 Data mining

Data mining is a process of discovering valuable information or making predictions on sets. Each record is an object to learn and each column is a feature. The value of a column of new record can be predicted based on the learned records or the records can be grouped. Data mining applications are:

- Anomaly detection is a technique for finding an unusual point, credit card fraud detection or, for example. A suspicious transaction may be discovered based on a change in consumer normal behavior.

- Association rules, for example, in a supermarket or on an e-commerce site. It can be found which items customers often buy together. In other words, which item does your customer usually buy next when buying item? Such information can be used as the basis for decisions about marketing activities

- Grouping, for example, in a SaaS platform, users are grouped by their behavior or by profile information.

- Predictions**,** the value columns (of a new record in the database). For example, the price of an apartment can be predicted based on the previous price data.

1.2.1.4. Video games and robotics

Video games and robotics are a big field where machine learning made its contribution to. If a character moves and needs to avoid obstacles in the game, machine learning can learn and does this task by using Reinforcement learning. Accordingly, reinforcement learning by machine aims at solving the above task. Reinforcement learning is negative if colliding obstacles. It is positive if reaching the destination.

1.2.2 Basic recognition techniques in machine learning

The ability to apply AI methods combined with image processing to object recognition is one of the most important issues of computer vision. machine learning technique is divided into two types that are supervised machine learning and unsupervised machine learning. The supervised machine learning techniques include decision trees, neural network, SVM, boosting, random forests, etc. Under the supervised machine learning, the classification is usually based on a sample dataset to be labeled in layers by "experts" to analyze and develop recognition model. The sample data set to be learned is called the training dataset. The analyzing and developing process of object is called a training process of recognition machine or model training. In contrast, under unsupervised method unlabeled data, algorithm itself do its classification and against data which is not labeled. Its identification of object layer is based on analysis and statistics from the input data set.

1.2.2.1 Decision tree

Decision trees are a specific field of research in machine learning. Decision tree techniques are widely used in the fields of knowledge exploitation and pattern recognition [19]. A decision tree is a predictive model, developed on a tree structure, used to layer data samples based on a series of rules. In these tree

structures, leaves represent classification decisions and branches represent conjunctions of features that lead to those classification decisions.

A decision tree can be trained by dividing the training data set into subsets for test of a single attribute value or a group of attributes. Classification can be described as simple classification combinations by using mathematical deductive techniques. The training of classification model is the development process of a decision tree.

1.2.2.2 Random forests

Random forests (RF)operate by constructing a multitude of decision trees with random selection of features. Random sub-multitude selection of features is not necessarily separate. Thus, selection of features and tree construction are made by the random algorithm. Random forests was created by Tin Kam Ho [20] in 1998 published on IEEE Journal.. Similarly, random forests are also a form of supervised algorithm. RF algorithm can be used for both classification and regression problems, capable of handling problems with lack of value. More trees in forests enable the problem with over-fitting data to be solved. Random forest techniques are widely used in the field of computer vision and object classification.

1.2.2.3 Boosting technique

Boosting technique is a machine learning ensemble algorithm by constructing multiple classifiers at the same time which can, then, be combined by weight. Each component classifier is called the weak classifier. Weak classifiers are combined to generate a one strong classifier. AdaBoost (Adaptive boosting) proposed by Freund and Schapire[21] in 1999 is one of the popular boosting algorithms. AdaBoosting is a nonlinear strong classifier, which works on the principle of weak classifiers combination by weights to generate a stronger classifier of adaptive type (with data samples).Accordingly, AdaBoost uses weights to mark difficult-to-categorize patterns. Meanwhile the easy classifications contain a smaller impact value, the deeper the levels of the weak classification are, the more the classifier focuses on difficult-to-categorize samples. That is, during the training,

each weak classifier updates its weight in the direction of decreasing the weight of the correct classification samples (easy samples) and increasing the weight of the weak classifications thereafter. Based on this idea, the later classifiers can handle primarily on difficult samples that the previous classifier can't. Finally, weak classifiers are combined by their weight depending on classification accuracy to generate a final strong classifier.

1.2.2.4 Support vector machine

The support-vector machine (SVM) is a supervised learning algorithm which is proposed by Corinna and Vapnik [22] in 1995. SVM was first designed for classification problem, expanded for application of various multilayer classifications [23, 24]later. The SVM algorithm conducts training to develop model for data sample classification by classes for each training data sample set of two predefined types. An SVM model is a representation of vectors that support classification in multi-dimensional space and the choice of hyperplane to classify between two classes so that the maximum distance from the training data samples (points in n-dimensional space) to the classification plane (Figure 1.2).Samples for classification must be represented in the same space and SVM is classified into one of two classes depending on the specific value of the data sample on which side of the classification super plane.

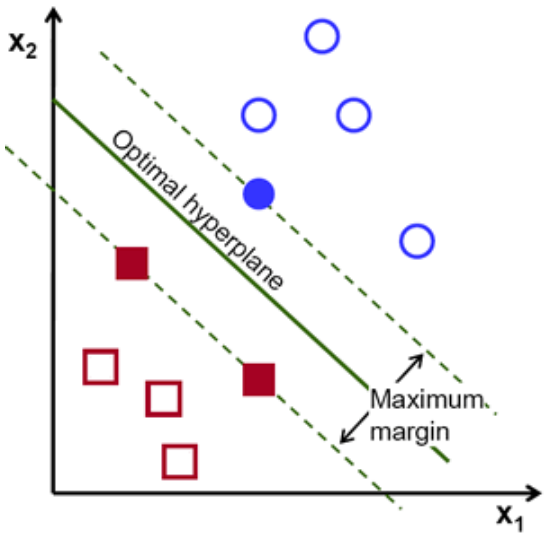


Figure 1.2 Classification simulation of SVM (Source: https://towardsai.net)

Up to now, SVM is one of the most widely used classification methods in the field of computer science and data analysis. SVM works effectively on large data sets and in a large number of dimensions, especially applied to classifications of image data, text, voice, etc. SVM is capable of being applied to many different kernel functions and can be classified by linear or non-linear methods. In fact, SVM reaches to a quietly high accuracy compared to other traditional machine learning techniques.

1.2.2.5 Artificial neural network

Artificial neural network (ANN) is often referred to as neural network. A neural network is vaguely inspired by a biological neural network. ANN architecture consists of the nodes (called neutral) and a set of arcs (called edges)(Figure 1.3). Set of the connections is organized into layers, including input layer, output layer. In between them are *hidden layers*. Each arc connects two pairs of neural including an input and an output to transmit information and process new values for output. The *propagation function*, with its corresponding weight set, presents the relationship between nodes. Normally, the NN architecture is developed in advance and weights are then defined during the training. However, some types of networks are capable of adapting to real data and its architecture can be changed by itself thank to information during its learning. Such networks are Multilayer neural network- MLNN and Self organizing maps- SOM.

Capable of self-learning in neural networks is one of the important components of NN [25]. A neural network is not only a complex system but also a complex adaptive one, which means that it is able to change the internal structure based on the information flow given. In particular, these changes are obtained through the weighting adjustment. In Figure 6, each arc represents a connection between the two neurons and indicates the direction of data flow through the network. Each connection has it own a certain weight, which is the value that controls the signal between the two neutrals. If the network gives good classification results, it is not necessary to adjust the weight again. In contract, if the

result is not good enough (at an error above the threshold) the weight must be adjusted to change the output for system's adaption.

Classified by training methods, neural networks can be the form of supervised learning or unsupervised learning, intensive learning. Under a supervised learning approach, training is conducted on labeled data sets (by experts). While under an unsupervised learning one, neural network's processing is done on unlabeled data set. The implementation process is considered as the process of finding potential features in a dataset.

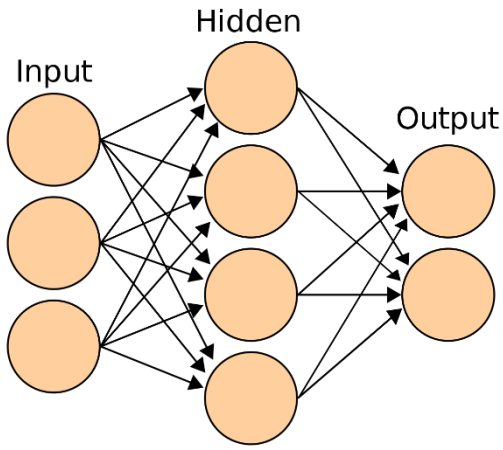


Figure 1.3 Illustration of neural network architecture (Source: https://techvidvan.com)

Neural networks have found applications with great achievement in Artificial intelligence. Pattern recognition is one of the popular applications of NN such as optical character recognition, face identification, body shape identification, object recognition through images, traffic signs identification, unusual action recognition in intelligent surveillance systems, and more. As a result, neural network scan be utilized to generate predictions and signal processing such as stock forecast, weather forecasts, audio filtration systems, noise interference elimination, and amplifying important sounds.

1.3 Deep Learning and Adaptive Learning

1.3.1 Overview of Deep Learning and Adaptive Learning

1.4.1.1 Deep Learning

This is a new study area of computer vision and machine learning. Deep Learning is a collection of algorithms that tend to solve high-level discrete data models by using multi layers or in combination with nonlinear changes with more complicated architecture compared to traditional machine learning. Deep Learning was introduced early in the 1990s with several other machine learning techniques. However, at that time, the method was not effective as expected due to deep architecture and computing complications, and the limits of hardware calculation resources. Lecun [26] was one of the pioneers in studying and proposing solutions in Deep Learning.

Deep Learning concept was first mentioned by Rina Dechter in a paper published in 1986. In 1989, Lecun and colleagues suggested a Deep Learning artificial neural network that used standard backpropagation algorithms to recognize hand writings and got highly accurate outcomes. Lecun's neural network was considered as one of Deep Learning basics for future study and application. Deep Learning neural network is a form of Deep Learning technique, which relates to the study and application of network models that solve sophisticated problems by artificial neural basics. The network is to present and extract features, classify and recognize samples in voice recognition, computer vision, natural language treatment and predictive analytic techniques. Recently, Deep Learning has been a special interest in computer science. The technique has brought optimistic outcomes with higher level of accuracy compared to conventional approaches and promoted other aspects in computer science.

1.3.1.2 Adaptive learning

Adaptive learning has its origins in the needs of building an intelligent system that imitates human brain. Artificial neural systems that have been

consecutively proposed have high level of accuracy and multiple object recognition. They are AlexNet [27], GoogLeNet [28], Microsoft ResNet [29], R-CNN [30], Fast R-CNN [31], Faster R-CNN [32] and VGGNet [33], etc.. However, most of improvements focused on changing network structure, adjusting parameters and training methods of models. There is no improvement in automatically increasing intelligence over time. The intelligence of models needs interference and labeling.

A proper Adaptive Learning model can automatically recognize objects, train, assess and update its intelligence to replace the old model. Humans only interfere at the beginning of the establishment of the model. The adaptation of a recognition model is represented by the update of more diverse data, the capability in recognizing strange and difficult objects, and the continuous changes of training parameters based on training datasets. For example, in an autonomous car system, an initial model can only recognize simple forms of other vehicles, lanes, walkers, trees, buildings and traffic signs. Over time, the system will recognize strange forms of each object, train and update to replace last models. The system will increasingly be more intelligent as the car moves on the road.

1.3.2 Deep neural network (DNN)

DNN is an artificial neural network (ANN) with hidden layer units that integrate with one another from input to output. Similar to an ANN, a DNN has a model integrating complicated non-linear relationships. The difference between a deep neural network and a simple network is that the former has more nodes in each layer and more hidden layers. This demonstrates by the number of layers and nodes where data go through in recognition processes.

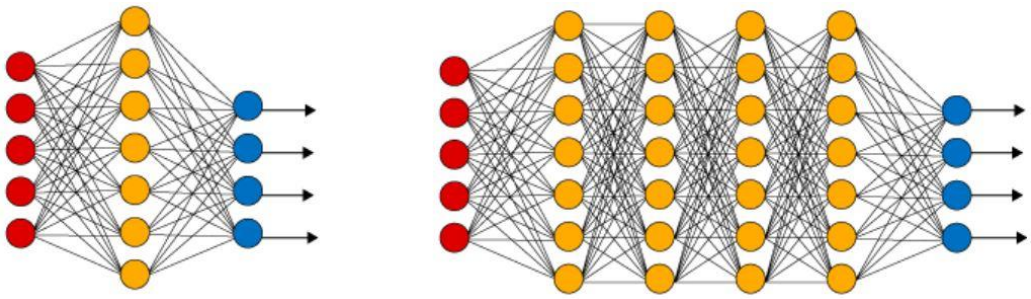


Figure 1.4 Simple Deep Learning network with one layer and Deep Learning network with multiple hidden layers (Source: https://www.kdnuggets.com)

In initial versions, deep neural learning networks, which were similar to one-layer networks, were formed by an input layer, an output layer and a hidden layer. Later, there were deeper networks with more than three layers (Figure 1.4). Thus, "deep" concept means the number of hidden layers in neural networks.

In each layer in Deep Learning networks, nodes will be extensively trained with unique features based on outcomes of prior layers. Once data go to inner layers of neural networks, they will be more complicated. Nodes can recognize, synthesize and recombine features from prior layers to display features in higher levels. This is known as "hierarchical featuring", which is the hierarchy process where data become more complicated and abstract. A deep neural learning network is to solve great datasets in multiple dimensions with billions of parameters treated by non-linear functions.

Deep neural learning networks can recognize potential structures in unlabeled and unstructured database that are very common in reality. Studies found that deep neural learning networks are very effective in analyzing unstructured data such as raw multimedia data, images, documents, sounds and video. This means that deep neural learning technique can effectively solve problems in analyzing, recognizing and classifying data that are unstructured, homologous or abnormal.

1.3.3 Convolution neural network (CNN)

CNN is a type of deep neural learning networks. LeCun [34] was one of pioneers in study and application of the network. CNNs use regularized versions of

multilayer perceptron to simplify pre-analyzing process. CNNs imitate biological brain processes in which the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field (input layer). The receptive fields of different neurons partially overlap such that they cover the entire visual field.

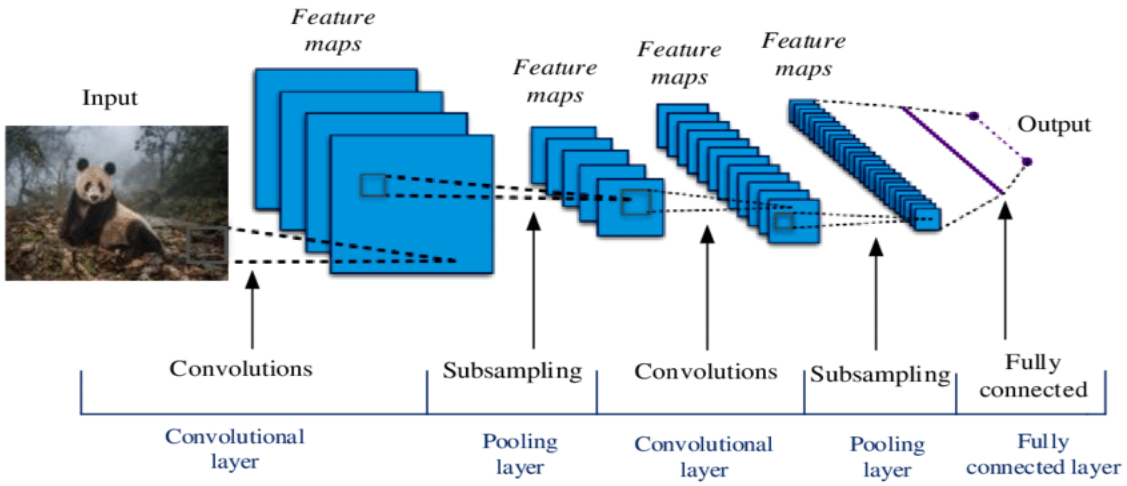


Figure 1.5 Architecture of a simple convolution neural network (Source: https://medium.com)

Architecture of a CNN includes an input layer, an output layer and a number of hidden layers in between. Hidden layers consist of convolutional, pooling, rectified linear unit (ReLU), full connected and normalization layers, as shown in Figure 1.5. Thus, regarding to general architecture, CNNs include multiple convolutional, pooling, normalization layers and may have full connected layer.

Some CNNs which have been introduced and commonly used are AlexNet [27], GoogLeNet [28], Microsoft ResNet [29], R-CNN [30], Fast R-CNN [31], Faster R-CNN [32] and VGGNet [33].

1.4 Domestic and international research

1.4.1 Domestic research

In Vietnam, from the 1990s to the early years of the 20th century, there were participation of the well-known researchers Assoc. Prof. Ngo Quoc Tao, Assoc. Dr. Do Nang Toan, Assoc. Dr. Luong Chi Mai, etc. in the field of AI research,

especially image processing and recognition. Their research works include handwriting recognition [35, 36], Vietnamese handwriting [37, 38], speech recognition, human face detection [39, 40, 41], simulation of the human body [42], etc. Most of the research and publications exploit classic algorithms such as SVM, Random Forest, hidden Markov models, artificial neutral networks, and so on. These researches are considered as significant foundations for students and graduate students' reference. Along with the publication of research, many publications on image processing and object recognition were also published.

After the first decade of the 20th century, AI growth, along with computer hardware, enables the fields of machine learning and object recognition to make advance. In Vietnam, however, in the first time, research on Artificial neural networks and Convolution Neural Networks were still very primitive with no domestic research on this specific field. The research and publications mostly come from oversea Vietnamese PhD students. From 2015 up to now, there have been many articles published on the international journal ISI, Scopus. These articles came from research groups such as Hanoi University of Technology [43, 44], Ton Duc Thang University, National University of Ho Chi Minh City, Duy Tan University - Da Nang, etc. Besides to the research groups of the institutes and labs, many independent research works have also been published, including researches assisting the fields of health, transport, agriculture and national defense such as autonomous cars, robots, and human action recognition, classifications, [45, 46, 47], etc.

1.4.2 International research

1.4.1.1 Overview

The AI history and machine learning has gone through many phrases. The intelligence of the machine has been simulated and demonstrated by Alan Turing Since 1950. By 1955, John McCarthy, an American computer scientist and cognitive scientist, first coined the term "Artificial intelligence", meaning the science subject

and intelligent computer engineering. One year later, he hosted the Dartmouth Conference, the first conference on this topic with the participation of experts from various universities and companies such as Carnegie Mellon University, Massachusetts Institute of Technology and IBM. Since then, the term "artificial intelligence" has been widely used.

Through many different stages, AI in general and the field of machine learning in particular are still growing, continuously fulfill their task of exploring many important algorithms such as Support vector machine, Random Forest, Neural network, K-mean, Decision tree, Booting, Hog, and so on. These algorithms are the fundamental for the growth of algorithms and applications in recognition, object classification, data processing, and so on. Along with the growth of computer hardware, the 1998s forward, Deep Learning and Convolution neural network which is one of the components of machine learning, has made great progress with many applications in life [48, 49, 50, 51, 52]. Yann LeCun is one of the pioneers in this particular field. LeNet, one of the most famous CNN networks, was developed by Yann LeCun in the 1998s. The structure of LeNet consists of 2 layers (Convolution + maxpooling) and 2 layers fully connected layer and the output (softmax layer) with the recognition accuracy up to 99%.

By 2012, AlexNet model [53] was introduced by Alex Krizhevsky and his colleagues. The AlexNet with a large margin (15.3% VS 26.2% error rates) is a CNN network that won the ImageNet LSVRC-2012 contest in 2012. AlexNet is a CNN training network with a very large number of parameters (60 million) compared to LeNet. Its characteristics are:

- ReLU is used instead of sigmoid (or tanh) to deal with non-linearity, increasing computing speed by 6 times.

- DropOut is used as a new regularization method applied to CNN. Dropouts not only enable the model to avoid over-fitting but reduce model training time.

- OverLap pooling is used reduce the size of the model (Traditionally pooling regions does not overlap).
- Local response normalization is used to normalize each layer.
- Data augmentation technique is used to create additional training data by translations and horizontal reflections.
- AlexNet is trained by 90 epochs within 5 to 6 days with 2 GTX 580 GPUs. Using SGD at learning rate 0.01, momentum 0.9 and weight decay 0.0005.

The architecture of AlexNet consists of 5 convolutional layers and 3 fully connection layers. Activation ReLU is used after each convolution and fully connection layer.

This was followed by new models proposed in turn, decreasing the error percentage, increasing the model's complexity with a deep architecture. The proposed models include VggNet 2014, GoogleNet2014, MicrosoftResNet 2015, Densenet 2016, etc. In parallel with the improvement of network architecture, experimental training and recognition to almost all objects in reality were conducted by the models with high accuracy. For example, AlexNet is capable of identifying and classifying 1,000 different objects.

In addition, many works from research institutes and universities in the world have been published that proposed solutions to each AI specific problem on robotic, auto vehicles, etc. Each field, then, continue to be broken down by different levels for solving. For instant, the problem of self-driving cars can be classified into the following cases [54]:

- The lane - recognition problem for self-driving cars
- The on – road object recognition problem for self-driving cars
- The traffic sign recognition problem for self-driving cars
- The distance measurement problem for self-driving cars

- The pedestrian movement prediction for self-driving cars
- The obstacle recognition problem for self-driving cars

1.4.2.2 On-road object detection

(1) Pedestrian

For pedestrian detection, nowadays, there are many contributions that using "tracking" technologies to detect and recognize the objects [55, 56]. Using tracking technologies can bring high accuracy; however, this approach which takes a long time to process becomes a challenge of AVs, especially in case of emergency.

Recently, there have been some proposed approaches for pedestrian recognition technologies. Histograms of Oriented Gradients (HOG) [57, 58], for instance, is a feature descriptor used in computer vision and image processing. HOG recognizes objects using information about direction and color/grayscale that change in each local area of the image and standardizes the contrast between blocks to improve accuracy. Latent SVM [59] is the algorithm classify objects by looking at their parts and its geometric location constraints. The detector requires a trained model that uses the image dataset including some desired images and some opposite images, or Kanade–Lucas–Tomasi (KLT) [60] algorithm,...

Among mentioned approaches, CNN is the promising solution to extracting features. There are some CNN models which are used such as AlexNet, GoogleNet, Microsoft ResNet, Region Based CNNs (R-CNN, Fast R-CNN, Faster R-CNN). Each model has different features in terms of processing speed and the rate of accuracy. In this thesis, for optimizing the process of feature extraction, we propose the models which have already been trained of the algorithm (pre-trained). Subsequently, extracted features from features extracted from CNN models are used for the classifier model. Depending on the model and the actual requirements, different classification algorithms for training such as k-Nearest Neighbor (kNN), SVM, Random Forest and Fully Connection...are applied.

There are a few algorithms for pedestrian action recognition which are proposed in previous works [61, 62, 63, 64,65]. However, they focus on recognizing pedestrians without basing on specific scenarios when attending traffic system; AVs is not able to detect the levels of alert with different level of dangers.

(2) Vehicles

There are many approaches to detecting and recognizing vehicles in single images or video images extracted from cameras on routes. Two main research directions are using traditional methods only or combining them with Deep Learning.

Traditional methods include the proposed Gauss mixture model (GMM) and the Kalman filter [66]. GMM is used to recognize vehicles and the Kalman Filter is used to track vehicles under light adaptive conditions. Another suggestion is the Optical Flow estimation method [67], which uses the edge features of images (as determined by the Canny algorithm) to determine the moving vehicles. In feature extraction, there are several methods, such as Scale Invariant Features Transform (SIFT) [68] and Histogram of Oriented Gradients (HOG) [69], followed by using the SVM classifier to determine means of transport [70]. The evaluating results show that the use of the HOG extraction features method and SVM classifier for recognition brings good results. A recent study proposes the use of feature extraction methods to recognize vehicles, vehicle counts, and classification [71]. In this method, GMM is used to segment images, and then Canny edge detector is used to define the boundary and extract features. Generally, traditional methods use the extraction of features associated with the shape, color, and texture of images to represent objects of interest (IO). After that, the classifier architecture is used to recognize the meaning of the transport situation.

Researches of Deep Learning often use high-performance Convolution Neural Network models, such as AlexNet[27], GoogleNet [28], Microsoft ResNet [29], R-CNN [30], Fast R –CNN [31], Faster R-CNN [32], etc. For example, a

recent research provides comparisons between the R-CNN and Faster R-CNN models [72] or between the AlexNet and Faster R-CNN models [73], in terms of vehicle recognition. Some proposed approaches bring high accuracy in vehicle recognition, even from satellite images [74, 75, 76] or from the low-resolution video [77]. However, most of the proposals, including the traditional methods and methods that use Deep Learning, only solve the problem of vehicle detection. There it very little proposals for the recognition of specific vehicles, such as motors, cars, coaches, trucks, etc.

(3) Other objects

The identification of unmoving objects is considered the most diverse, including recognizing and identifying roadsides and curbs. A possible solution is the Detection of Roadside Vegetation [78], which uses a set of color features extracted from the camera image and the Support Vector Machine (SVM) model to identify objects. Besides, in the urban road sections, there are solutions which identify road markers[79,80] helping automatic vehicles determine the moving trajectory. These solutions focus on the use of Gaussian and Kalman filters in conjunction with the Hogh algorithm to identify the position of road markers serving the automatic direction. Some approaches use inductive devices [81, 82]installed along the curbs and line lanes of the road, allowing AVs to continuously transmit signals and determine the exact direction of the car.

Recently, high accuracy of solutions such as image segmentation[83,84] color label assigning, and training and identifying on the pixel of the image has helped AVs to identify multiple objects interacting in the frame. In terms of computer vision, the image segmentation is a process in which a digital image is split into many different parts (a set of pixels, also known as super pixels). The target of image segmentation is to simplify or

change the image expression into a direction which is more meaningful and easier to analyse. The image segmentation is usually used to identify the position of objects and borders (straight lines or curves).

1.4.2.3 Adaptive learning to object detection

In recent years, many studies on Adaptive Learning solutions have been published [85, 86, 87]. with the main focus on three main directions which are Select detection models, adapting in terms of data and adapting in terms of algorithms, parameters.

Select detection models: These solutions focus on the automatic selection of recognition model types without using a specific default model (e.g., choosing between CNN and SVM) [88, 89]. The selection of training models will make it possible to solve each specific case of the data and provide greater accuracy. In addition, these solutions also allow assessment of data types, data models, and so on for automatic selection of an appropriate model [90, 91]. However, the studies and assessments of these solutions still have several limitations in terms of quantity and quality, and these issues remain unresolved. In addition, a higher hardware configuration is required for a solution during data processing.

Solving problems with data includes developing a model that is capable of self-improving data, enabling a specific CNN model to automatically update data without any human intervention. Adaptive data is a potential research trend where the CNN model can learn and update models so that they resemble the behavior model of the human mind. For adaptive data, there are many studies regarding online tracking that adjust data to track objects or extract features of these objects[90, 92]. Our recent paper proposed using an object tracking process, collecting appropriate data, and automatically retraining the recognition model as a solution to enhance automatic recognition quality of objects. As a result, the post-trained model is more identifiable than the old one. In particular, the model works automatically without any human intervention and adjustment.

Solving problem with algorithms and parameters is solution focusing on algorithms and parameters of CNN models, which creates adaptive changes in CNN layers during training. Specifically, some proposals centralize building frames that are embedded in different layer positions to change weights in the training and recognition process of the CNN model [5]. Other research focus on changing features, including alignment of local depth features to achieve model simplification [88, 93]or changing features between convolutional layers and customizing layers [94, 95, 89]. In general, the research aims to optimize the structure of the CNN model. The changes in layer layout and layer customization have resulted in positive changes. The new models offer a smarter training process, especially in terms of their ability to retain important features during the training [96, 92, 97, 98, 99, 100, 101], 102]. Recently, several proposals have focused on the automatic selection of training process parameters [96, 92, 97, 98, 99, 100, 101, 102]. Among them, one solution initially used a small dataset to evaluate the parameters' effectiveness before selecting the appropriate parameters and conducting training evaluation on the entire data [92]. A different solution used the random search method of parameters before conducting cross-validation on a given number of times to choose parameters matching the recognition model [96]. Furthermore, a combination of evolutionary algorithms to automatically optimize CNN structure by hyperparameters was used in one study [101]. Regarding the configuration hyperparameter selection of a CNN model, methods using the random search [103], the grid search, or Bayesian algorithms [104, 105, 106] are the most notable

Up to now, it can be said that Deep Learning network applied in AI has made a fairly long step on the "intelligent" path but been unable to be "self-intelligent" which is as a a big barrier . Solutions that enable systems to be capable to learn and be self-intelligent as human can. It is also the exploration direction included in the goal of this research thesis, contributing to the path of conquering new heights of artificial intelligence.

CHAPTER 2. RECOGNIZING OBJECTS BY DEEP LEARNING

This chapter presents the Deep Learning models of object recognition which were studied by the author with focus on identifying objects in traffic such as pedestrians, vehicles, road, etc. From the research and experiment process, the author has tested and proposed the solutions to improve the recognition accuracy of Deep Learning models

2.1 Object recognition problems

Artificial intelligence has developed and affected all aspects of the life. machine learning with its subset, Deep Learning, has greatly contributed to the artificial intelligence. With the basic of convolution neural network, Deep Learning has great contributions such as voice recognition, object recognition, medical applications, smart transportation and robot.

One of improvements in Deep Learning is the capability to recognize and process images. Specifically, Deep Learning models can precisely recognize objects via training sessions. The current study focuses on object recognition of CNN models. This chapter will focus on object recognition for autonomous vehicles to assess effectiveness of recognition of the CNN model. The objects include:

- Roadway
- Pavement
- Vehicles: motorbikes, cars and vans.
- Pedestrian movement
- Other objects such as houses, trees and sky

2.1.1 Problem: Pedestrian action prediction

Of all objects relating to the movement of autonomous vehicles, pedestrian recognition is considered the most difficult due to its complications in recognition and movement area and orbit. Therefore, it is a priority to precisely predict

pedestrian's action and walking speed to ensure the safety of pedestrians and vehicles. There are mainly three types of pedestrians: crossing, walking and waiting pedestrians. The three types involve in all possible interactions between pedestrians and autonomous vehicles. When pedestrians move or stand on the side roads, features are presented in their gestures, locations and scenes (roadway, side roads, road edges, etc..). Thus, it is possible to extract features from images of pedestrians and uses these features to train to predict and recognize pedestrian movement.

The proposed approach includes two phases which are: i) training a classifier model, which is used to predict pedestrian movement, with features extracted from CNN models (Figure 2.1); ii) with the frame image from real-time video of AV on the road, the order of process is: detecting pedestrians, extracting region of interest (ROI), extracting features of ROI and predict pedestrian movement in this ROI (Figure. 2.2). To extract features, CNN model of AlexNet is proposed [27]. To detect pedestrian, ACF algorithm is proposed [2 ,3 , 107] to train and predict pedestrian movement, SVM model is proposed.

Figure 2.1 The process of extracted features by CNN model from image dataset

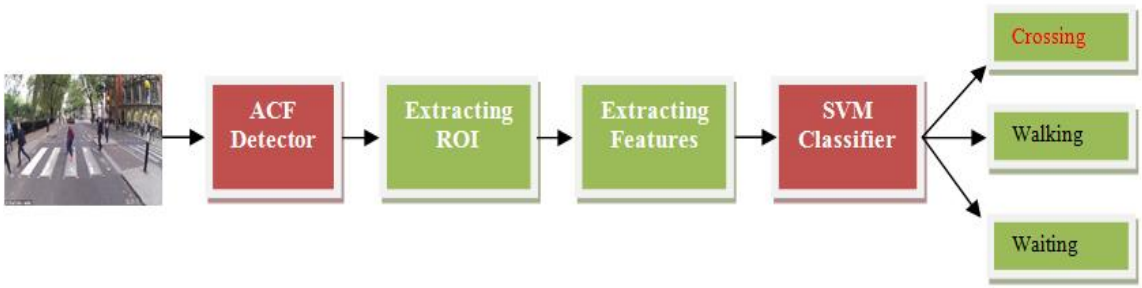


Figure 2.2 The process of pedestrian movement prediction

The resolution of used camera is 2 Megapixels or more with the minimum resolution of collected images of 72 dpi.

2.1.2 Problem: Vehicle recognition

It is useful to detect and recognize vehicles in traffic control and separation. On the line of technology development, the need to travel and number of vehicles have increased. There are various problems in managing and separating vehicles which raises the need to apply automatic control systems with high level of precision. There are a number of solutions for monitoring systems and decision making in Intelligent Transportation Systems (ITS) such as sensors that help with reading data from devices attached to vehicles and use of the internet to network vehicles. However, some solutions could not apply in the reality due to limits in device production, internet bands and high expenses in establishment. Thus, it is essential to introduce automatic recognition and classification system for vehicles.

The proposed solution commences with the acquisition of images from the surveillance camera in ITS. Collected images are used to recognize objects of interest and determine the type of transportation. There are many methods for detecting vehicles, yet in this article, we focus on recognition models instead of detecting vehicles. By default, we use the semantic segmentation model based on Segnet's CNN architecture [108, 84]. Vehicles detected will then be extracted to determine regions of interest (ROI). Area of interest is a sample of the vehicle. Depending on the proposed method, it is possible to use the CNN model as well as combine with data augmentation to improve accuracy. Recognition results are used in the ITS system to alert vehicles when they are not allowed to enter the limit line and to handle violations.

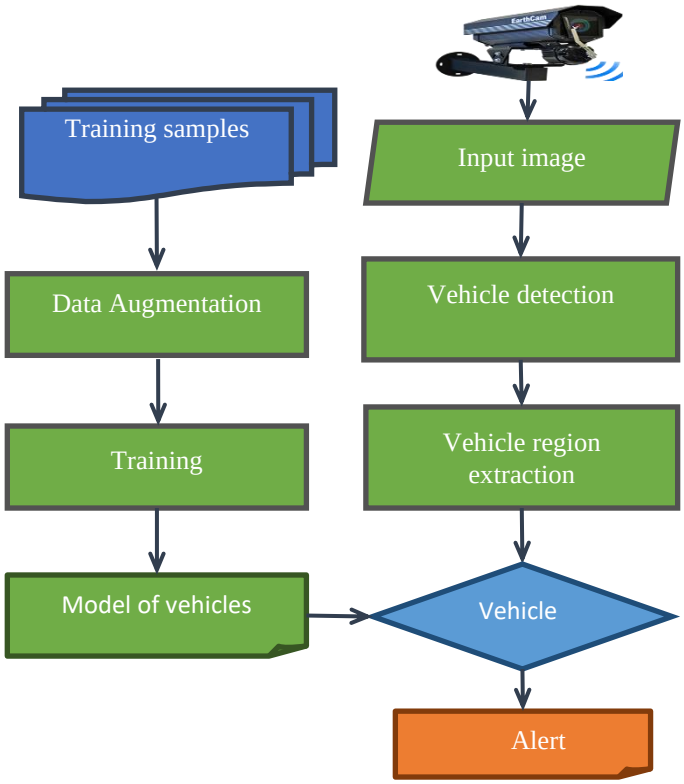


Figure 2.3 Proposed vehicle detection model

2.2 Suggested solution

Object recognition has been introduced with three basic steps:

(1) Detecting and extracting areas of interest

(2) Extracting features and training recognition models

(3) Recognizing objects

However, the step 1 may be unnecessary once target object was identified. Each step can have different techniques:

- Detecting and extracting areas of interest: use image meaning to extract areas of interest (pedestrians, vehicles, traffic signs, etc..).

- Extracting features and training recognition models: build and introduce Deep Learning models to extract features of objects. It is suggested to use SVM model to train recognition models.

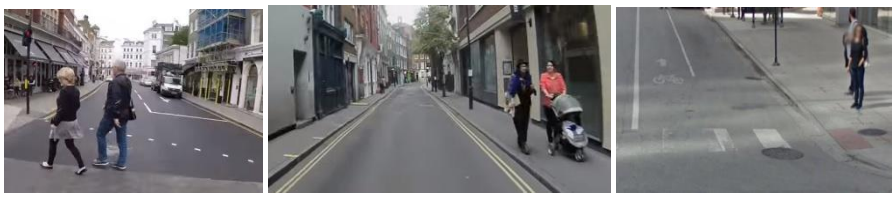
- Recognizing objects: use trained recognition models to recognize and classify objects according to individual problems.

2.2.1 Solution to pedestrian recognition

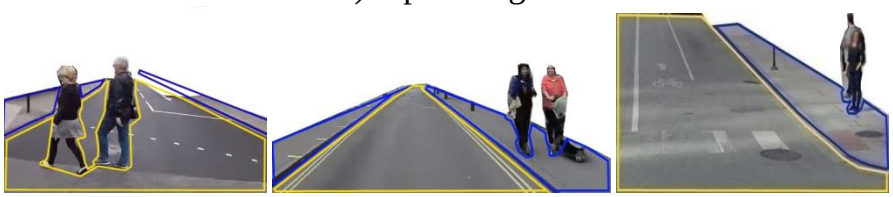
2.2.1.1 Extracting features and training classifier model

In machine learning, a convolution neural network is a class of deep that is usually applied in analyzing visual imagery. In the CNN, many models are created and proposed. Each model has its own characteristics of architecture, size and number of layers, etc. Common models such as AlexNet, GoogleNet, Microsoft ResNet, Region Based CNNs (R-CNN, Fast R-CNN and Faster CNN) are low error-rate models. In this thesis, CNN model of AlexNet, which reduces the time of process, is proposed.

The AlexNet model extracts and keeps the basic features of input images as stimulated in Figure 2.4. 3000 input images are used, including 1000 images of crossing pedestrians, 1000 images of walking pedestrians and 1000 images of waiting pedestrian. These images taken from the real street videos on the Internet were processed (selected and cut into the suitable frames) (http://youtube.com). With each image, the CNN will extract the rich features, which are pedestrian postures, roadways, roadsides and positions of pedestrians on road, figure 2.5. The rich features extracted will be used for training SVM classifier model.



a) Input image.



b) Rich features simulation.

Figure 2.4 Input images and simulate rich features of image

.

In CNN model, many feature layers can be extracted such as convolution layer or full connected layer but the more advantageous layer is layer 19 (fc7 – 4096 fully connected layer) – the one right before the classification layer.

Literally, in cases of object recognition such as animals, things and vehicles, the rate of recognizing object is higher (90% to 100%). In case of predicting the action of pedestrians, the features of input images focus on not only a specific object but also others such as vehicles, buildings, trees, and things around roadsides as shown in Figure 2.5.

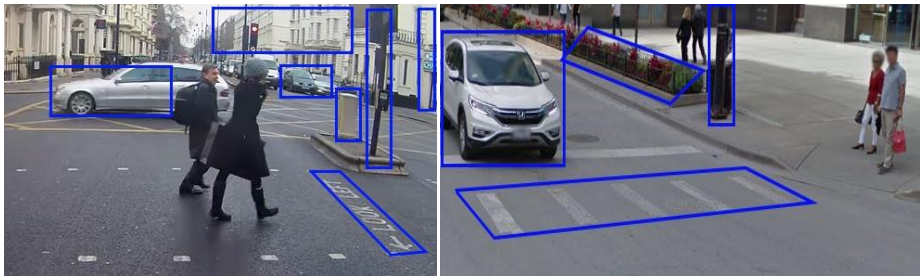


Figure 2.5 Influence of other objects on the road on pedestrian movement prediction

In this regard, in term of accuracy, ACF algorithm is used to detect pedestrians before extracting ROI, classifying and predicting the action of pedestrians.

2.2.1.2 Pedestrian action prediction

Pedestrian detection by ACF. ACF classification model, specified as *'inria-100x41'* or *'caltech-50x21'*, is person detection. The *'inria-100x41'* model was trained using the INRIA Person data set. The *'caltech-50x21'* model was trained using the Caltech Pedestrian dataset. The ‘*inria-100x41’* model (default) is proposed in ACF. In ACF algorithm, detection scores value - confidence value - return an M-by-1 vector of classification scores in the range of [0..1]. When a pedestrian is detected, a bounding box will appear. The scores on top of bounding box are confidence value (by percentage). Larger score values indicate higher accuracy in the detection. In some complex images, the ACF algorithm sometimes

recognizes errors. During the real-time experimental process, the score value 0.25 is proposed to avoid error-recognizing cases. For example, if the score value is 0.1, the result will not be accurate in some cases (Figure 2.7 (a)) and if the score value is 0.25, the result will be of higher accuracy (Figure 2.7 (b)).

Figure 2.6 Example input image for recognition

a) b)

Figure 2.7 Pedestrian detection with scores = 0.1 (a) and scores = 0.25 (b)

In particular, when the AV moves on the roads, there are some cases in which so many pedestrians appear in one frame of the video. Therefore, to ensure the accuracy, it is considered that a frame be extracted into many separate frames to be easily recognized in each case. The region extracted is called ROI (Figure 2.8). Also, in real-time, the image received from AV is in big size and contains a lot of irrelevant data. Hence, extracting ROI of image at a certain scale, which removes irrelevant objects around, is necessary for each pedestrian detected. Extracting ROI of image helps the CNN model extract the exact features and reduce the error rate in the process of action recognition and classification of the SVM. The size of ROI is

proposed as follows:

Supposing that H and W are height and width of the rectangle covering pedestrian object; x and y are the coordinates of the top left of rectangle and Width and Height are the size of input image, the values x1, y1, W1, H1 describe the size of ROI which are defined as follow:

$$\begin{cases} x1 = x - H \times 1.5 \\ y1 = y - W \\ W1 = W + H \times 3 \\ H1 = H + W \times 2 \\ if\ (W1 > Width)\ then\ W1 = Width \\ if\ (H1 > Height)\ then\ H1 = Height \end{cases} \quad (1)$$

In special cases, when x1, y1, W1, H1 are smaller than the edge value of the frame or bigger than the size of the input image, the values equal the edge values of the image.

$$\begin{cases} if\ (x1 < 0)\ then\ x1 = 0 \\ if\ (y1 < 0)\ then\ y1 = 0 \\ if\ (x1 + W1 > Width)\ then\ x1 = Width - W1 \\ if\ (y1 + H1 > Height)\ then\ y1 = Height - H1 \end{cases} \quad (2)$$

On the other hand, when ROI is out of image input size, the offset value of ROI on the opposite side is proposed in Figure 2.8.

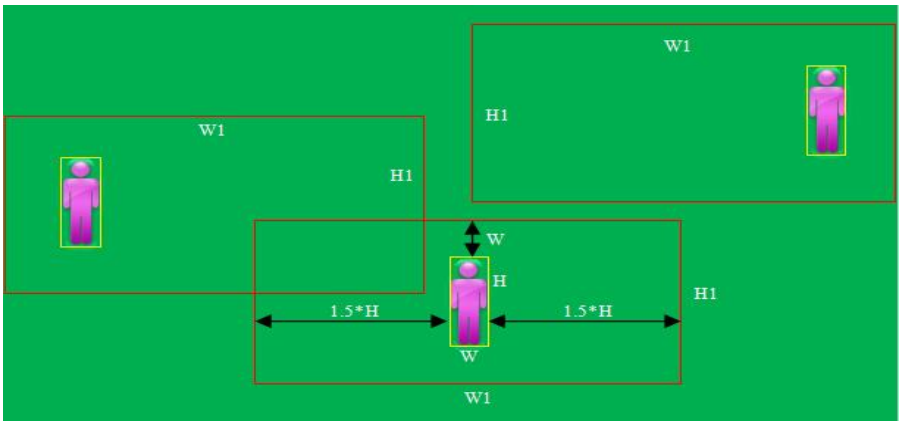


Figure 2.8 ROI extraction from pedestrian image

.

Pedestrian movement prediction: After ROI is extracted into a single image, the features are extracted (by CNN model) to be classified (by SVM classifier model). The outputs are labeled according to values of prediction of pedestrian case (i.e., Pedestrian_crossing, Pedestrian_waiting, Pedestrian_walking).

(i) Pedestrian_crossing: When a pedestrian is crossing or walking in the road of other vehicles.

(ii) Pedestrian_waiting: When a pedestrian is standing on the roadside and waiting to cross.

(iii) Pedestrian_walking: When a pedestrian is walking on the edges of the road.

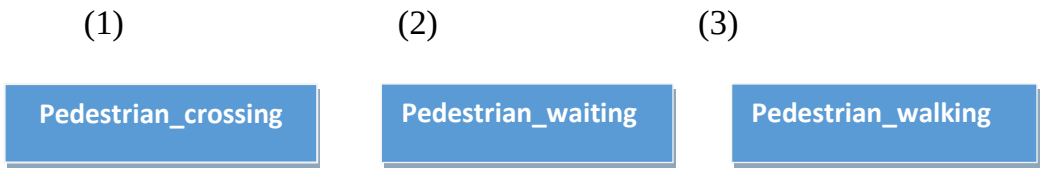


Figure 2.9 The order of classifications of pedestrians when there are many pedestrians on the road in an input image

2.2.2 Solution to vehicle recognition

2.2.2.1 Sequential Deep Learning architecture

Usually, available pre-trained network models can be used to re-train the vehicle recognition models. However, in our approach, reusing the trained model is inappropriate, as the size of the old models differs from the actual images obtained simultaneously. Besides, the training parameters do not support accuracy improvement. Some proposed models, such as AlexNet [53], GoogleNet [28],... are only effective for general recognition problems, not for this specific recognition problem. There are many different approaches to building a CNN model in vehicle recognition. In this study, we constructed a 24-layer CNN architecture, shown in Table 2.1, consisting of the input layer, convolution layer, rectified linear unit layer (ReLU), cross-normalization, max-pooling, and fully connected layer. The network model transforms the input image into a serial hierarchical descriptor. The neural aggregate input is the intensity values of the image applied to the CNN model. Input

sample includes 128×128×3 images. In this model, filters at the first layer concern to three-color channels, namely R-G-B. Filters operate independently and jointly among hidden layers, involving three channels of the input image. The final layer handling the feature vector will be extracted into the classification layer. A convolutional layer implements a combination of mapped input images with a filter size $n_x \times n_y$.

Table 2.1 CNN architecture with 22 hidden layers, 1 input layer, and the final classification layer

TT	Layer type	Parameter
1	Image Input	image size 128x128x3
2	Convolution	64 7x7x3 convolutions with stride [1 1]
3	ReLU	ReLU
4	Normalization	Cross channel normalization
5	Max Pooling	3x3 max pooling with stride [1 1]
6	Convolution	64 7x7x64 convolutions with stride [1 1]
7	ReLU	ReLU
8	Max Pooling	2x2 max pooling with stride [1 1]
9	Convolution	64 7x7x64 convolutions with stride [1 1]
10	ReLU	ReLU
11	Normalization	Cross channel normalization
12	Max Pooling	2x2 max pooling with stride [1 1]
13	Convolution	64 7x7x64 convolutions with stride [1 1]
14	ReLU	ReLU
15	Max Pooling	2x2 max pooling with stride [1 1]
16	Convolution	64 7x7x64 convolutions with stride [1 1]
17	ReLU	ReLU
18	Normalization	Cross channel normalization
19	Max Pooling	2x2 max pooling with stride [1 1]
20	Fully Connected	1024 fully connected layer
21	ReLU	ReLU
22	Fully Connected	4 fully connected layer
23	Softmax	softmax
24	Classification Output	crossentropyex with 4 other classes

2.2.2.2 Data augmentation

The training data set classified during the collection is shown in Figure 2.10.

In order to improve the accuracy of vehicle recognition, we propose to augment data about 10 times. Images are rotated [-5^0, 5^0], flipped or added noise, yet no changes will be made to the image quality during training. The training data set after augmentation is shown in Table 2.5.

2.3. Experimental evaluation

2.3.1 Pedestrian detection

2.3.1.1 Extracting features and training classifier model

The experiment is carried out with about 3,000 images being extracted by CNN model. There features are used for training of SVM classifier model. Table 2.2 shows the image and label datasets of extracted and trained features.

Table 2.2 Image and label datasets of extracted and trained features

Class	Number	Label
Pedestrian crossing	1,000	Pedestrian_crossing
Pedestrian waiting	1,000	Pedestrian_waiting
Pedestrian walking	1,000	Pedestrian_walking

90% of images from each set is used for the training data and the rest 10% is used for the data validation.

2.3.1.2 Pedestrian detection and action prediction

With the input images (i.e., Figure 2.6), after using pedestrian detection ACF algorithm, the output is executed as in Figure 2.11. In case of the input images with many pedestrians in a frame, we extract ROI into a single image for action prediction by SVM classifier as shown in Figure 2.11. Each image in Figure 2.11 will be extracted features; finally, the system will rely on the SVM classification model to conduct action prediction of pedestrian and issue appropriate alerts for AV accordingly in Figure 2.9.

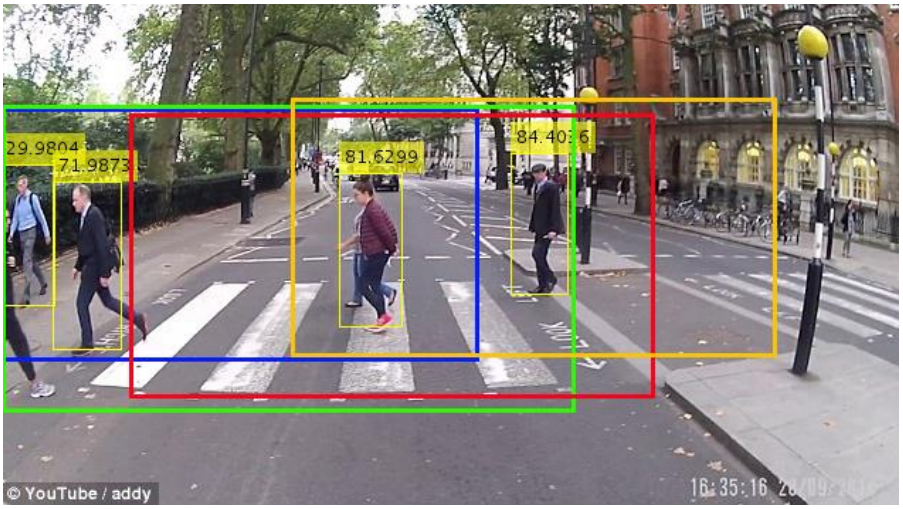


Figure 2.10 Pedestrians detected and ROI extracted

The maximum results of rate-recognition after training and comparing with dataset in Table 2.2 are as follow:

Table 2.3 Maximum confusion matrix for pedestrian action prediction

	Pedestrian crossing	Pedestrian waiting	Pedestrian walking
Pedestrian crossing	0.9796	0.0204	0
Pedestrian waiting	0.0612	0.9286	0.0102
Pedestrian walking	0.0102	0.0408	0.9490

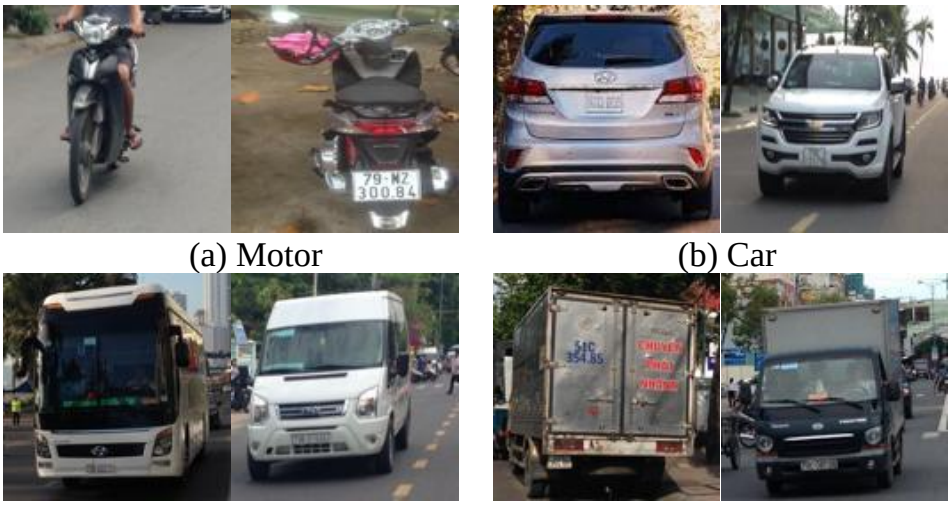
The result of experiment in real-time video on the road gives minimum accuracy rate of 82%, maximum of 97% and the speed for processing reaching 0.6 second per pedestrian detected. They are promising results for potential self-driving.

2.3.2 Vehicle recognition

2.3.2.1 Experimental data

We have conducted experiments on a real database of vehicles including motors, cars, coaches, trucks taken from actual traffic situations. Camera systems typically receive signals in front of or behind the vehicles in traffic. This dataset is

collected from different practical contexts on different traffic routes. The training dataset is divided into 4 different vehicle classes, including motors, cars, coaches, trucks simulated in Figure 2.10, with 8,558 vehicle images. The dataset was actually collected in Nha Trang city, Khanh Hoa province, Vietnam. Dataset is partitioned into 60% for training and the remaining 40% for evaluation as shown in Table 2.4.



(a) Motor (b) Car

(c) Coach (d) Truck

Figure 2.11 Some examples of vehicle categories

Table 2.4 Training data

Categories	Number of samples			Sample size
	Overall	Train	Evaluation	
Motor	2673	1604	1069	128x128
Car	2808	1685	1123	128x128
Coach	1640	984	656	128x128
Truck	1437	862	575	128x128

Table 2.5 Training data after augmentation and balance data

Categories	Number of samples
Motor	16040
Car	16850
Coach	17712
Truck	17240

2.3.2.2 Training CNN

Result obtained after CNN model training is shown as follows:

(i) Filter parameters: The first convolution layer uses 64 filters, whose filter's weight is shown in Figure 2.12:

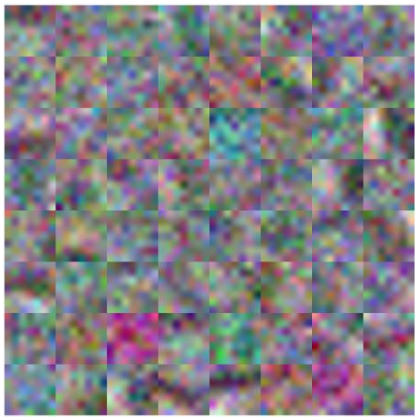
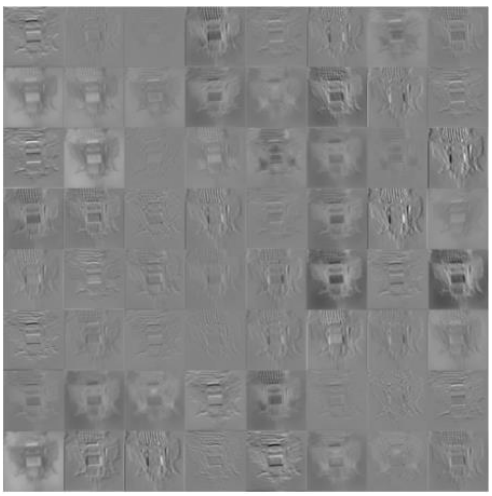


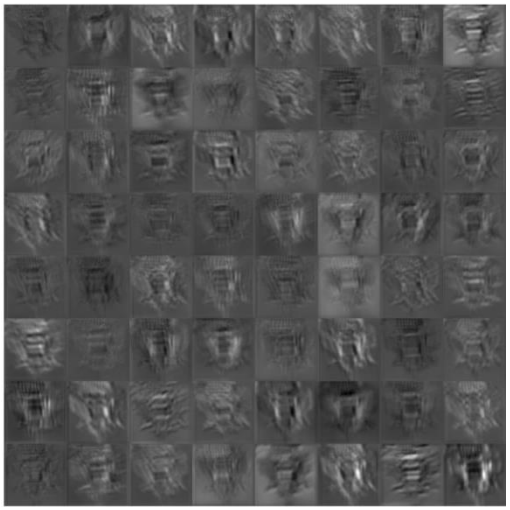
Figure 2.12 The weight values of the filter of the first convolution layer. This layer consists of 64 filters size 7x7, each of which is connected to three RGB image input channels

(ii) Convolution result: The sample images fed into the network through a convolution filter and the obtained data show components distinct from the original RGB image with various feature result, creating a variety of vehicle features. The output value of the convolution set contains a negative value, which should be normalized by linear adjustment. The output of some layers is shown below, with the input pattern of the motor sample.



(a) The output of 64 convolutions at the first convolution layer

(b) The linear correction value after the first convolution layer



(c) The output of 64 samples at the second Convolution layer

Figure 2.13 Some results of linear convolution and linear correction for the input images being motors

2.3.2.3 Categorical vehicle recognition

Based on the experiment, three different methods have been evaluated on the same set of sample data as shown in Table 2.4. Methods include: (i) Traditional methods of HOG and SVM; (ii) CNN network; (iii) CNN network in combination with data augmentation.

The accuracy of the HOG and SVM method on the sample data set was 89.31%. Details of the sample size for each type and recognition result are shown in Table 2.6.

Table 2.6 Confusion matrix of vehicle recognition using HOG and SVM

	Motor		Car		Coach		Truck	
	1069		1123		656		575	
	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)
Motor	**1029**	**97.26**	16	1.53	15	1.87	9	1.75
Car	25	2.36	**989**	**94.37**	77	9.59	32	6.23
Coach	1	0.09	23	2.19	**599**	**74.60**	33	6.42
Truck	3	0.28	20	1.91	112	13.95	**440**	**85.60**

The evaluated accuracy of the CNN method based on original data was achieved 90.10% on average, as shown in Table 2.7.

Table 2.7 Confusion matrix of vehicle recognition using CNN

	Motor		Car		Coach		Truck	
	1069		1123		656		575	
	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)
Motor	**1026**	**95.98**	38	3.38	1	0.15	5	0.87
Car	32	2.99	**953**	**84.86**	17	2.59	24	4.17
Coach	6	0.56	104	9.26	**617**	**94.05**	58	10.09
Truck	5	0.47	28	2.49	21	3.20	**488**	**84.87**

The evaluated accuracy of the CNN method based on data augmentation was achieved 95.59% on average, as shown in Table 2.8.

Table 2.8 Confusion matrix of vehicle recognition using CNN and data augmentation

	Motor		Car		Coach		Truck	
	1069		1123		656		575	
	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)	#Num	Per(%)
Motor	**1060**	**99.16**	11	0.98	0	0	1	0.17
Car	5	0.47	**1057**	**94.12**	8	1.22	13	2.26
Coach	0	0	41	3.65	**645**	**98.32**	51	8.87
Truck	4	0.37	14	1.25	3	0.46	**510**	**88.70**

In this study, we also evaluated the proposed CNN model to another traditional approach based on HOG feature descriptor and SVM classifier. Results of the comparison are shown in Figure 2.14.

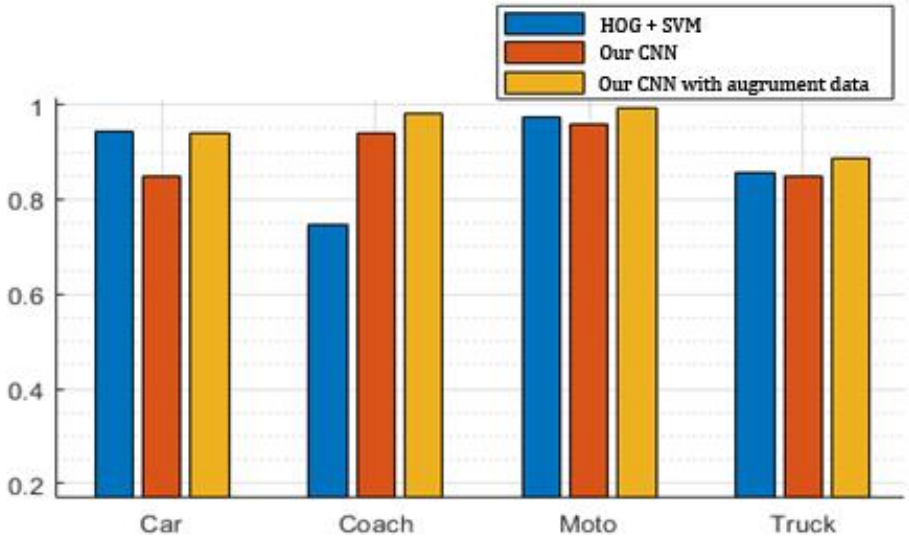


Figure 2.14 Comparison of HOG+SVM, CNN model and CNN with augmenting data

2.4 Conclusion

Artificial intelligence with the development of machine learning, especially recent Deep Learning network, has brought great improvements in computer systems. Study content in Chapter 2 demonstrates the ability to recognize objects of CNN models and intelligence of CNN models in specific cases. Although the study was conducted in a small recognition area, the content clearly demonstrates basic techniques of Deep Learning in recognizing objects and the potential of application. However, a limit of artificial intelligence is the lack of self-study, self-update and self-thinking capabilities. Artificial intelligence would become perfect if learning and data training do not need the interference of humans. Therefore, Chapter 3 aims to build an Adaptive Learning to help autonomous systems in self-study, self-update and self-thinking to narrow the gap between artificial and human intelligence. In the

Chapter 2, the author mentions the two research works which are papers PP 1.1, PP 1.2, PP 1.3.

CHAPTER 3: DEVELOPMENT OF ADAPTIVE LEARNING TECHNIQUE IN OBJECT RECOGNITION

In this Chapter, basing on the research results stated in Chapter 2, the Adaptive Learning solution of self-driving vehicle system data is continuously proposed. The proposed model is capable of self-learning and self-intelligence without any human intervention

3.1 Adaptive learning problem in object recognition

Nowadays, object recognition techniques have achieved high accuracy due to the advent of advanced technologies, such as the deep convolutional neural network. With the growing support of computer hardware, the CNN models have increasingly complex structure, more layers, and a large amount of training data. These systems are capable of identifying most object classes with high accuracy. However, the models just well recognize objects in the case they are a high similarity to the trained data. Meanwhile, the change of status or appearance of objects existing in practice is considerable variety and the image obtaining process of devices is affected by environmental conditions, such as brightness, rain, fog, vibration by movement, etc. Thus, the training dataset, large as it is, cannot cover almost all status of objects in practice. Additionally, training on too large data sets leads to impossible task due to limited computer resources and consuming time. To deal with these problems, proposed an approach solution, which is adaptive for automatically upgrading the recognition model with expected to reach higher accuracy.

3.2 Suggested solutions

3.2.1 Overview of solutions

In this chapter, a solution will be suggested based on Adaptive Learning by CNN models. In this suggested method, the recognition model will automatically update by directly collecting data in the normal operation of an ADAS, training, comparing the accuracy and updating the model. The updating mission will focus on datasets that are different from those in previous training. The solution aims to

update the old model so that it would be more adaptive and accurate. In the Adaptive Learning method, recognition systems can learn and add information by themselves without the help of experts in data labeling. Especially, thank to the increasingly developed online storage technology, development of infrastructure and data transmission solutions on new platforms available (5G, Cloud data, etc.), the problems of the proposed model are expected to be handled by storage and updating of online data. Suggested solutions include five main stages:

(1) Object detection with low reliability

(2) Object tracking in n images in following processes to identify if they are objects of interest.

(3) In case recognized objects with high reliability: label Positive for datasets recognized with low reliability in previous processes. In case recognized objects are not of interest, label Negative for all objected tracked in previous images.

(4) Establishing a training dataset based on the collective combination of training dataset and new dataset.

(5) Retraining and re-updating model if the new version has higher accuracy than the old one.

Trials were conducted to compare suggested model PDNet with modern models such as AlexNet and Vgg. Results showed that the suggested model have higher accuracy than a model that is self-taught over time. Further, the suggested Adaptive Learning model can be applied with conventional recognition models such as AlexNet and Vgg to improve their accuracy.

3.2.2. Analysis

3.2.2.1 Concept Definitions of System Components

Before going into detail the block functions of the system, some concepts are classified and defined as follows:

(1) *Adaptive learning* The self-learning, self-adaptability of a Deep Learning model. The adaptive process supports to automatically improve the ability to recognize objects of the system without the need of manually data complementation and expert support.

(2) *Interest objects* (*IO*) The object of interest to detect and recognize; for example, traffic signs, vehicles, etc.

(3) *Confidence scores* A measure of reliability when an object is detected as *IO*. The confidence score of object O is denoted as Conf(O). $Confidence_H$ is a highly confident threshold.

(4) *Confident tracking* The process of object tracking when an object is detected as an *IO*.

(5) *Lost object* (*LO*) Objects initially recognized as low confident ones, which are tracked through n frames yet are still recognized as *IO* with low confidence score and then do not appear in the next frame.

$LO = O_1, O_2, \ldots O_n \mid Conf(O_i) < Confidence_H$ where i = 1, . . . n

(6) *Negative object* (*NO*) Objects initially recognized as an interest object (*IO*) with the low Confidence score (less than $Confidence_H$), which are tracked through n frames and are eventually recognized not to be an *IO*.

$NO = O_1, O_2, O_n | O_i \in IO$ and $Conf(O_i) < Confidence_H$ where $i = 1, n-1$ and $O_n \in / IO$

(7) *Positive object* (*PO*) Objects initially recognized as an interesting object (*IO*) with the low Confidence score (less than $Confidence_H$), which are tracked through N frames and finally their Confidence score reaches $Confidence_H$

$PO = O_1, O_2, O_n | O_i \in IO$ and $Conf(O_n) \geq Confidence_H$

(8) *Store data* A temporary data set that contains data sets for each object in the confident tracking process. Given S_k set, it is initialized in the Store data when an object is defined as $I\ O_k$. The S_k set is deleted when it is defined as *LO*. It is moved to *ND* if *IO* is *NO* or moved to *PD* if *IO* is *PO*.

(9) *Negative data* (*ND*) This is a bin where *NO* data sets are stored.

(10) *Positive data* (*PD*) This is a bin where *PO* data sets are stored.

(11) *Retraining* The process of retraining the recognition model using a set of data that consists of *PD* sets and *ND* sets when the number of *PD* and *ND* sets reaches a certain score.

(12) *Retrain dataset* The dataset that contains X% collected from all last data sets and the remaining Y% from Confident tracking.

(13) *Updating the recognition model* The process of updating the recognition model of a system when the new model obtained after the evaluation has higher accuracy than the previous one.

3.2.2.2 General Structure of the System

The general idea of Adaptive Learning for recognition model using CNN technology is illustrated in Figure 3.1. The recognition system can be applied to different types of objects. However, for convenience in analyzing the proposed method and functional blocks, we only apply to problems of vehicle and traffic sign classification to illustrate the idea.

There are two CNN models used in this method, the IONet model for the vehicle and traffic sign detection and the PDNet model for confidence determination and recognition.

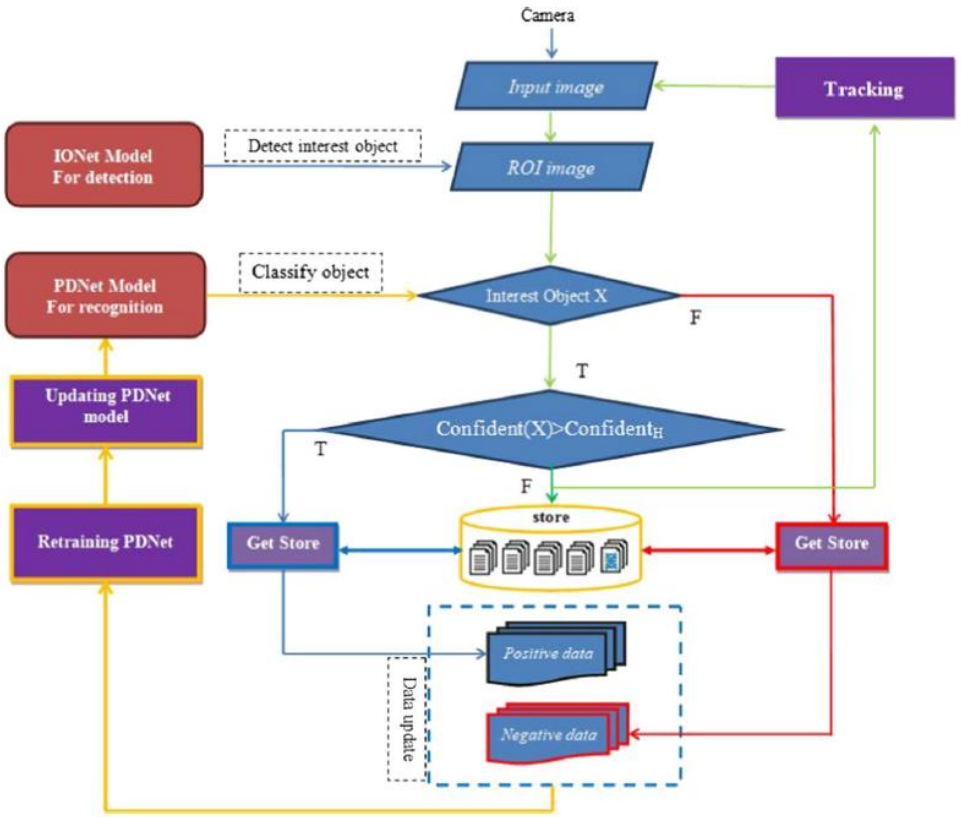


Figure 3.1 General flowchart of the system

Problem description Assume that we have trained two original CNN models, IONet and PDNet with the initial dataset. During the on-the-road journey, ADAS utilizes models to recognize vehicle, traffic sign and make appropriate decisions. However, during processing and recognizing, there are some cases in which the system recognizes vehicle and traffic sign with the low confidence score. This situation occurs when the system encounters data that is not similar to the trained dataset or the information is incomplete. The data is not homologous to original and the noise is often caused by long distance, vehicle and traffic sign obscured by other objects, warped or blurred signs, vehicles moving in conditions of light lack, rain, snow, motion noise, etc. This is the time to launch Adaptive Learning. The system will store images with low confidence score (*IO*) and continue to track (confident tracking) objects. The tracking process aims to identify cases: (i) Lost object; (ii) Negative Object; (iii) Positive Object. When the amount of data in the Positive Data and Negative Data sets is large enough, the retraining model CNN task is processed.

The new trained model is selected and compared with the previously retrained models, the best of which is used to update the recognition model of the system. The Adaptive learning process is ongoing throughout the ADASs working process. Once updated, the new CNN model is able to recognize objects more accurately.

3.2.2.3 Details of the Proposed Architecture

a) Semantic Segmentation for ROI Extraction

Table 3.1 The color map

RGB Color	Objects
[0 255 0]	Other objects: tree, sky,...
[255 123 0]	Traffic sign
[255 0 0]	Road
[0 0 255]	Road side
[255 255 0]	Car
[255 255 0]	Moto
[0 255 255]	Pedestrian

(a) (b)

Figure 3.2 Simulation of training dataset, consisting of (a) original image set and (b) labeled set

The CNN architecture network is proposed for image semantic segmentation using the default architecture of SegNet in [30,31] with 59 layers, input image size [180 360 3]. For convenience in the description processing, this model is proposed to be called IONet. In this problem of building ADAS system, only 6 object groups are of our concern: traffic sign, pedestrians, motorcycle, car (car, coach, truck), road, pavement and other objects (those objects appearing in the image such as buildings, trees, sky, etc.). Labeling is the process of making the annotation mask of

each labeled image with 7 object classes corresponding to 7 RGB color codes (Table 3.1, Figure 3.2).

Once detected by the IONet image semantic segmentation model, the vehicle and traffic sign in the input image will be labeled with the color codes specified in Table 1. The region of interest is the rectangular surrounding the objects. The system will automatically extract the region of interest and transfer the ROI image to the objects recognition of the PDNet model (Figure 3.3).

Figure 3.3 Simulation of extracting Region of interest

The IONet model is capable of segmenting 7 different object types for the purpose of recognizing object areas. However, our method is using IONet to semantically segment the interest area of vehicle (Motorcycle, car, coach, truck) and traffic sign in order to determine the confidence score of vehicle and traffic sign classification using PDNet model.

b) CNN Model Classification

In order to classify vehicle and traffic sign, we proposed a SeriesNetwork CNN architecture, which consists of 27 layers, input image size [64 × 64 × 3] as shown in Figure 3.4. This initial model is proposed to be called $PDNet_0$. The accuracy of vehicle and traffic sign recognition, AC_0, is the initial accuracy by using $PDNet_0$. While processing, the system uses adaptive approach to continuously upgrade $PDNet_0, \ldots, PDNet_x$ to $PDNet_1, \ldots, PDNet_{x+1}$ with desired accuracy as following: $AC_0<AC_1<AC_2<AC_3< \ldots AC_x<AC_{x+1}. \ldots <AC_n$ over time, with AC_1, AC_2, $AC_3. \ldots AC_x$, $AC_{x+1}. \ldots AC_n$ is the accuracy value of $PDNet_0$, $PDNet_1$, $PDNet_2$, $PDNet_3 \ldots PDNet_x, PDNet_{x+1}. \ldots PDNet_n$ after being retrained, filtered and updated.

1	'imageinput'	64x64x3 images with 'zerocenter' normalization
2	'conv_1'	128 7x7x3 convolutions with stride [1 1] and padding [2 2 2 2]
3	'relu_1'	ReLU
4	'crossnorm_1'	cross channel normalization with 5 channels per element
5	'maxpool_1'	3x3 max pooling with stride [2 2] and padding [0 0 0 0]
6	'conv_2'	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
7	'relu_2'	ReLU
8	'crossnorm_2'	cross channel normalization with 5 channels per element
9	'maxpool_2'	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
10	'conv_3'	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
11	'relu_3'	ReLU
12	'maxpool_3'	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
13	'conv_4'	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
14	'relu_4'	ReLU
15	'maxpool_4'	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
16	'conv_5'	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
17	'relu_5'	ReLU
18	'maxpool_5'	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
19	'fc_1'	1024 fully connected layer
20	'relu_6'	ReLU
21	'dropout_1'	50% dropout
22	'fc_2'	1024 fully connected layer
23	'dropout_2'	50% dropout
24	'fc_3'	4 fully connected layer
25	'dropout_3'	50% dropout
26	'softmax'	softmax
27	'classoutput'	crossentropyex with 'CamDungDo' and 3 other classes

Figure 3.4 PDNet model structure

Model is proposed to be called $PDNet_0$. The accuracy of vehicle and traffic sign recognition, AC_0, is the initial accuracy by using $PDNet_0$. While processing, the system uses adaptive approach to continuously upgrade $PDNet_0, \ldots, PDNet_x$ to $PDNet_1, \ldots, PDNet_{x+1}$ with desired accuracy as following: $AC_0<AC_1<AC_2<AC_3<\ldots AC_x<AC_{x+1}\ldots<AC_n$ over time, with $AC_1, AC_2, AC_3 \ldots AC_x, AC_{x+1} \ldots AC_n$ is the accuracy value of $PDNet_0$, $PDNet_1$, $PDNet_2$, $PDNet_3 \ldots PDNet_x$, $PDNet_{x+1} \ldots PDNet_n$ after being retrained, filtered and updated.

c) Tracking Object

The tracing process commences as ADAS uses the IONet to detect vehicle and traffic sign (IO), as illustrated in Figure 3.5. During object tracking, ADAS will continuously evaluate the accuracy of image recognition in each frame (ROI) obtained with the PDNet model. However, to ensure the processing speed and storage capacity of Store data sets, we recommend that only one in 48 image frames be stored. Image frames are continuously stored in the Store's S_k data set when the accuracy value of the recognition process (evaluated by the PDNet model) is in the range of [**0**, $Confidence_H$]. The recommended threshold value is $Confidence_H$= 70%. For tracking techniques, there are now many possible algorithms such as

Histogram [32], Kalman filter [33], Kanade–Lucas–Tomasi algorithm [34, 37], etc., each of which has different advantages and disadvantages. Nevertheless, within the framework of the proposed model, we use the algorithm of Kanade–Lucas–Tomasi as default instead of specializing into studying the tracking object algorithm.

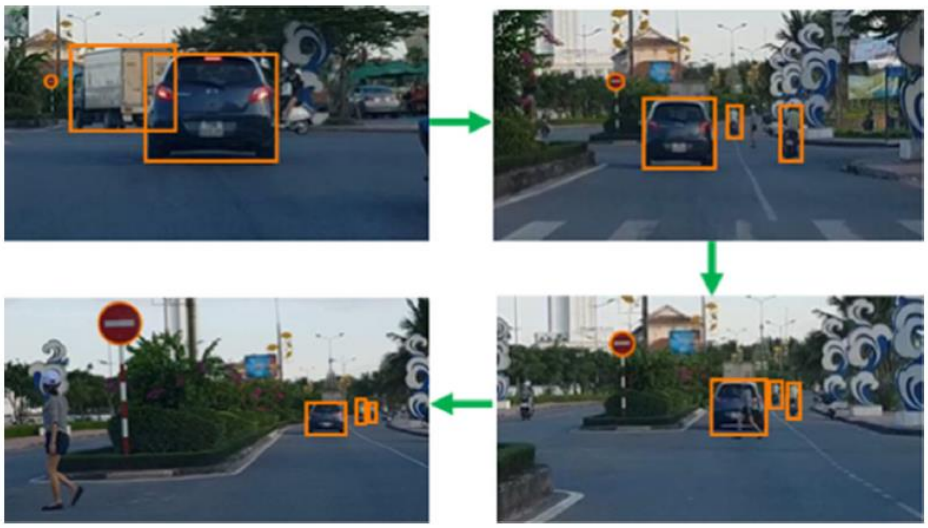


Figure 3.5 Simulation of tracking process of objects

d) Data Storage and Dataset Labeling

In the process of tracking objects, the confidence and accuracy of the traffic sign and vehicle recognition often change. There are three types of S_k data sets that are stored when an object is IO:

(i) The system deletes the S_k data set in the **Store data** (**Lost Object**).

(ii) The S_k data set is moved to the **Negative data** set (**Negative Object**).

(iii) The S_k data set is moved to the **Positive data** set (**Positive Object**).

For a negative object, sometimes object O_{k1} is tracked over time with the confidence value Conf(O_{k1}); however, in with frame, O_{k1} is identified to be O_{k2} with Conf (O_{k2}). Then, there are two situations:

Store data $O_{k2} \in$ **Negative data** | Conf$(O_{k2}) < Confidence_H$

Store data $O_{k2} \in$ **Positive data** | Conf$(O_{k2}) \geq Confidence_H$

e) Model Retraining and Update

When the amount of negative data and positive data reaches a certain threshold N, the system trigs to retrain the recognition CNN model. We use a set of 70% randomly collected data from the previous training data sets and rest received from new data of Positive and Negative data. The purpose of reusing 70% of the previous data is to diversify the training data, enabling the new model to be able to accurately identify objects, avoiding data matching. The model update is based on selecting and evaluating the accuracy of $PDNet_{x-1}$ and $PDNet_x$ with $PDNet_x$ being the current model after retrain. Data collection, self-training, self-recognition and self-updating are ongoing throughout ADAS's operation. This process is non-stop and completely automatic, independent of human intervention.

3.3. Experimental evaluation

3.3.1 Training CNN Model

In order to prove the experimental results of the model, we propose to test it on two different groups of objects: Vehicle and Traffic sign. The two object groups are tested on the same detection model (IONet). However, they are tested on different independent recognition models, structurally identical (Tables 3.2, 3.3). The image datasets (Vehicle and Traffic sign) served as a source for training were derived from the real street images in Dong Hoi city, Quang Binh province, Vietnam.

Table 3.2 The vehicle objects serving recognition by PDNet model

ROI	Name	Description
	Motocycle	Vehicle
	Car	Vehicle
	Coach	Vehicle
	Truck	Vehicle
	No Vehicle	Not vehicle: bicycle, wall, tree, billboard, etc.

Table 3.3 The traffic objects serving recognition by PDNet model

ROI	Name	Description
	No Entry	The traffic sign: No driving beyond this sign
	No parking, stopping	The traffic sign: No parking or stopping beyond this sign
	Non-priority intersection to the right	The traffic sign: Non-priority intersection to the right

Let the vehicle detection be IONet and the the recognition model be PDNet-Vehicle.

Let the traffic sign detection be IONet and the the recognition model be PDNet-TrafficSign.

3.3.1.1 IONet model

The CNN IONet training model with 2133 images is shown in Table 3.4, results in accuracy value as illustrated in Table 3.5 and Table 3.6.

Table 3.4 Images and labels dataset to train $PDNet_1$

Class	Image quantity
Images	2133
Labeled images	2133

Table 3.5 Global accuracy of IONet model

Global Accur	Mean Accur	Mean IoU	Weighted IoU	Mean BFScore
0.93975	0.81079	0.66629	0.89434	0.67542

Table 3. 6 Accuracy of objects of IONet model

Class	Accuracy	IoU	Mean BFScore
Other objects	0.96154	0.94234	0.76156
Road	0.95068	0.91191	0.82045
Pavement	0.82738	0.65641	0.58987
Pedestrian	0.46907	0.29844	0.41667
Car	0.82363	0.65324	0.58445
TrafficSign	0.72029	0.56554	0.59069
Motorcycle	0.92296	0.63615	0.60471

3.3.1.2 PDNet model

The PDNet adaptive model was tested on two independent PDNet-Vehicle and PDNet-TrafficSign models of the two groups Vehicle and Traffic sign. In order to ensure accuracy and reliability, each trained and updated model will be evaluated on a set of independent testing data that has never been used for training. Each testing data set contains data covering possible cases of actual vehicle or traffic sign

identification. The testing data sets of PDNet-TracfficSign model and PDNet-Vehicle model are shown in Tables 3.7, 3.8.

Table 3.7 Image datasets for testing PDNet-TrafficSign model

Category	Image quantity
Motorcycle	406
Car	402
Coach	413
Truck	409
Negative	402

PDNet-Vehicle Training the CNN PDNet-Vehicle model, with 2,700 images is shown in Table 3.9. The training progress is shown in Figure 3.6, results accuracy value as illustrated in Table 3.10.

Table 3.8 Image datasets for testing PDNet-Vehicle model

Category	Image quantity
No entry	277
No parking, stopping	288
NonPITTR	297
Negative	279

Table 3. 9 Image datasets for training PDNet-Vehicle

Category	Image quantity
Motorcycle	600
Car	600
Coach	600
Truck	600
Negative	300

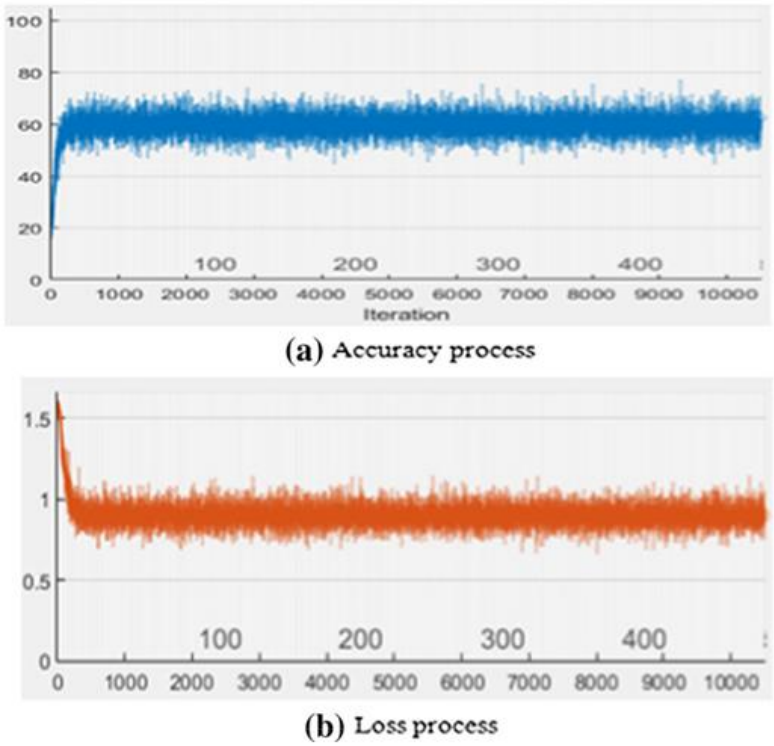


Figure 3.6 Training progress of PDNet-Vehicle$_0$ model

PDNet-TrafficSign Training CNN PDNet-TrafficSign model with 1000 images is shown in Table 3.11. The training progress is shown in Figure 3.7, results in accuracy value as illustrated in Table 3.12.

This is the initial training result of the PDNet model (PDNet-Vehicle model and PDNet-TrafficSign model), which is called *PDNet*$_0$ (*PDNet-Vehicle*$_0$, *PDNet-TrafficSign*$_0$). Since the CNN model is self-learning and self-adaptable, the number of training images and initial accuracy value of the model is not necessarily large, yet must be greater than the recommended value *Confidence*$_H$= 70%. As the Adaptive Learning process is on-going and not much training data is required, there is no need for large hardware configuration. The configuration as shown in Table 3.13 is considered to be enough.

Table 3.10 The confusion matrix of the accuracy of PDNet-Vehicle$_0$ model

	Motor cycle	Car	Coach	Truck	Negative
Motor cycle	**0.5622**	0.0746	0.0995	0.0876	0.2065
Car	0.0847	**0.6247**	0.0266	0.1404	0.1235
Coach	0.0665	0.0246	**0.5911**	0.3030	0.0148
Truck	0.0622	0.0597	0.1244	**0.7164**	0.0373
Negative	0.1834	0.1345	0.0636	0.2910	**0.3276**

Table 3.11 Image datasets for training PDNet-TrafficSign

Category	Image quantity
No entry	290
No parking, stopping	210
NonPITTR	200
Negative	300

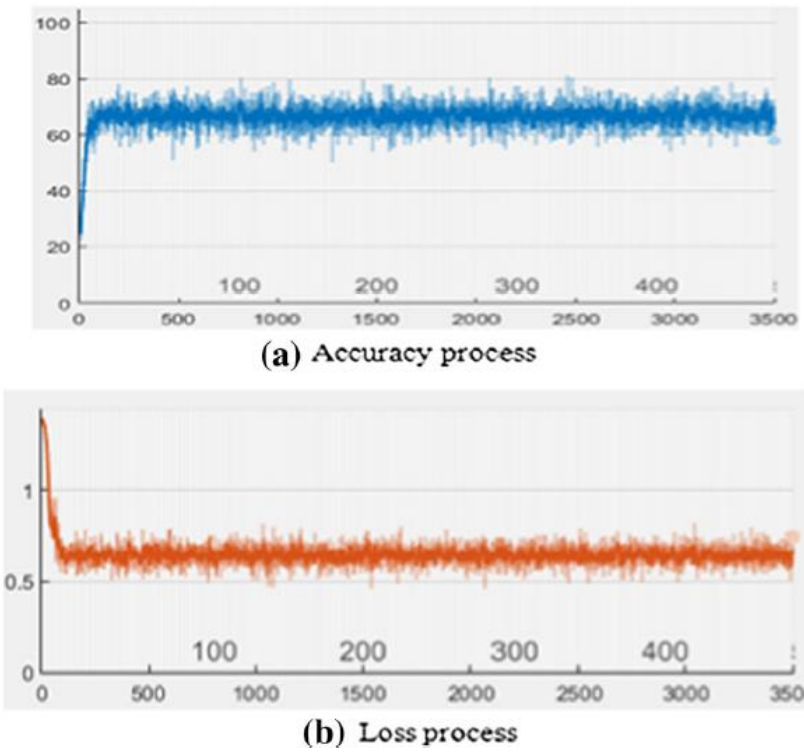


Figure 3.7 Training progress of PDNet-TrafficSign$_0$ model

Table 3.12 The confusion matrix of the accuracy of PDNet-TrafficSign$_0$ model

	No Entry	No parking, stopping	Non PITTR	Negative
No entry	**0.7083**	0.0556	0.0104	0.2257
No parking stopping	0.0108	**0.5776**	0.0758	0.3357
NonPITTR	0	0	**0.8249**	0.1751
Negative	0.0502	0.0072	0.2401	**0.7025**

Table 3.13 The configuration of the device to test the process speed

Device	Description
CPU	I3 3.6 GHz
GPU	Geforce 1060 6 Gb
RAM	16 Gb
HDD	SSD 160 Gb

3.3.2 Retraining and updating model

In the framework of this research, we do not mention the time of data update. The recognition model retraining can be launched based on the number of images collected from positive data and negative data. Alternatively, the recognition model can be updated according to ADAS's timeline.

In this thesis, we use a threshold N = 30% of images from the smallest dataset of all available datasets.

PDNet-Vehicle: N = 90 images.

PDNet-TrafficSign: N = 60 images.

If all of datasets are larger than N, the model retraining will start. Throughout the experiment, the number of images collected in both positive and negative is shown in Table 3.14(for training *PDNet-Vehicle*$_0$ model), Table 3.15 (for training *PDNet-TrafficSign*$_0$), Table 3.16 (for training *PDNet-Vehicle*$_1$ model), and Table 3.17(for training *PDNet-TrafficSign*).

Table 3.14 Image data for retraining PDNet-Vehicle$_0$ model

	Quantity confident tracking	Quantity last model (70%)	Total data for retraining
Motorcycle	186	420	606
Car	195	420	615
Coach	190	420	610
Truck	200	420	620
Negative	90	210	300

Table 3.15 Image data for retraining PDNet-TrafficSign$_0$ model

	Quantity confident tracking	Quantity last model (70%)	Total data for retraining
No entry	64	203	267
No parking stopping	68	147	215
NonPITTR	60	140	200
Negative	72	210	282

The accuracy of recognition models (*PDNet-Vehicle*$_1$, *PDNet-TrafficSign*$_1$ and *PDNet-Vehicle*$_2$, *PDNet-TrafficSign*$_2$)and results after being retrained is shown in Tables 3.18, 3.19, 3.20, 3.21.

There are changes in accuracy when comparing vehicle sign and traffic sign recognition result of the initial model (*PDNet-Vehicle*$_0$, *PDNet-TrafficSign*$_0$) and retrained models (*PDNet-Vehicle*$_1$, *PDNet-TrafficSign*$_1$ and *PDNet-Vehicle*$_2$, *PDNet-TrafficSign*$_2$) on the Retrain dataset (70% data is used for training the last model and 30% data received via Confident tracking object), shown in Figure 3.8.

Table 3.16 Image data for retraining PDNet-Vehicle$_1$ model

	Quantity confident tracking	Quantity last model (70%)	Total data for retraining
Motorcycle	261	424	685
Car	189	431	620
Coach	189	427	616
Truck	190	434	624
Negative	90	210	300

Table 3.17 Image data for retraining PDNet-TrafficSign$_1$ model

	Quantity confident tracking	Quantity last model (70%)	Total data for retraining
No entry	157	187	344
No parking stopping	82	151	233
NonPITTR	60	140	200
Negative	121	197	318

Table 3.18 The confusion matrix of the accuracy of PDNet-Vehicle$_1$ model

	Motor cycle	Car	Coach	Truck	Negative
Motor cycle	**0.6716**	0.1219	0.1368	0.0299	0.0398
Car	0.1017	**0.7482**	0.0242	0.0291	0.0969
Coach	0.0911	0.0197	**0.7709**	0.0764	0.0419
Truck	0.0796	0.0622	0.1642	**0.5473**	0.1468
Negative	0.1320	0.1369	0.0758	0.0733	**0.5819**

Based on improved accuracy, the retrained models (*PDNet-Vehicle*$_1$, *PDNet-TrafficSign*$_1$ and *PDNet-Vehicle*$_2$, *PDNet-TrafficSign*$_2$)will be updated, replacing the initial models (*PDNet-Vehicle*$_0$, *PDNet-TrafficSign*$_0$ and *PDNet-Vehicle*$_1$, *PDNet-TrafficSign*$_1$).

Table 3.19 The confusion matrix of the accuracy of PDNet-TrafficSign$_1$ model

	No Entry	No parking stopping	Non PITTR	Negative Negative
No entry	**0.9097**	0.0208	0	0.0694
No parking stopping	0.0325	**0.8195**	0.0433	0.1047
NonPITTR	0	0	**0.7037**	0.2963
Negative	0.1111	0.0143	0.1362	**0.7384**

Table 3.20 The confusion matrix of the accuracy of PDNet-Vehicle$_2$ model

	Motor cycle	Car	Coach	Truck	Negative
Cycle	**0.7836**	0.1070	0.0697	0.0249	0.0149
Cycle					
Car	0.1090	**0.7458**	0.0218	0.0630	0.0605
Coach	0.0616	0.0148	**0.8547**	0.0443	0.0246
Truck	0.0746	0.0597	0.1070	**0.6119**	0.1468
Negative	0.1002	0.1565	0.0513	0.0269	**0.6650**

Table 3.21 The confusion matrix of the accuracy of PDNet-TrafficSign$_2$ model

	No entry	No parking stopping	NonPITTR	Negative
No entry	**0.9688**	0.0035	0.0069	0.0208
No parking stopping	0.0144	**0.9097**	0.0217	0.0542
NonPITTR	0	0.034	**0.8889**	0.1077
Negative	0.1792	0.0287	0.1577	**0.6344**

3.3.3 Compared results

This section presents some experimental results of our proposed methods and some states of the art method of Deep Learning, such as AlexNet and Vgg. Initial results show that the accuracy of PDNet model is lower than that of AlexNet and Vgg models. However, after Adaptive learning process applied the accuracy of PDNet model is higher than that of the original AlexNet and Vgg model (Figure 3.9). The processing speed of AlexNet and Vgg models is slower than that of PDNet model (Table 3.22), since the PDNet model has a smaller input image size (64 × 64), while the AlexNet and Vgg models have large image sizes (227 × 227 and 224 × 224 respectively).

Our proposed Adaptive Learning method is also applied to AlexNet, Vgg models, whose results show that *Adap-AlexNet*$_1$, *Adap-AlexNet*$_2$ and *Adap-Vgg*$_1$, *Adap-Vgg*$_2$ models (after being retrained) bring higher accuracy than original *AlexNet*$_0$ and *Vgg*$_0$ (Figure 3.10). The results illustrated in the graphs Figure 3.9, Figure 3.10 show that no matter which model is used for training, the Adaptive learning process will improve that original model to bring asymptotically maximum accuracy over time.

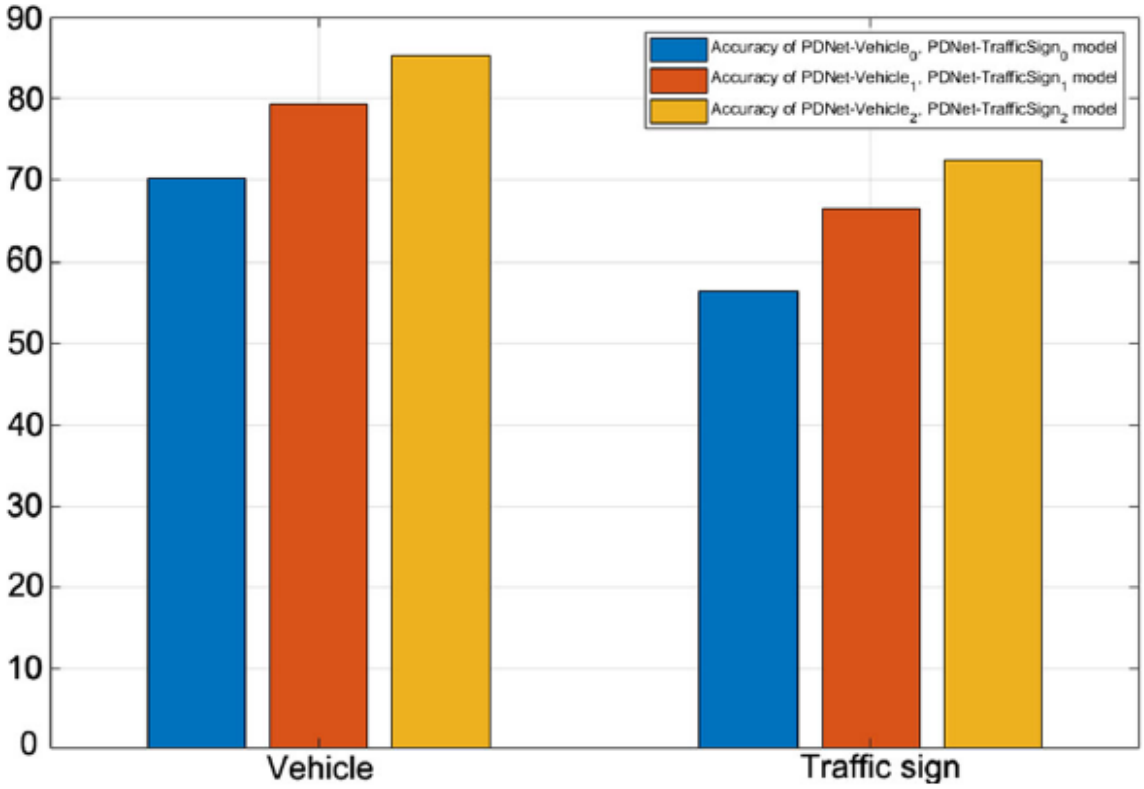


Figure 3.8 Comparing the accuracy of recognition results of retrained Vehicle and Traffic sign model

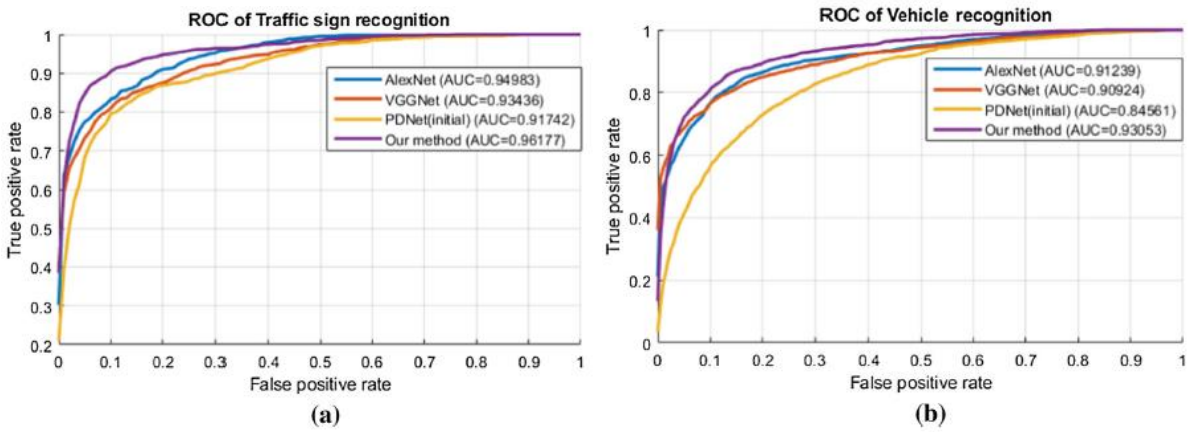


Figure 3.9 Comparison results of our proposed approach and other methods

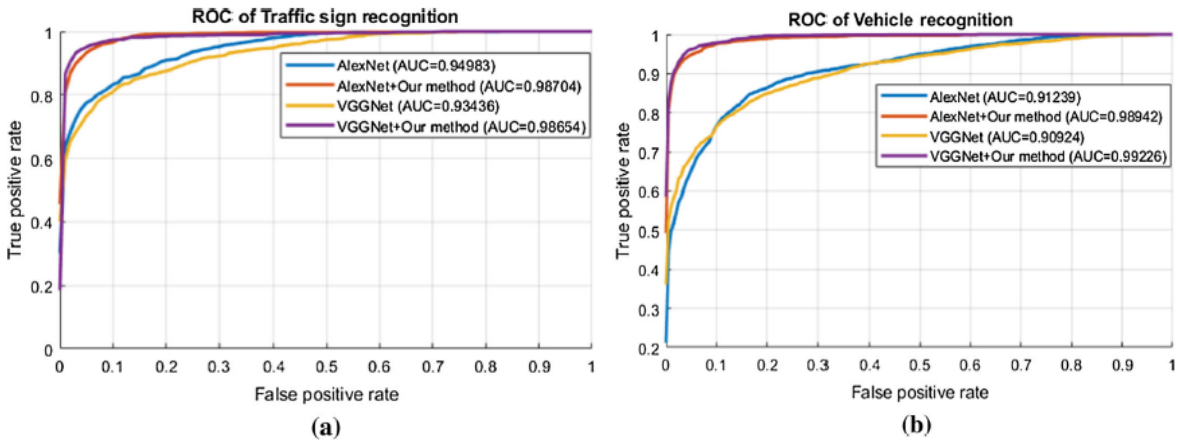


Figure 3.10 Comparison results by applying our Adaptive Learning to other methods

Table 3.22 Comparing the processing speed on traffic sign and vehicle sign between our proposed model and AlexNet, Vgg model

	Time for traffic sign recognition (s)	Time for vehicle recognition (s)
$PDNet0$ (initial model)	0.471	1.657
$PDNet1$ (adaptive model)	0.396	1.674
$PDNet2$ (adaptive model)	0.400	1.627
$AlexNet0$ (initial model)	1.521	3.208
$AlexNet1$ (adaptive model)	1.371	3.225
$AlexNet2$ (adaptive model)	1.511	3.339
$Vgg0$ (initial model)	15.603	28.833
$Vgg1$ (adaptive model)	15.564	28.857
$Vgg2$ (adaptive model)	15.788	29.094

3.4. Conclusion

Although the research is based on the experiments conducted on self-driving car to recognize the objects which are traffic signs and vehicles, the results obtained demonstrate the effectiveness of the proposed Adaptive Learning model. Experimental results show that the proposed model has made significant contributions:

(1) The initialized and trained CNN model is able to increase its accuracy of object detection over time without any interference of specialist.

(2) The CNN model, after being self-trained, selected and updated, is capable of recognizing a variety of different embodiments of objects, improving CNN models intelligence.

(3) Extensive applicability: Not only confined within the framework of an ADAS, but the model is also applicable to all intelligent devices utilizing Deep Learning model.

Although there are some good results, the suggested method may not properly adaptive. Recognition models and training parameters keep the same in the operation of an auto robot system whereas input data continuously change. Thus, Chapter 4 will aim to complete the Adaptive Learning solution by building an Adaptive Learning system of input data and optimizing hyperparameters in training processes. In the Chapter 3, the author mentions the two research works which is paper PP 1.4.

CHAPTER 4. OPTIMIZING HYPERPARAMETERS IN ADAPTIVE LEARNING

In this Chapter, basing on the proposed model mentioned in Chapter 3, the Adaptive Learning solution of algorithms and parameters is continuously studied, improving efficiency and on-road object detection accuracy.

4.1 Problem of optimizing hyperparameters

Currently, studies on artificial intelligence and automotive systems focus on building solutions to optimize Adaptive Learning models and their parameters. Two main focused areas are model selection (e.g., CNN, ANN, LSTM and Segment) and model hyperparameters selection. However, in this thesis, parameter optimization of specific CNN models will be focused instead of training model selection.

In Chapter 2 and 3, current CNN models are trained on two types of parameters:

Optimization hyperparameters

- *Learning rate*
- *Mini-Batch Size*
- *Number of Epochs*
- ...

Model hyperparameters

- *Number of hidden units*
- *First hidden layer*
- *Number of layers*
- ...

Model hyperparameters decide the changes of CNN models and change little during training processes of CNN models. Thus, to solve the problem of adaptive algorithms, solutions are sought to optimize the hyperparameters.

There are three main solutions to hyperparameters optimization of CNN models: Random Search, Gird Search and Bayesian Optimization. Gird Search collect all available hyperparameters, conduct training and select the best hyperparameters in training. The method has the great capability in finding optimized hyperparameters. However, it is only effective with small hyperparameter

domains and datasets due to its slow and time-consuming processes. Random Search operates by randomly selecting hyperparameters and training models, assessing and choosing the best hyperparameters. The method is effective with great hyperparameter domains and datasets and can shorten searching time, but the probability to find optimized hyperparameters is low. Both Random Search and Gird Search direct their search to new areas without consulting the previous searching areas.

Optimizing hyperparameters by Bayesian algorithm can save processing time and optimize the effectiveness of parameter search by building a post distribution function (Gaussian process), which is to assess previous search when choosing hyperparameters for consequent assessment. This way allows Bayesian algorithm to search following parameters in a selective manner, reduce repetitiveness and help to increase searching effectiveness. Specifically, it can skip unnecessary parameters in finding optimized values.

In the suggested model in Chapter 3, after reaching a certain amount of data, CNN model will retrain with 30% old data and 70% new data, thus training database of $PDNet_x$ is different from that of $PDNet_{x+1}$. However, in this model, parameters do not change during training processes.

Therefore, in Chapter 4, an algorithm will be proposed to optimize parameters in each retraining session of PDNet to ensure that after training processes, PDNet model has the best recognition capability.

4.2. Optimization method

In machine learning, **hyperparameter optimization** is the process of choosing a set of optimal value of hyperparameters.

Popular algorithms include:

- Grid search
- Random search
- Bayesian optimization

4.2.1 Grid search

Grid search is the traditional way of searching hyperparameters. Grid search is usually made on the following values:

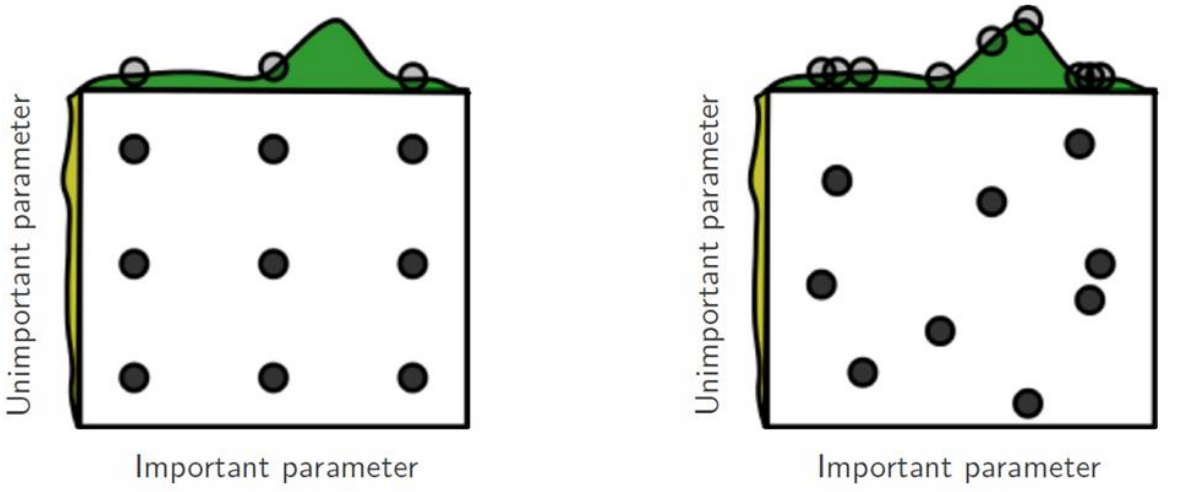
- Learning Rate
- Number of Layers

Algorithm is trained by grid search for all sets by using learning rate and number of layers, typically measured by cross-validation as a technique. This technique ensures the trained model to receive most of the samples from the data set (one of the best methods to measure by using K-Fold Cross Validation which provides rich data for model training and validation). As an algorithm, grid search is simple to use, but it suffers from the curse of dimensionality, causing overflows the device's memory.

4.2.2 Random search

Sampling search space and evaluating data samples from probability distributions are done randomly. For example, instead of trying to test all 100,000 samples, only 1,000 parameters are tested randomly.

However, it is the disadvantage of ransom search algorithm that information from previous tests is not used to select the next set. Moreover, it is difficult to predict next experiments.



a) Standar grid search b) Random search

Figure 4.1 Stimulation of searching way of Hyperparameter values by Grid Search (a) and Random Search (b) (Source: Medium.com)

4.2.3 Bayesian search

Mapping hyperparameter values enable recognition model to optimize its performance on the validator. The model's hyperparametric values are often required to adjust by machine learning algorithms. However, those adjustments are considered *"black functions"* because they are unable to be written into a formula (derivatives of the unidentified function).

A better way to optimize and refine Hyperparameters is to *enable automatic model adjustments* by using Bayesian optimization. The model used to perform approximately the target function is called the substitution model. The Gaussian Process (GP), which is a popular substitution model, is selected to Bayesian optimization. Bayesian optimization's operation is as follows: assuming that an unknown parameter value is sampled from the Gaussian process and "fixing" distribution is conducted for this value later, then there are two main options for performing Bayesian optimization. They are:

- Preselecting presumptive values of parameters need to be optimized, using *Gaussian Process*

- Next, an Acquisition function must be selected that will be used to develop a utility function from the post-processing model, allowing us to locate the next value to evaluate.

Gaussian Process: A Gaussian process, that defines a prior classification on functions, can be converted into a later classification on functions when data is found. The Covariance matrix is used by the Gaussian process, aims to ensure values are close together. The covariance matrix, along with a value of μ, proposes the desired values $f(x)$ that define a Gaussian process.

- *The Gaussian process is used first for Bayesian's inference.*

- *Post computing is used to make predictions for unseen test cases.*

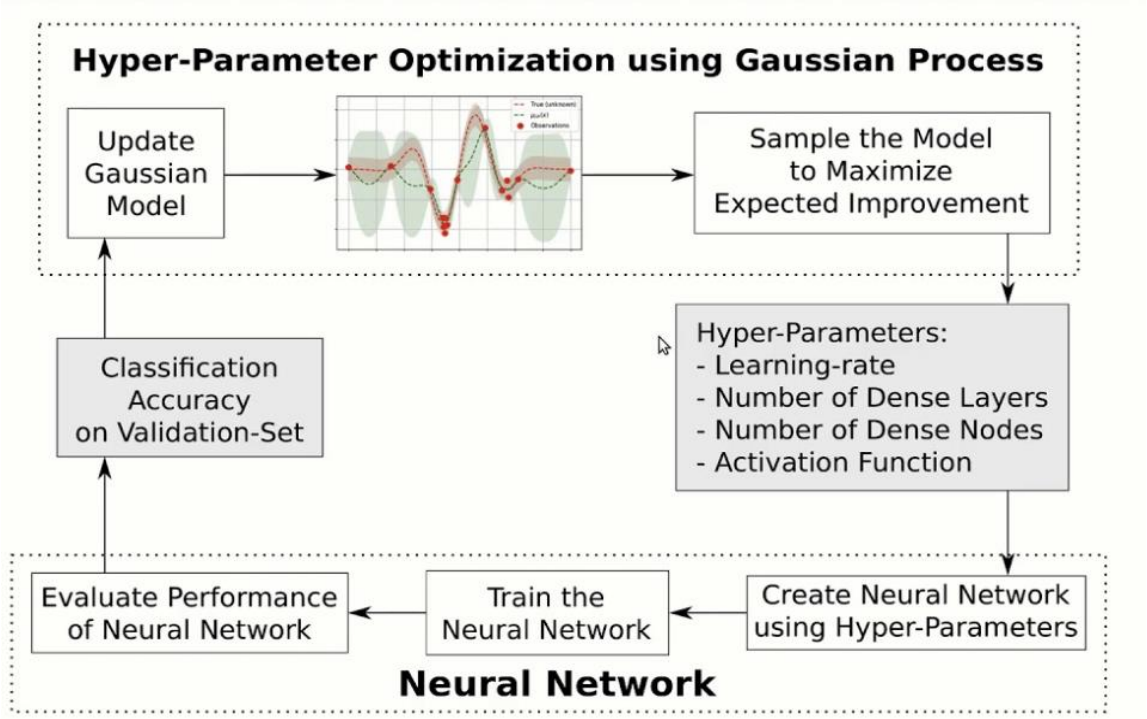


Figure 4.2 Operation model of Bayesian optimization

Acquisition function: Sampling from the search space is conducted by acquisition functions. This maximizes acquisition function for defining the next point to sample. Popular acquisition functions are:

- Maximum Probability of Improvement (MPI)
- Expected Improvement (EI)
- Upper Confidence Bound (UCB)

Expected Improvement (EI) is one of the popular acquisition functions and is defined:

$$EI(x)=\mathbb{E}[\max\{0,f(x)-f(\hat{x})\}]$$

In which, $f(\hat{x})$ is the current set of optimal hyperparameters. Maximization of the hyperparameter values is done thanks to f.

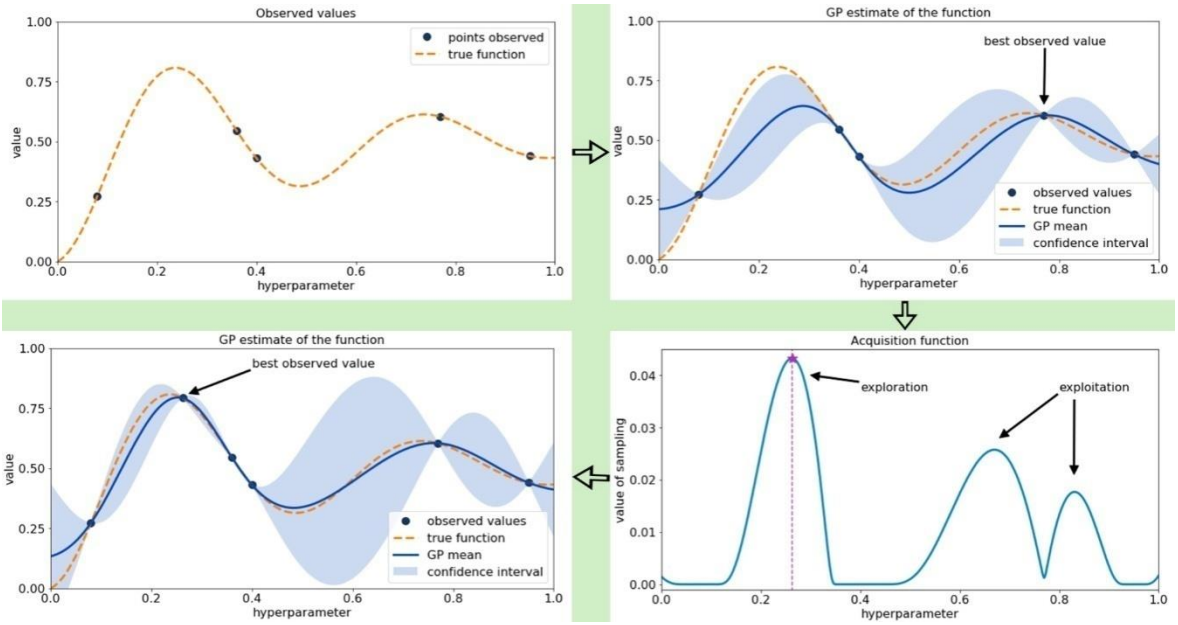


Figure 4.3 Gaussian process (Source: https://www.researchgate.net/profile/Akshara_Rai)

4.3. Suggested solutions

4.3.1. Solution overview

The training model and the proposed solution were inherited from the model proposed in Chapter 3. This proposed method makes a new contribution by changing the Retrained PDNet function block as illustrated in Figure 3.1. The HyperNet function is added to enable the hyperparameter search for the training model, which improves recognition efficiency. The appropriate hyperparameter is automatically solved by the Bayesian approach. The overall proposed method is presented in Figure 4.4

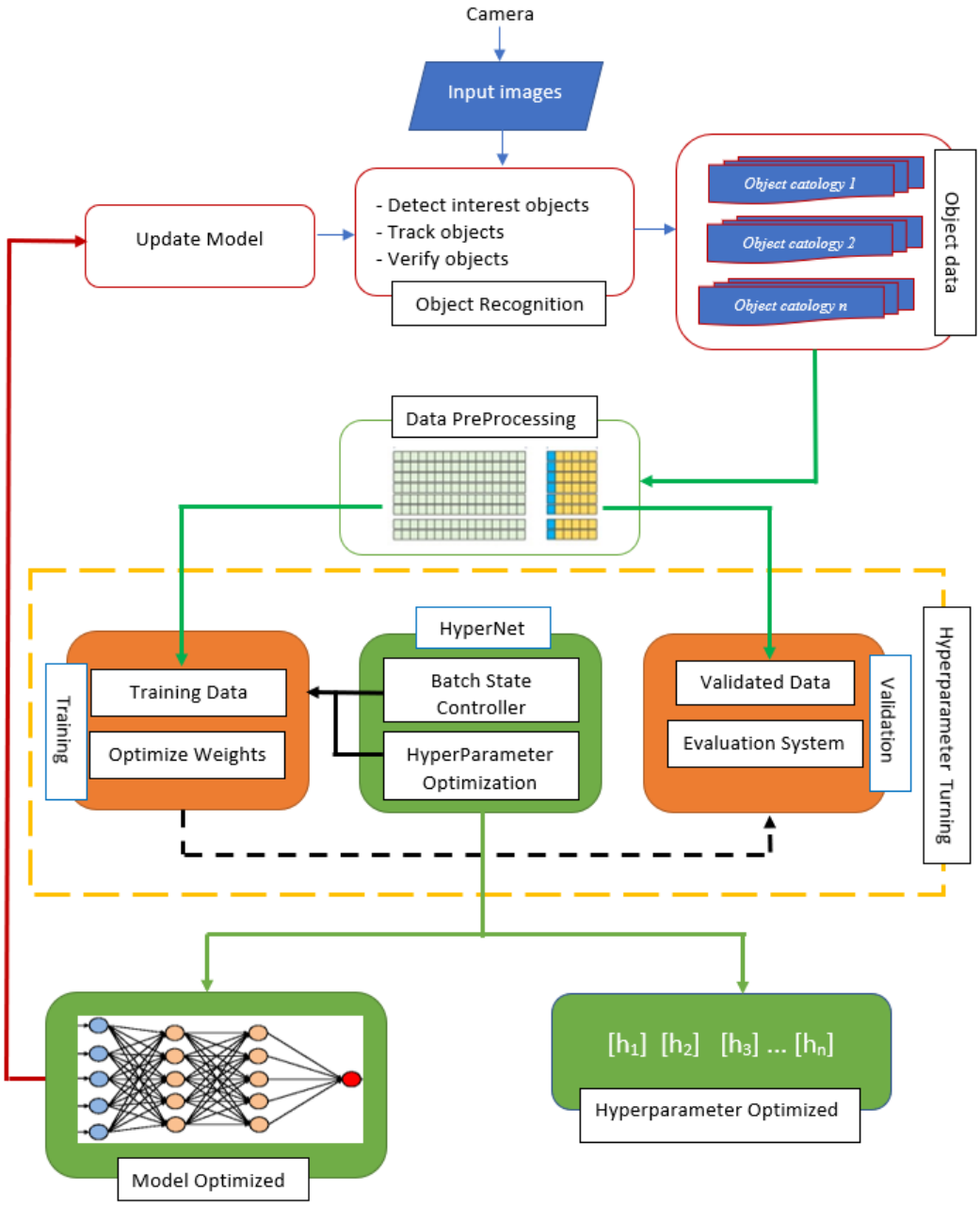


Figure 4.4 Overall proposed model

Furthermore, the data collected during ADAS movement is constantly changing and refreshed. There is no change in the structural parameters of the CNN model and training parameters in the retraining process of the previous CNN model. Therefore in theory, it is necessary to change the architecture of the CNN model and training parameters to ensure that they match with each

new dataset. However, our adaptive solution for retraining the recognition model inherits its ‘intelligence’ from the previous model because of the nature of the proposed solution. Searching and changing the CNN model’s architecture is not suggested. The solution will focus on finding important hyperparameters of the training process. Then, the most equivalent and optimal model is expected to be found.

4.3.2. Analysis

4.3.2.1 PDNet architecture

Table 4.1 PDNet model structure and parameters

Layer	Name	Description
1	‘imageinput’	64x64x3 images with ‘zerocenter’ normalization
2	‘conv_1’	128 7x7x3 convolutions with stride [[1 1] and padding [2 2 2 2]
3	‘relu_1’	ReLU
4	‘crossnorm_1’	Cross channel normalization with 5 channels per element
5	‘maxpool_1’	3x3 max pooling with stride [2 2] and padding [0 0 0 0]
6	‘conv_2’	‘128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
7	‘relu_2’	ReLU
8	‘crossnorm_2’	Cross channel normalization with 5 channels per element
9	‘maxpool_2	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
10	‘conv_3’	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
11	‘relu_3’	ReLU
12	‘maxpool_3’	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
13	‘conv_4’	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
14	‘relu_4’	ReLU
15	‘maxpool_4’	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
16	‘conv_5’	128 7x7x128 convolutions with stride [1 1] and padding [2 2 2 2]
17	‘relu_5’	ReLU
18	‘maxpool_5’	2x2 max pooling with stride [1 1] and padding [0 0 0 0]
19	‘fc_1’	1024 fully connected layer
20	‘relu_6’	ReLU
21	‘dropout_1’	50% dropout
22	‘fc_2’	1024 fully connected layer
23	‘dropout_2’	50% dropout
24	‘fc_3’	4 fully connected layer
25	‘dropout_3’	50% dropout
26	‘softmax’	Softmax
27	‘classoutput’	Crossentropyex with ‘NoEntry’ and 3 other classes

The proposed PDNet architecture remained the same as in Chapter 3. Our architecture was constructed based on a series network of CNN, consisting of 27 layers and an input image size of 64 × 64 × 3. Within the algorithm, the proposed method was able to find the PDNet model with the best architecture that was suitable to each dataset found during ADAS movement. However, because of the PDNet model's inheritance, the architecture of the PDNet remained the same during retraining. In addition, the main advantage of the proposed method was the use of a model with simple architecture that could still attain high accuracy. The Adaptive Learning process will gradually help the model to increase the accuracy and recognition of a variety of objects. The model's 'intelligence' will increase over time and under ADAS movement. Moreover, the simple model architecture helped the retraining process to be shortened in terms of time. The details of the network are presented in Table 4.1.

4.3.2.2 Hyperparameters selection

In the retraining process of the CNN model (PDNet model), many hyperparameters were configured, as shown in Table 4.2. These hyperparameters determined the quality, time, and more of the training process of a CNN model.

In the study, using the Bayesian algorithm was proposed to find adaption and change over six hyperparameters. These included '*InitialLearnRate*', '*L2Regularization*', '*Momentum*', '*MiniBatchSize*', *'GradientThreshold'* and *'GradientThresholdMethod'*. These hyperparameters were the important ones that were capable of changing and adapting to new datasets, directly affecting the accuracy of the post-trained CNN model.

Table 4.2 Hyperparameters in the training process of CNN (Training option)

Hyeperparameters	Description
'Plots'	Plots to display during network training
'Verbose'	Indicator to display training progress information
'VerboseFrequency'	Frequency of verbose printing
'MaxEpochs'	Maximum number of epochs
'MiniBatchSize'	Size of mini-batch
'Shuffle'	Option for data shuffling
'ValidationData'	Data to use for validation during training
'ValidationFrequency'	Frequency of network validation
'ValidationPatience'	Patience of validation stopping
'InitialLearnRate'	Initial learning rate
'LearnRateSchedule'	Option for dropping learning rate during training
'LearnRateDropPeriod'	Number of epochs for dropping the learning rate
'LearnRateDropFactor'	Factor for dropping the learning rate
'L2Regularization'	Factor for L_2 regularization
'Momentum'	Contribution of previous step
'GradientDecayFactor'	Decay rate of gradient moving average
'SquaredGradientDecayFactor'	Decay rate of squared gradient moving average
'Epsilon'	Denominator offset
'ResetInputNormalization'	Option to reset input layer normalization
'GradientThreshold'	Gradient threshold
'GradientThresholdMethod'	Gradient threshold method
'SequenceLength'	Option to pad, truncate, or split input sequences
'SequencePaddingDirection'	Direction of padding or truncation
'SequencePaddingValue'	Value to pad input sequences
'ExecutionEnvironment'	Hardware resource for training network
'WorkerLoad'	Parallel worker load division
'DispatchInBackground'	Use background dispatch
'CheckpointPath'	Path for saving checkpoint networks

4.3.2.3 HyperNet processing

Regarding the training CNN model, the task of finding and configuring parameters for training is a complicated and time-consuming process. With a certain CNN model architecture and a certain dataset, searching and selecting the training parameter set that best suits the existing model and dataset is required. Among many hyperparameter search solutions, this paper proposes using the Bayesian optimization search solution [109, 110]. Bayesian optimization is a suitable algorithm for optimizing the hyperparameters of classification and regression

models. Bayesian optimization can be used to optimize the complex, discontinuous and time-consuming evaluation process.

Of which, Sequential Model-Based Optimization (SMBO) is seen as a highlight hyperparameter search algorithm with its validated effectiveness [111]. Technically, the idea of this solution is that setting a representation model named M over the target black-box, then this model is updated sequentially by querying f(θ) at new locations to optimize the acquisition function throughout Expected Improvement (EI) which is calculated as follows:

$$EI_{y*}(\theta, M) = \int_{-\infty}^{y*} max(y^* - y, 0) p_M(y|\theta) dy \quad (1)$$

y and y* present the real values and corresponding thresholds. The SMBO algorithm in details is shown in Algorithm 1. The operating model of the Bayesian optimization algorithm is demonstrated in Figure 4.5

Algorithm 1: Sequential Model-Based Optimization
SMBO($f, \theta,$ Θ, T)

Data: Target function f, Hyperparameters θ
Search Space Θ, Iteration T
Result: Best Hyperparameters θ^*
Initialize model M
$H \leftarrow \emptyset$
For t ← 1 to T **do**
$\theta^* = \underset{\theta \in \Theta}{argmax} EI(\theta, M_{t-1})$;
Evaluate f(θ^*);
$H \leftarrow H \cup \left(\theta^*, f(\theta^*)\right)$;
Fit model $M_t by H$
end
Return $\theta^* given H$

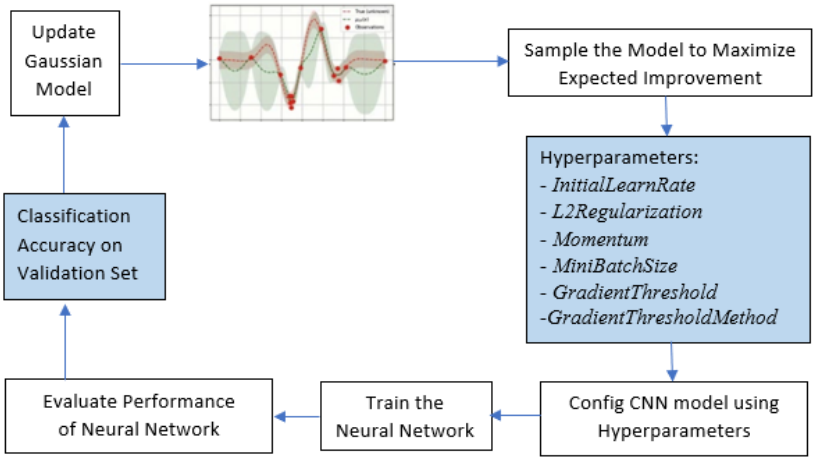


Figure 4.5 Operating model of the Bayesian algorithm

4.4. Experimental evaluation

For object recognition, ADAS uses the IONet recognition model (semantic segmentation model), which was proposed in Chapter 3. IONet is an independent CNN model that remains unchanged during the ADAS performance process. However, discussion of this model is beyond the purview of the current research. When an object is identified by the IONet model with certain reliability, the system will perform tracking and recognition using the PDNet model.

In order to assess the overall experimental process of the proposed solution, two different groups of objects were experimented on. They included vehicles and traffic signs, as described in Table 4.3 and Table 4.4, respectively. The vehicle

Table 4.3 The object for PDNet model recognition

ROI	Name	Description
	No Entry	The traffic sign: No driving beyond this sign
	No parking, stopping	The traffic sign: No parking or stopping beyond this sign
	Non-priority intersection to the right	The traffic sign: Non-priority intersection to the right

recognition model is referred to as PDNet-Vehicle, and the recognition model of traffic signs is PDNet-TrafficSign. To ensure accuracy and reliability, post-trained PDNet-Vehicle and PDNet-TrafficSign models were evaluated on an independent test dataset that had never been used before. Each test dataset contained data that included possible cases of actual vehicles and traffic signs. The test dataset for vehicles and traffic signs is shown in Tables 4.5 and 4.6. To retrain PDNet models, the use of the OptionTrain training parameter set corresponding to each type of object is suggested: OptionTrain_Vehicle and OptionTrain_TrafficSign.

Table 4.5 The object for PDNet model recognition

ROI	Name	Description
	Motorcycle	Vehicle
	Car	Vehicle
	Coach	Vehicle
	Truck	Vehicle
	No Vehicle	Not vehicle: bicycle, wall, tree, billboard, etc.

Table 4.4 Image datasets for testing the PDNet-Vehicle model

Dataset name	Quantity
Motorcycle	406
Car	402
Coach	413
Truck	409
Negative	402

Table 4. 6Image datasets for testing PDNet-TrafficSign model

Dataset name	Quantity
No Entry	277
No parking, stopping	288
Non-priority intersection to the right	297
Negative	279

Among the hyperparameters in Table 4.2, the selection of only six important parameters directly affecting the accuracy of the model and training time was proposed. Selecting too many hyperparameters would not provide a real idea of the process due to the amount of time consumed if the Bayesian algorithm was expected to work optimally. In contrast to the limited time and large number of

Table 4.7 Parameter domain values

Hyperparameters	Values range
InitialLearnRate	[0.0001 0.9]
Momentum	[0.1 0.99]
L2Regularization	[0.0001 0.01]
MiniBatchSize	[64 256]
GradientThreshold	[10 10^{15}]
GradientThresholdMethod	['l2norm', 'global-l2norm', 'absolute-value']

hyperparameters to be searched, the Bayesian algorithm would fail to choose a truly optimal hyperparametric solution. In addition, the search domain of six hyperparameters was also considered and adjusted appropriately, avoiding the wide parameter domain that affects accuracy and search time. From the experimental process, the proposed parameter value domain of six hyperparameters is shown in Table 4.7.

Table 4.8 The configuration of the device

Device	Description
CPU	I3 3.6 GHz
GPU	Geforce 1060 6 Gb
RAM	16 Gb
HDD	SSD 160 Gb

To illustrate the search process of hyperparameter value and training the PDNet model, a minimum hardware system was used while still ensuring the processing time of the system, as shown in Table 4.8.

4.4.1 Training the initial PDNet model

The initial PDNet architecture (Table 4.1) is initialized and trained PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$ using the PDNet-Vehicle data and PDNet-TrafficSign data correspondence, as shown in Table 4.9, Table 4.10. The unified dataset in Table 4 and Table 5 were used for evaluation on both the optimal parameters using Bayesian algorithm and accuracy testing of the trained PDNet model. The accuracy of the initial PDNet-Vehicle and PDNet-TrafficSign models is shown in Figure 4.6.

Table 4.9 Image datasets for training initial PDNet-Vehicle

Dataset name	Quantity
Motorcycle	600
Car	600
Coach	600
Truck	600
Negative	300

Table 4.10 Image datasets for training initial PDNet-TrafficSign

Dataset name	Quantity
No Entry	290
No parking, stopping	209
Non-priority intersection to the right	199
Negative	306

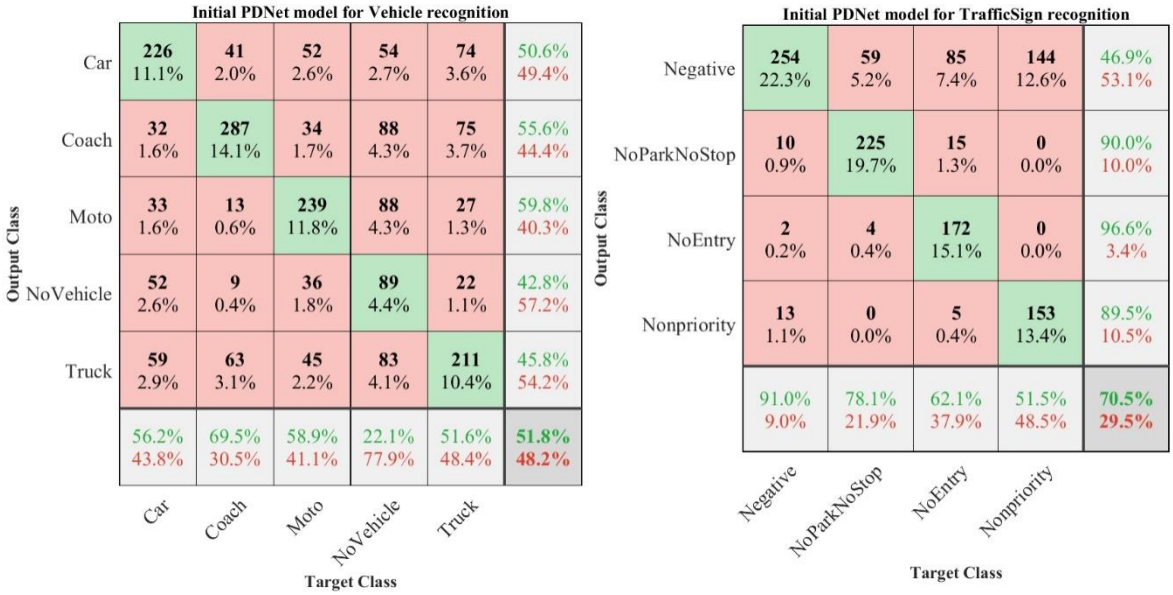


Figure 4.6 The confusion matrix of the accuracy of initial PDNet-Vehicle and PDNet-TrafficSign model

Trained PDNet-Vehicle and PDNet-TrafficSign models are named as PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$.

4.4.2 Optimization of learning parameters, update PDNet model

During ADAS movement, IONet model and PDNet model were used by the system to recognize and acquire continuously new image data from objects. Once the number of images reaches N value, the system will initiate a Pre-retrain dataset with X% of the dataset coming from the newly acquired image and [100-X]% from the previous training dataset. Using an old data part aims at avoiding Overfitting. From the experiment, value at X= 30% of the number of images of the previous smallest dataset is proposed.

For PDNet-Vehicle: N = 30% x 277 = 90 images.

For PDNet-TrafficSign: N = 30% x 277 = 90 images.

Data for re-training PDNet-Vehicle$_0$ (referred to as Data-Vehicle$_0$) and PDNet-TrafficSign$_0$ (referred to as Data-TrafficSign$_0$) are shown in Table 4.11, Table 4.12 and Data-Vehice$_0$ and Data-TrafficSign$_0$ are the new ones, thus it is needed to search hyperparameters (OptionReTrain_Vehicle$_0$ and OptionReTrain_TrafficSign$_0$) for PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$ models that are fixed with these new ones.

Table 4.12 Image data (Data-Vehicle$_0$) for searching hyperparameters and the PDNet-Vehicle$_1$ model

Class	Quantity of confident tracking	Quantity of last model (70%)	Total data for retraining
Motorcycle	186	420	606
Car	195	420	615
Coach	190	420	610
Truck	200	420	620
Negative	90	210	300

Table 4.11 Image data (Data-TrafficSign$_0$) for searching hyperparameters and the PDNet-TrafficSign$_1$ model

Class	Quantity of confident tracking	Quantity of last model (70%)	Total data for retraining
No Entry	91	203	294
No parking, stopping	70	147	217
Non-priority intersection to the right	60	140	200
Negative	108	210	318

The Bayesian algorithm was used to search for hyperparameters with architecture of fixed CNN model (PDNet model, Table 4.1), MaxObjectiveEvaluations = 60, MaxEpochs = 200 and the value domain of hyperparameters need to be found, as shown in Table 4.7. From the experimental process, it is recommended that MaxEpochs = 200 is appropriate, ensuring the training time and the accuracy of the model. The model found by the Bayesian algorithm is just the trained model on the optimal parameters. Therefore, it is not necessary to use hyperparameters to retrain PDNet models. The algorithm to search

hyperparameters which is set up in Matlab language used for Traffic signs using PDNet model includes the following basic parts:

Set up the values of the initial variables:

```
1.  ModelPath='D:\PHUC\Paper09\NewModel_30_3\PDnet\';
2.  DataPath='D:\PHUC\Paper09\Data\TrafficSign\';
3.  ModelName='PDNet_TrafficSign0';
4.  ModelSaveName='PDNet_TrafficSign1';
6.  validationSet = imageSet(strcat(DataPath,'Test'),
'recursive');
7.  [XValidation,YValidation]= ReadSample(validationSet);
8.  PDnet=load(strcat(ModelPath,ModelName));
9.  Layers=PDnet.PDnet.Layers;
10. trainSet = imageSet(strcat(DataPath,'Train\Train1'),
'recursive');
11. [XTrain,YTrain]=ReadSample(trainSet);
```

Select and config the value domain of the hyperparameters:

```
1. optimVars = [
    optimizableVariable('InitialLearnRate',[0.0001
0.9],'Transform','log')
    optimizableVariable('Momentum',[0.1 0.99])
    optimizableVariable('L2Regularization',[0.0001
0.01],'Transform','log')
optimizableVariable('MiniBatchSize',[64
256],"Type","integer")
    optimizableVariable('GradientThreshold',[10
10^15],"Type","integer",'Transform','log')

optimizableVariable('GradientThresholdMethod',{'l2norm','g
lobal-l2norm','absolute-value'},"Type","categorical")];
```

The main algorithm searches hyperparameters basing on the newly collected data setof ADAS system for PDNet recognition model

```
1.  ObjFcn =
makeObjFcn(XTrain,YTrain,XValidation,YValidation,Layers);
2.  BayesObject =
bayesopt(ObjFcn,optimVars,...'MaxTime',14*60*60,'MaxObject
iveEvaluations',60,'IsObjectiveDeterministic',false,'UsePa
rallel',false);
3.  BestIdx = BayesObject.IndexOfMinimumTrace(end);
4.  fileName = BayesObject.UserDataTrace{BestIdx};
5.  savedStruct = load(fileName);
6.  valError = savedStruct.valError;
7.  Op=savedStruct.options;
8.
A=[Op.InitialLearnRate,Op.L2Regularization,Op.Momentum,Op.
MiniBatchSize,Op.GradientThreshold,
Op.GradientThresholdMethod];
9.  PDnet=savedStruct.trainedNet;
10. [YPredicted,probs] = classify(PDnet,XValidation);
11. Acc=mean(YPredicted == YValidation);
12. testError = 1 - Acc;
13. [Acc testError]
14. fileName=strcat(ModelPath,ModelSaveName,'.mat');
save(fileName,'Op','PDnet','XValidation','YPredicted','Acc
```

Objective Function for Optimization: Train the network and plot the training progress during training. Create a file name containing the validation error and save the network, validation error, training options to disk. The objective function returns |fileName| as an output argument, and |bayesopt| returns all the file names in |BayesObject.UserDataTrace|. The additional required output argument |cons| specifies constraints among the variables.

```
1.  function ObjFcn =
makeObjFcn(XTrain,YTrain,XValidation,YValidation,Layers)
    ObjFcn = @valErrorFun;
2.function [valError,cons,fileName] = valErrorFun(optVars)
3.      numClasses = numel(unique(YTrain));
4.      options = trainingOptions('sgdm', ...
'InitialLearnRate',optVars.InitialLearnRate,
'Momentum',optVars.Momentum, ...
```

```
'L2Regularization',optVars.L2Regularization,
'MiniBatchSize',optVars.MiniBatchSize, ...
'GradientThreshold',optVars.GradientThreshold,
'GradientThresholdMethod',string(optVars.GradientThreshold
Method),
'MaxEpochs',200, ...
'LearnRateSchedule','piecewise', ...
'LearnRateDropPeriod',8, ...
'LearnRateDropFactor',0.1, ...
'Verbose',false, ...
'Plots','training-progress');%,...
5.       trainedNet =
trainNetwork(XTrain,YTrain,Layers,options);
6.
close(findall(groot,'Tag','NNET_CNN_TRAININGPLOT_FIGURE'))
7.       YPredicted = classify(trainedNet,XValidation);
8.       valError = 1 - mean(YPredicted == YValidation);
9.       fileName = "Temp/"+num2str(valError) + ".mat";
10.      save(fileName,'trainedNet','valError','options');
11.      cons = [];
12.   end
13.   end
```

The result obtained from the search algorithm is the PDNet model which was trained on a set of six found optimal hyperparameters(Table 4.13):

```
+ optVars.InitialLearnRate
+ optVars.Momentum
+ optVars.L2Regularization
+ optVars.MiniBatchSize
+ optVars.GradientThreshold
+ optVars.GradientThresholdMethod
```

This PDNet model is used for further searching and identifying subsequent values.

The graph displays the Bayesian function's objective value evaluated on objective function evaluations is shown in Figure 4.7. The confusion matrix for test data in the search process of hyperparameter values is shown in Figure 4.8 (PDNet-TrafficSign). Table 4.13 presents the results of searching for hyperparameter values of the PDNet-Vehicle and PDNet-TrafficSign models.

Table 4.13 Found optimal hyperparameter values of PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ model

Hyperparameters	OptionReTrain_Vehicle$_0$	OptionReTrain_TrafficSign$_0$
InitialLearnRate	0.0018	0.0011
Momentum	0.8574	0.9517
L2Regularization	0.0011	1.0991e-04
MiniBatchSize	75	200
GradientThreshold	7.6048e+14	5.0344e+14
GradientThresholdMethod	l2norm	absolute-value

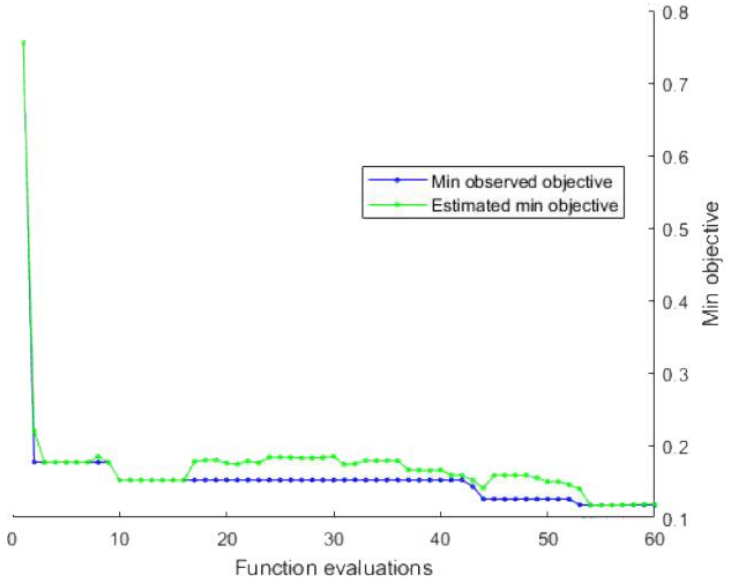


Figure 4.7 The Bayesian function's objective value evaluated on objective function evaluations

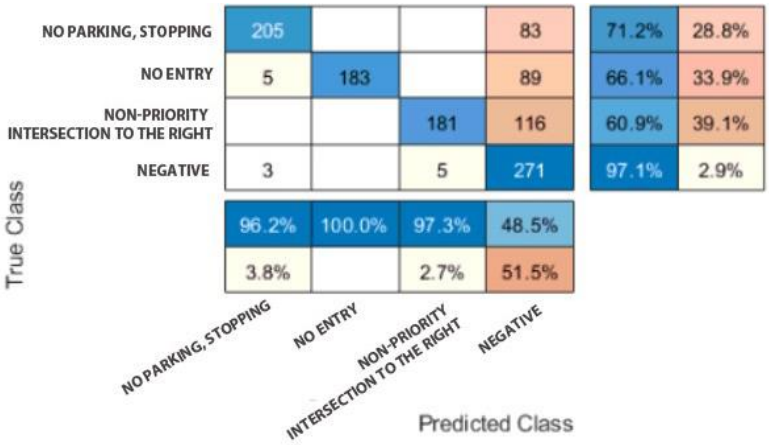


Figure 4.8 The confusion matrix for test data in the search process of optimal hyperparameter and model

The Bayesian algorithm was applied to estimate the optimal hyperparameters for training PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ models using PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$. The confusion matrix of the accuracy of PDNet-Vehicle$_1$, PDNet-TrafficSign$_1$ model is shown in Figure 4.9 with the accuracy of 62.3% and 85.2%. It can be seen from the evaluation results on the same dataset that recognition accuracy of PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ is higher than those of PDNet-Vehicle$_0$, PDNet-TrafficSign$_0$. Thus, these models are in turn updated and replaced the old recognition model in the ADAS.

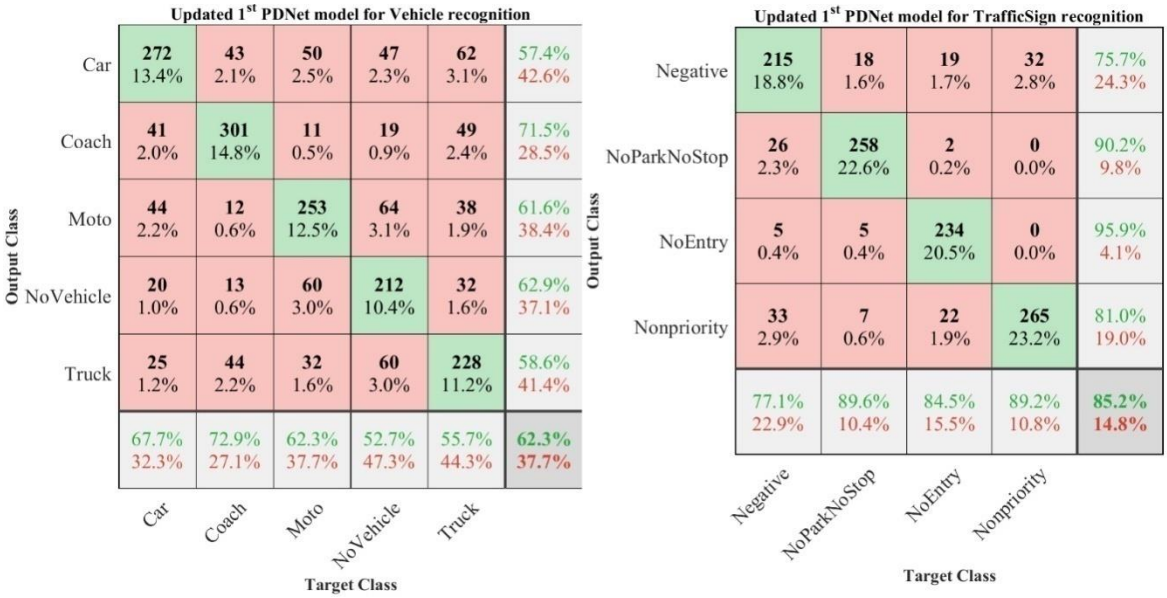


Figure 4.9 The confusion matrix of the accuracy of PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ model

The ADAS continues to recognize and acquire new data of objects over time.

The training dataset is created as the number of images reaches the N value. The data set for retraining PDNet-Vehicle$_1$ (Data-Vehicle$_1$) and PDNet-TrafficSign$_1$ (Data-TrafficSign$_1$) models is shown in Table 4.14 and Table 4.15.

It is continuous to search the hyperparameters (OptionReTrain_Vehicle$_1$ and OptionReTrain_TrafficSign$_1$) that match the newly found datasets for PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$. The search result of optimal hyperparameter value and model by using Bayesian algorithm is shown in Table 4.16.

The PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ models, that were that were

Table 4.16 Image data (Data-Vehicle$_1$) for searching hyperparameters and the PDNet-Vehicle$_2$ model

Class	Quantity of confident tracking	Quantity of last model (70%)	Total data for retraining
Motorcycle	260	425	685
Car	189	431	620
Coach	189	427	616
Truck	190	434	624
Negative	90	210	300

Table 4.15 Image data (Data-TrafficSign$_1$) for searching hyperparameters and the PDNet-TrafficSign$_2$ model

Class	Quantity of confident tracking	Quantity of last model (70%)	Total data for retraining
No Entry	138	206	344
No parking, stopping	81	152	233
Non-priority intersection to the right	60	140	200
Negative	95	223	318

Table 4.14 Found optimal hyperparameter values of PDNet-Vehicle$_2$ and PDNet-TrafficSign$_2$ model

Hyperparameters	OptionReTrain_Vehicle$_1$	OptionReTrain_TrafficSign$_1$
InitialLearnRate	0.0030	1.1968e-04
Momentum	0.9115	9747
L2Regularization	0.0034	0.0068
MiniBatchSize	247	79
GradientThreshold	621	5.2430e+12
GradientThresholdMethod	absolute-value	absolute-value

searched (with Bayesian algorithm) are referred to as PDNet-Vehicle$_2$ and PDNet-TrafficSign$_2$. The confusion matrix of the accuracy of PDNet-Vehicle$_2$, PDNet-TrafficSign$_2$ model is shown in Figure 4.10, the accuracy is at 70.0% and 92.9%.

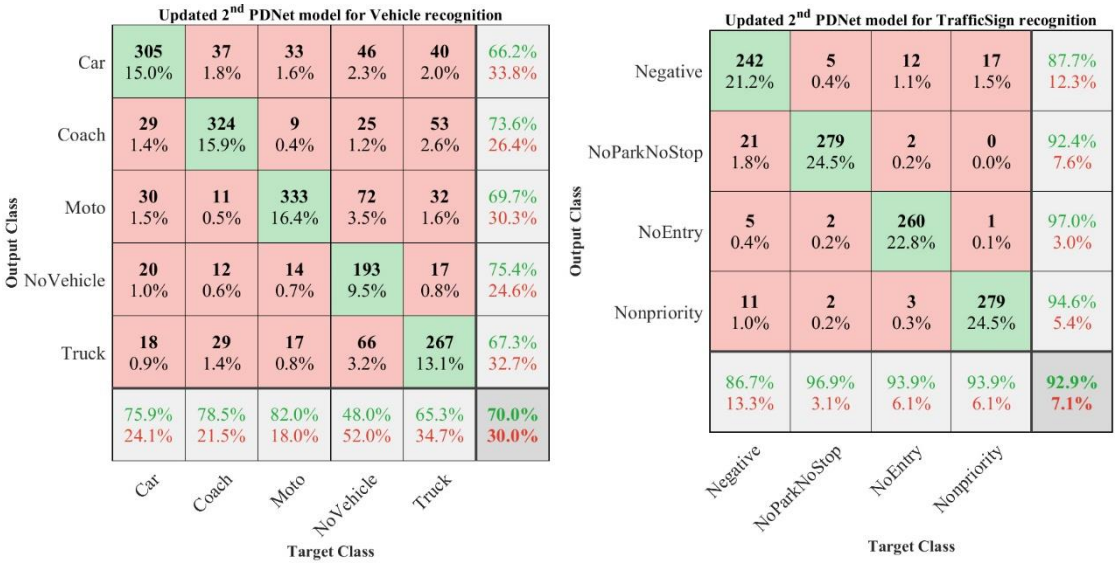


Figure 4.10 The confusion matrix of the accuracy of PDNet-Vehicle$_2$ and PDNet-TrafficSign$_2$ model

The intelligence of ADAS has been constantly improved without any human interference thank to its continuous operation during the on-the – road journey.

There are changes in model accuracy when comparing vehicle and traffic sign recognition result of the initial model (PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$) and optimal models (PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ and PDNet-Vehicle$_2$ and PDNet-TrafficSign$_2$) shown in Figure 4.11.

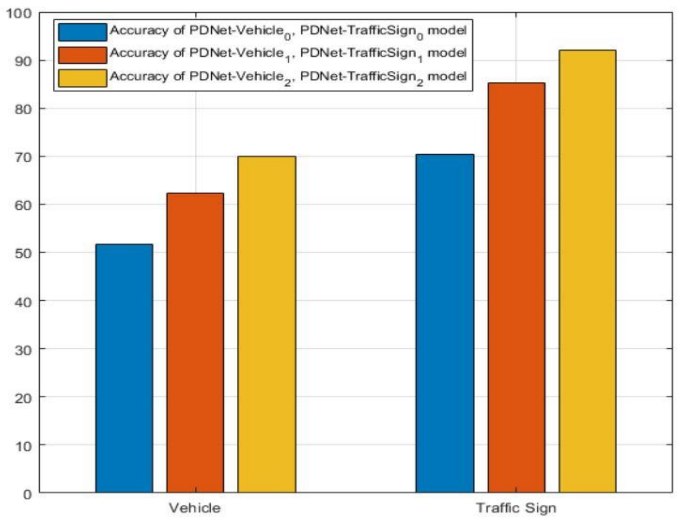


Figure 4.11 Comparing the accuracy of recognition results of Vehicle and Traffic sign model

4.4.3 Compare with the state - of – the - art models

Basing on the results obtained through the Adaptive Learning emulation of PDNet-Vehicle and PDNet-TrafficSign models, these models were continuously applied and compared the state - of – the - art of Deep Learning such as AlexNet and Vgg models. These models are trained and evaluated on the same data set. At first, the recognition results demonstrate that the accuracy on recognition of PDNet model is lower than those of AlexNet and Vgg models. Those of PDNet-Vehicle$_1$, PDNet-TrafficSign$_1$ and PDNet-Vehicle$_2$, PDNet-TrafficSign$_2$, then, is in turn higher than AlexNet and Vgg models or equivalent to these models after getting the Adaptive Learning, shown in Table 4.17 (the automatic retraining system using the hyperparameters that match the new dataset).

To demonstrate the proposed solution’s effectiveness, this solution was applied to AlexNet and Vgg models. PDNet model was replaced by AlexNet and Vgg models which get the Adaptive Learning by data sets in turn. The result, as shown in Figure 4.12, Figure 4.13, Figure 4.15, Figure 4.16, proves that the accuracy of AlexNet$_2$ and Vgg$_2$ models is higher than both the accuracy of AlexNet$_1$, Vgg$_1$ models and of the initial AlexNet, Vgg models. The chart in Figure

4.14, Figure 4.17 show the increasing accuracy on recognition of AlexNet and Vgg model after the recognition model was updated with optimal hyperparameters applied.

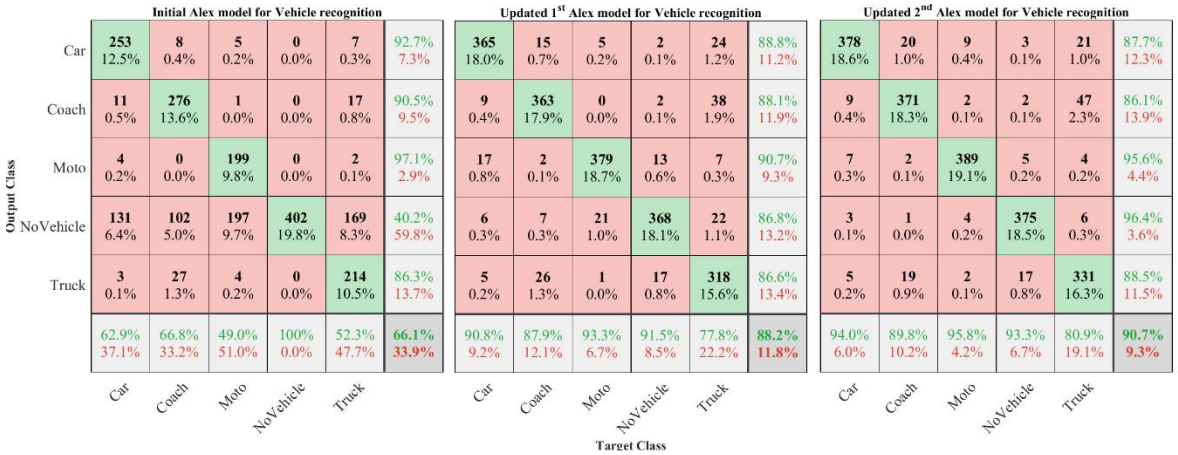


Figure 4.12 The confusion matrix of the accuracy of AlexNet model for vehicle recognition

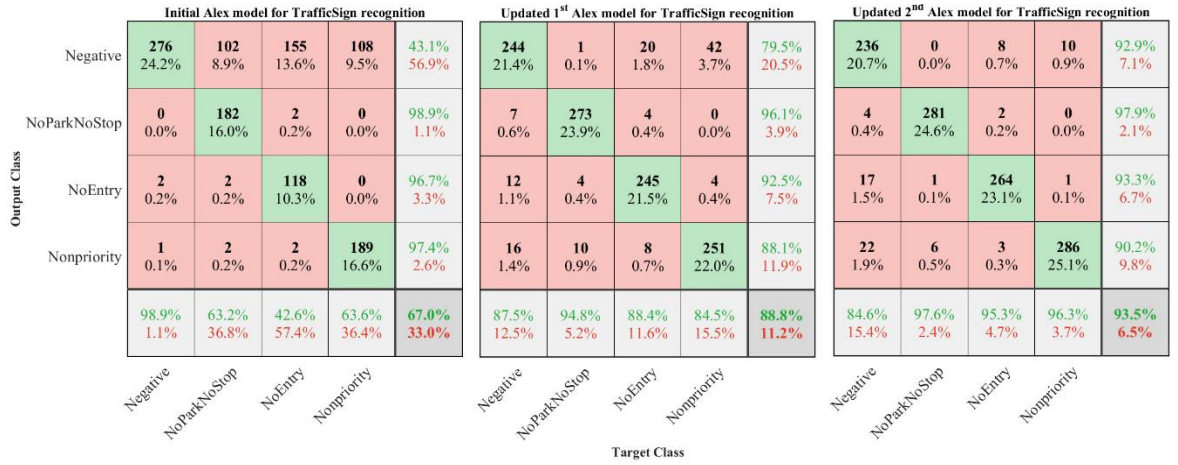
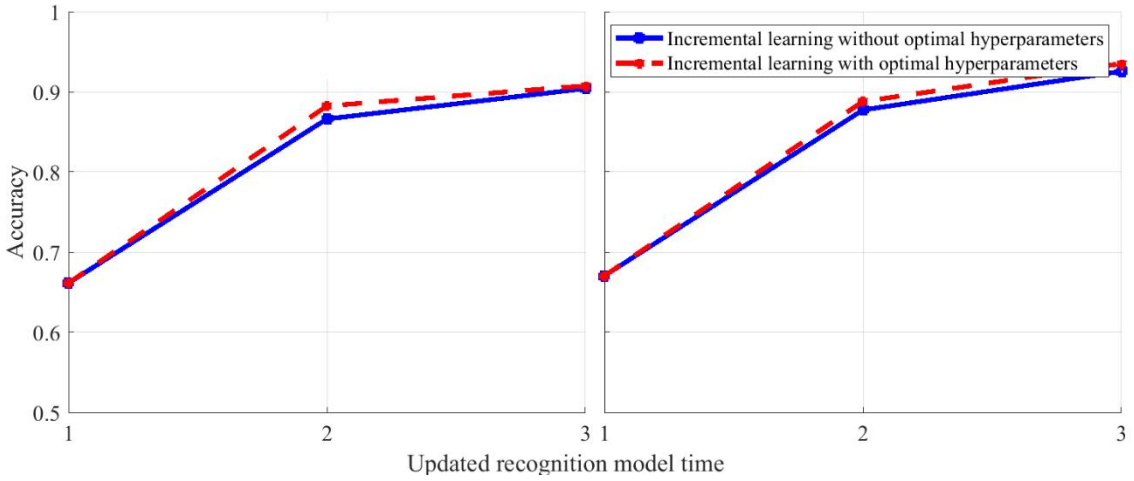


Figure 4.13 The confusion matrix of the accuracy of AlexNet model for traffic sign recognition



(a) Vehicle object (b) Traffic sign object

Figure 4.14 The chart showing the increasing accuracy on recognition of AlexNet model after the updated recognition model with optimal hyperparameters applied

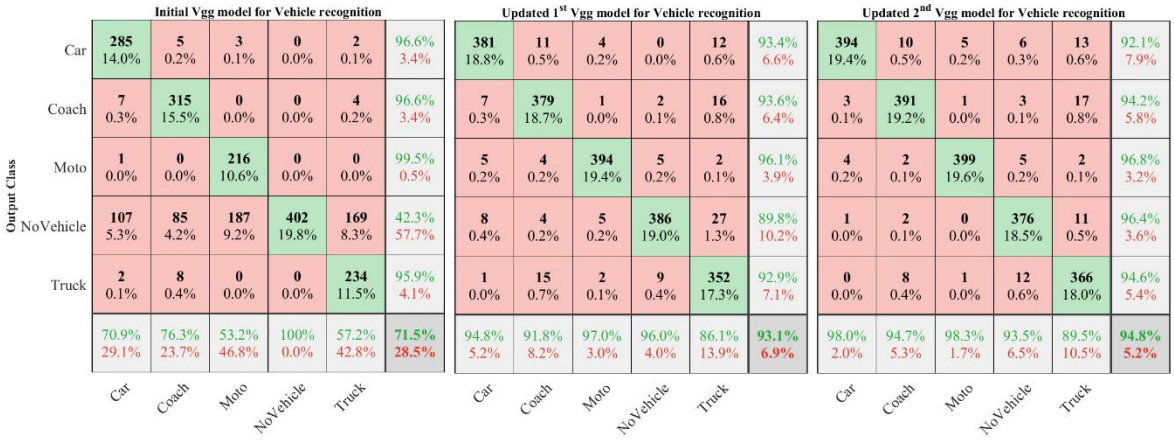


Figure 4.15 The confusion matrix of the accuracy of Vgg model for vehicle recognition

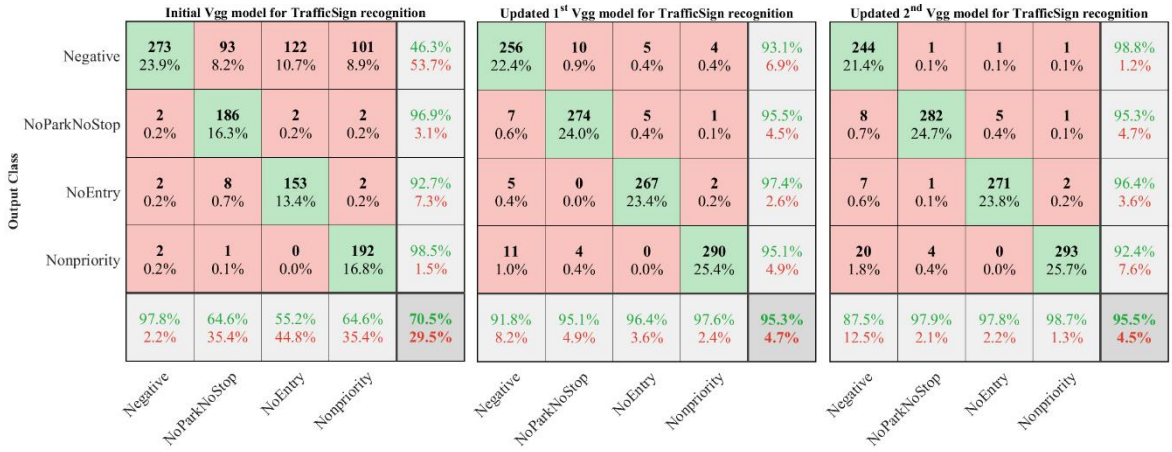
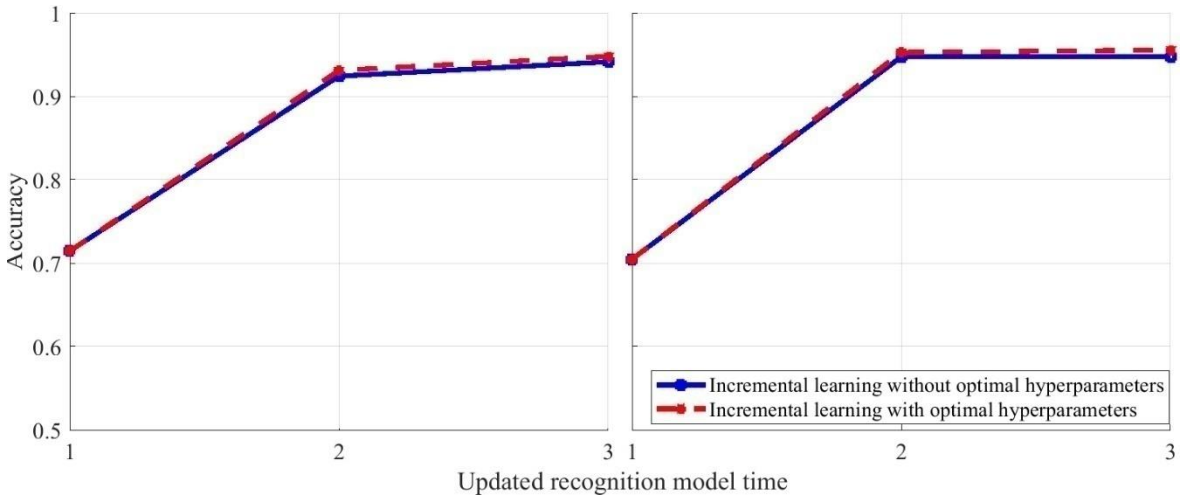


Figure 4.16 The confusion matrix of the accuracy of Vgg model for traffic sign recognition



(a) Vehicle object (b) Traffic sign object

Figure 4.17 The chart showing the increasing accuracy on recognition of Vgg model after the updated recognition model with optimal hyperparameters applied

Particularly, the application of Bayesian algorithm to search hyperparameters and model has made the accuracy on PDNet and AlexNet, Vgg models higher than those of the similar models stated in the chapter 3, when being evaluated on the same dataset. The comparison results are shown in Table 4.17.

Table 4. 1 Results of proposed methods compared to those of the Chapter 3

Models	Our method (%)	Previous method (%)
PDNet-Vehicle$_{0\ (initial\ model)}$	51.77	51.77
PDNet-Vehicle$_1$	62.30	60.58
PDNet-Vehicle$_2$	69.98	68.41
PDNet-TrafficSign$_{0\ (initial\ model)}$	70.46	70.46
PDNet-TrafficSign$_1$	85.19	84.93
PDNet-TrafficSign$_2$	92.90	90.36
AlexNet-Vehicle$_{0\ (initial\ model)}$	66.14	66.14
AlexNet-Vehicle$_1$	88.24	86.61
AlexNet-Vehicle$_2$	90.75	90.40
AlexNet-TrafficSign$_{0\ (initial\ model)}$	67.05	67.05
AlexNet-TrafficSign$_1$	88.78	87.73
AlexNet-TrafficSign$_2$	93.51	92.55
Vgg-Vehicle$_{0(initial\ model)}$	71.46	71.46
Vgg-Vehicle$_1$	93.11	92.42
Vgg-Vehicle$_2$	94.78	94.14
Vgg-TrafficSign$_{0(initial\ model)}$	70.46	70.46
Vgg-TrafficSign$_1$	95.27	94.74
Vgg-TrafficSign$_2$	95.53	94.74

4.5. Conclusion

The research content and proposal of this chapter emulated the operation of ADAS in practice. Despite the fact that testing was made on only two objects (vehicle and traffic signs), they were representative and covered all possible objects of the on-the-road journey of ADAS. Moreover, the proposed model is expected to be widely applied in all intelligent systems using object recognition complexes. The results of the proposed method have provided a number of useful contributions:

(1) It demonstrated that Adaptive Learning methods were effective, improving performance and diversifying the recognition mode of an intelligent system without relying on any human intervention. In particular, the system had the capacity to continuously learn and be ‘smart’ during its operation.

(2) It improved training and adaptive parameters on each dataset and created a rather comprehensive proposed model in terms of Adaptive Learning in intelligent systems.

(3) The proposed model matched with systems with low equipment configuration, thus lacking resources for complex or multiple object recognition. Throughout the Adaptive Learning process of the proposed model, the system was able to recognize objects with accuracy, which is equivalence and higher over time.

However, the following steps need to be taken to make the proposed solution possible and to improve recognition performance:

(1) The recognized objects should be expanded in order to diversify the capabilities of the ADAS system or to develop it into a complete robotic system capable of Adaptive Learning for all subjects.

(2) The number and value domain of the hyperparameters adapting to new datasets should be expanded before training the recognition models.

In the Chapter 4, the author mentions the two research works which is paper PP 1.5.

CONCLUSION AND DEVELOPMENT DIRECTION

1. Conclusion

The research results, which are presented in each chapter of the thesis, have been proved and confirmed through research works published in domestic and international conferences and journals. The research contents have been basically completed according to the stated objectives. In particular, outstanding contributions are:

(1) Having study and generalizing the indispensable fundamental role of traditional machine learning algorithms, the recent domestic and international researches on artificial intelligence, machine learning, Deep Learning object recognition techniques and Adaptive Learning techniques as well.

(2) The basic techniques of Deep Learning are demonstrated in the Chapter 2 (Pedestrian recognition, vehicle recognition,...). Through the simulation experiments of ADAS equipment in traffic, it has shown that the CNN models' ability to recognize is great when being trained. The research results in this chapter are considered as a foundation for an overall model development of an ADAS system which is capable of self-learning and become more intelligent.

(3) The main contribution of the thesis is to propose a comprehensive model for Adaptive Learning solution. The operation of the ADAS model demonstrated that an auto robot system is capable of self-learning and recognizing by simulation of the human brain. The proposed solution, along with adaptation and automatic updating of actual data, enables the system to change and adapt to the training hyperparameter set matched with the input data. It is this combination that has generated a quite complete model for the Adaptive Learning solution of auto robot systems in the future.

(4) Through the experiments on the research contents, the author has collected and develop a dataset of many different objects such as a data set of actual

pedestrians, a data set of pedestrian posture, a data set of traffic signs, and a dataset of vehicles as well. Because data for the experimental process are not available (including published famous datasets), the data sets of images stated in the thesis were in real ones which were collected directly from real movement of car on road or from internet videos.

(5) However, although there have been encouraged results, some following issues still remain to be solved to improve and prove the effectiveness of the Adaptive Learning model.

- A few numbers of experimental objects that have not covered many other cases. Limited images in the data set leaded to low accuracy of recognition model.

- Some parameter values for training are proposed default that have not been proved to bring the highest efficiency (For example: value of N image number at the start of retraining process of model, % of image data of the previous dataset is reused for next model training, etc.).

- The hyperparameter value range is only estimated through experiment does not value range need to be searched.

2. Development direction

The proposed model shows the Adaptive Learning solution of ADAS devices. However, it can be seen that further research and development in following different directions may be of potential:

- Extend objects for recognition to diversify the capabilities of the ADAS system or develop into a complete auto robot system capable to Adaptive Learning on all objects.

- Evaluate and search appropriate values replacing fixed values during training of Adaptive Learning model. Extend the search parameter range to increase the ability to select the appropriate parameters for retraining the model corresponding to the new data set. At the same time, the study will find a solution in

which the complexity in the hyperparameter searching process of the proposed model is reduced with minimized time and improved processing efficiency.

- In the proposed model, the continuous adaptive learning process will enable the training dataset to rapidly increase in number. Thus, the point is to develop a lean solution with a selective training dataset in order to eliminate easy samples while prioritizing hard samples. This is expected to make the model possible to reduce training time and improve the accuracy and quality of the adaptive learning process.

- Develop a complete and large data set with a variety of different types of objects for the initial training of the Adaptive Learning model.

LIST OF PUBLISHED SCIENTIFIC WORKS RELATED TO THE THESIS

1. Major publication papers

PP 1.1 D.-P. Tran, N. G. Nhu, and V.-D. Hoang, "Pedestrian action prediction based on deep features extraction of human posture and traffic scene," in *Asian Conference on Intelligent Information and Database Systems*, 2018, pp. 563-572.

PP 1.2 D.-P. Tran, V.-D. Hoang, T.-C. Pham, and C.-M. Luong, "Pedestrian activity prediction based on semantic segmentation and hybrid of machines," *Journal of Computer Science and Cybernetics*, vol. 34, pp. 113-125, 2018.

PP 1.3 D.-P. Tran and V.-D. Hoang, "Vehicle Categorical Recognition for Traffic Monitoring in Intelligent Transportation Systems," in *Asian Conference on Intelligent Information and Database Systems*, 2019, pp. 670-679.

PP 1.4 D.-P. Tran and V.-D. Hoang, "Adaptive Learning Based on Tracking and ReIdentifying Objects Using Convolutional Neural Network," *Neural Processing Letters*, vol. 50, pp. 263-282, 2019.

PP 1.5 D.-P. Tran, N. G. Nhu, and V.-D. Hoang, "Hyperparameter Optimization for improving Recognition Efficiency of an Adaptive Learning System", *IEEE Access*, vol. 08, pp.160569 - 160580, 2020.

2. Supplementary publication papers

PP 2.1 V.-D. Hoang, V.-D. Dang, T.-T. Nguyen, and D.-P. Tran, "A solution based on combination of RFID tags and facial recognition for monitoring systems," in *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, 2018, pp. 384-387.

PP 2.2 V.-H. Pham, D.-P. Tran, and V.-D. Hoang, "Personal Identification Based on Deep Learning Technique Using Facial Images for Intelligent Surveillance Systems," *International Journal of Machine Learning and Computing*, vol. 9, 2019.

PP 2.3 Tri-Cong Pham, Chi-Mai Luong, Antoine Doucet, Van-Dung Hoang, Diem-Phuc Tran, Duc-Hau Le , "Meta-analysis of computational methods for breast cancer classification," *International Journal of Intelligent Information and Database Systems*, vol. 13, 2020.

PP 2.4 V.-D. Hoang, D.-P. Tran, N. G. Nhu, and V.-H. Pham, "Deep Feature Extraction for Panoramic Image Stitching," in *Asian Conference on Intelligent Information and Database Systems*, 2020, pp. 141-151.

RESFERENCES

[1] J. Hariyono and K.-H. Jo, "Detection of pedestrian crossing road: A study on pedestrian pose recognition," *Neurocomputing,* vol. 234, pp. 144-153, 2017.

[2] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 304-311.

[3] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence,* vol. 34, pp. 743-761, 2012.

[4] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2325-2333.

[5] V.-D. Hoang, M.-H. Le, and K.-H. Jo, "End-to-end detection using multiple scale of cell based histogram of oriented gradients and adaboost learning," in *International Conference on Computational Collective Intelligence*, 2012, pp. 61-71.

[6] J. Yu, "Adaptive hidden Markov model-based online learning framework for bearing faulty detection and performance degradation monitoring," *Mechanical Systems and Signal Processing,* vol. 83, pp. 149-162, 2017.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 886-893.

[8] S. Mittal, T. Prasad, S. Saurabh, X. Fan, and H. Shin, "Pedestrian detection and tracking using deformable part models and Kalman filtering," in *SoC Design Conference (ISOCC), 2012 International*, 2012, pp. 324-327.

[9] B. Chandra and R. K. Sharma, "Deep learning with adaptive learning rate using laplacian score," *Expert Systems with Applications,* vol. 63, pp. 1-7, 2016.

[10] E. Chatzilari, S. Nikolopoulos, Y. Kompatsiaris, and J. Kittler, "Salic: Social active learning for image classification," *IEEE Transactions on Multimedia,* vol. 18, pp. 1488-1503, 2016.

[11] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-effective active learning for deep image classification," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 27, pp. 2591-2600, 2017.

[12] L. Zhang, Z. He, and Y. Liu, "Deep object recognition across domains based on adaptive extreme learning machine," *Neurocomputing,* vol. 239, pp. 194-203, 2017.

[13] P. Liu, H. Zhang, and K. B. Eom, "Active deep learning for classification of hyperspectral images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. 10, pp. 712-724, 2017.

[14] Z. Zhang, E. Pasolli, M. M. Crawford, and J. C. Tilton, "An active learning framework for hyperspectral image classification using hierarchical segmentation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. 9, pp. 640-654, 2016.

[15] V. Srivastava. (2019). *History of Artificial Intelligence – A Brief History of AI*. Available: https://www.appypie.com/history-of-artificial-intelligence

[16] L. Schultebraucks. (2017). *A Short History of Artificial Intelligence*. Available: https://dev.to/lschultebraucks/a-short-history-of-artificial-intelligence-7hm

[17] S. Russell and P. Norvig, "Artificial intelligence: a modern approach (4th Edition)," 2020.

[18] H. V. Dũng, *Giáo trình Nhận dạng và xử lý ảnh*: Nhà xuất bản Khoa học và kỹ thuật, 2018.

[19] R. C. Barros, A. C. De Carvalho, and A. A. Freitas, *Automatic design of decision-tree induction algorithms*: Springer, 2015.

[20] I. Barandiaran, "The random subspace method for constructing decision forests," *IEEE transactions on pattern analysis and machine intelligence,* vol. 20, 1998.

[21] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence,* vol. 14, p. 1612, 1999.

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning,* vol. 20, pp. 273-297, 1995.

[23] J. Weston and C. Watkins, "Multi-class support vector machines," Citeseer1998.

[24] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST),* vol. 2, p. 27, 2011.

[25] D. Shiffman, *The Nature of Code: Simulating Natural Systems with Processing*: Daniel Shiffman, 2012.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature,* vol. 521, p. 436, 2015.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov*, et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1-9.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.

[31] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91-99.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[34] Y. LeCun, "LeNet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet,* p. 20, 2015.

[35] N. Q. T. Lê Minh Hoàng, Lương Chi Mai, "Ứng dụng mô hình Markov ẩn trong nhận dạng chữ," *Tạp chí công nghệ, , số đặc biệt,* vol. tập 40, 2002.

[36] L. C. M. Dang Ngoc Duc John-Paul Hosom, "HMM/ANN System for Vietnamese Continuous Digit Recognition," *IEA/AIE: Developments in Applied Artificial Intelligence,* pp. 481-486, 2003.

[37] N. Q. T. a. L. C. M. P. A. Phuong, "An Efficient Model for Isolated Vietnamese Handwritten Recognition," *2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing,* pp. 358-361, 2008.

[38] N. Q. T. Phạm Anh Phương, Lương Chi Mai, "Kết hợp các bộ phận phân lớp SVM cho việc nhận dạng chữ Việt viết tay rời rạc," *Tạp chi Tin học và Điều khiển học,* vol. tập 25, 2009.

[39] L. T. H. Ngô Quốc Tạo, Nguyễn Thị Ngọc Hân, "Xác định mặt người dựa trên mạng nơron," *Kỷ yếu hội thảo quốc gia, Đà Nẵng,* 2004.

[40] T. V. L. Lam Thanh Hien , Ha Manh Toan , Do Nang Toan, "Modeling the Human Face and its Application for Detection of Driver Drowsiness," *International Journal of Computer Science and Telecommunications.,* vol. 3, 2012.

[41] N. Q. T. I. t. Lê Thanh Hà, Hải Phòng, "Một số kết quả ứng dụng svms cho nhận dạng mặt người," *Kỷ yếu Hội thảo quốc gia,* 2006.

[42] Đ. N. Toàn, "Nghiên cứu, phát triển hệ thống mô phỏng các bộ phận chính của cơ thể con người phục vụ cho việc giảng dạy và tra cứu," *Đề tài nghiên cứu tại Trường Đại học Y dược Thái Nguyên,* 2011.

[43] T. Y. Pham Ngoc Hung, "Adaptive Learning of Hand Movement in Human Demonstration for Robot Action," *Journal of Robotics and Mechatronics,* vol. 29, 2017.

[44] N. H. Quang, "The Impact of Each Deep Neural Network Layer on the Performance of End-To-End Vietnamese Speech Recognition," *JP Journal of Heat and Mass Transfer,* vol. Special Volume, 2018.

[45] C.-M. L. Tri-Cong Pham, Muriel Visani, Van-Dung Hoang, "Deep CNN and data augmentation for skin lesion classification," *Asian Conference on Intelligent Information and Database Systems,* pp. 573-582, 2018.

[46] M.-H. L. Van-Dung Hoang, Truc Thanh Tran, Van-Huy Pham, "Improving Traffic Signs Recognition Based Region Proposal and Deep Neural Networks," *Asian Conference on Intelligent Information and Database Systems,* pp. 604-613, 2018.

[47] V. D. VD Hoang, TT Nguyen, DP Tran, "A solution based on combination of RFID tags and facial recognition for monitoring systems " *NAFOSTED Conference on Information and Computer Science (NICS),* pp. 384-387, 2018.

[48] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng*, et al.*, "A survey of deep learning-based object detection," *IEEE Access,* vol. 7, pp. 128837-128868, 2019.

[49] X. Jiang, A. Hadid, Y. Pang, E. Granger, and X. Feng, *Deep Learning in object detection and recognition*: Springer, 2019.

[50] K. Chowdhary, *Fundamentals of Artificial Intelligence*: Springer, 2020.

[51] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems,* vol. 30, pp. 3212-3232, 2019.

[52] X. Wu, D. Sahoo, and S. C. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing,* vol. 396, pp. 39-64, 2020.

[53] I. S. Alex Krizhevsky, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems 25 (NIPS 2012),* vol. 25, pp. 1106-1114, 2012.

[54] A. Paul, R. Chauhan, R. Srivastava, and M. Baruah, "Advanced driver assistance systems," SAE Technical Paper 0148-7191, 2016.

[55] F. X. Shubham Mittal, Suraj Saurabh, Twisha Prasad, Hyunchul Shin, "Pedestrian Detection and Tracking Using Deformable Part Models and Kalman Filtering," *Journal of Computer-Mediated Communication,* vol. 10, pp. 960-966, 2013.

[56] A. A. Yu Xiang, Silvio Savarese, "Learning to Track: Online Multi-object Tracking by Decision Making," *Computer Vision (ICCV), 2015 IEEE International Conference on,* pp. 4705-4713, 2015.

[57] B. T. Navneet Dalal, "Histograms of Oriented Gradients for Human Detection," *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05),* vol. 1, pp. 886-893, 2005.

[58] M.-H. L. Van-Dung Hoang, Kang-Hyun Jo, "Robust Human Detection Using Multiple Scale of Cell Based Histogram of Oriented Gradients and AdaBoost Learning," *Computational Collective Intelligence. Technologies and Applications,* vol. 7653, pp. 61-71, 2012.

[59] R. G. Pedro F Felzenszwalb, David McAllester, Deva Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 32, pp. 1627-1645, 2010.

[60] R. J. Mstafa and K. M. Elleithy, "A video steganography algorithm based on Kanade-Lucas-Tomasi tracking algorithm and error correcting codes," *Multimedia Tools and Applications,* vol. 75, pp. 10311-10333, 2016.

[61] V.-D. Hoang, "Multiple classifier-based spatiotemporal features for living activity prediction," *Journal of Information and Telecommunication,* vol. 1, pp. 100-112, 2017.

[62] K.-H. J. Joko Hariyono, "Detection of Pedestrian Crossing Road A Study on Pedestrian Pose Recognition," *Neurocomputing,* vol. 234, pp. 144-153, 2016.

[63] M. A. Russell Stewart, Andrew Y. Ng, "End-to-end people detection in crowded scenes," *2016 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 2325-2333, 2015.

[64] A. L. a. M. V. D. S. Piérard, "A probabilistic pixel-based approach to detect humans in video streams," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* pp. 921-924, 2011.

[65] C. R. Dow, H. H. Ngo, L. H. Lee, P. Y. Lai, K. C. Wang, and V. T. Bui, "A crosswalk pedestrian recognition system by using deep learning and zebra-crossing recognition techniques," *Software: Practice and Experience,* vol. 50, pp. 630-644, 2020.

[66] I. Amirullah, R. Yusliana Bakti, I. Areni, and A. A. Alimuddin, *Vehicle detection and tracking using Gaussian Mixture Model and Kalman Filter*, 2016.

[67] Y. Chen and Q. Wu, *Moving vehicle detection based on optical flow estimation of edge*, 2015.

[68] J.-y. Choi, K.-S. Sung, and Y. Yang, *Multiple Vehicles Detection and Tracking based on Scale-Invariant Feature Transform*, 2007.

[69] G. Yan, Y. Ming, Y. Yu, and L. Fan, *Real-time vehicle detection using histograms of oriented gradients and AdaBoost classification* vol. 127, 2016.

[70] S. G. d. S. Filho, R. Z. Freire, and L. d. S. Coelho, "Feature Extraction for On-Road Vehicle Detection Based on Support Vector Machine," *Conference Proceedings,* 2017.

[71] Z. Moutakki, M. I. Ouloul, A. Karim, and A. Abdellah, *Real-Time System Based on Feature Extraction for Vehicle Detection and Classification* vol. 19, 2018.

[72] A. A. Yilmaz, M. S. Guzel, I. Askerbeyli, and E. Bostanci, "A vehicle detection approach using deep learning methodologies," *arXiv preprint arXiv:1804.00429,* 2018.

[73] J. Espinosa Oviedo, S. Velastin, and J. W. Branch, *Vehicle Detection Using Alex Net and Faster R-CNN Deep Learning Models: A Comparative Study*, 2017.

[74] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, *Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks* vol. 11, 2014.

[75] S. Qu, Y. Wang, G. Meng, and C. Pan, *Vehicle Detection in Satellite Images by Incorporating Objectness and Convolutional Neural Network*, 2016.

[76] Y. Koga, H. Miyazaki, and R. Shibasaki, "Counting vehicles by deep neural network in high resolution satellite images."

[77] C. Migel Bautista, C. Austin Dy, M. Inigo Manalac, R. Angelo Orbe, and M. Cordel, II, *Convolutional neural network for vehicle detection in low resolution traffic videos*, 2016.

[78] M. S. I. Harbas, "Detection of roadside vegetation using features from the visible spectrum," *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO),* pp. 1204-1209, 26-30 May 2014 2014.

[79] M. Aly, "Real time Detection of Lane Markers in Urban Streets," *IEEE Intelligent Vehicles Symposium,* p. 6, 2014.

[80] M. M. T. R. K. Satzoda, "Vision-Based Lane Analysis: Exploration of Issues and Approaches for Embedded Realization," *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops,* pp. 604-609, 23-28 June 2013 2013.

[81] Q. W. Wang Hua, Wang Y, R Miller Gregory, "Dual Roadside Seismic Sensor for Moving Road Vehicle Detection and Characterization," *Sensors,* vol. 14, pp. 2892-910, 2014.

[82] J. Z. Q. Wang, H. Xu, B. Xu, R. Chen, "Roadside Magnetic Sensor System for Vehicle Detection in Urban Environments," *IEEE Transactions on Intelligent Transportation Systems,* vol. 19, pp. 1365-1374, 2018.

[83] F. J. J. Brostow Gabriel, Cipolla Roberto, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters,* vol. 30, pp. 88-97, 2009.

[84] A. K. V. Badrinarayanan, R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, pp. 2481-2495, 2017.

[85] X. Li, M. Ye, Y. Liu, and C. Zhu, "Adaptive deep convolutional neural networks for scene-specific object detection," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 29, pp. 2538-2551, 2017.

[86] T.-K. Lin, "Adaptive learning method for multiple-object detection in manufacturing," *Advances in Mechanical Engineering,* vol. 7, p. 1687814015618906, 2015.

[87] L. Cheng, X. Liu, L. Li, L. Jiao, and X. Tang, "Deep Adaptive Proposal Network for Object Detection in Optical Remote Sensing Images," *arXiv preprint arXiv:1807.07327,* 2018.

[88] K. Blix and T. Eltoft, "Machine learning automatic model selection algorithm for oceanic chlorophyll-a content retrieval," *Remote Sensing,* vol. 10, p. 775, 2018.

[89] S. Raschka, "Model evaluation, model selection, and algorithm selection in machine learning," *arXiv preprint arXiv:1811.12808,* 2018.

[90] L. Li and A. Talwalkar, "Random search and reproducibility for neural architecture search," *arXiv preprint arXiv:1902.07638,* 2019.

[91] H. Bertrand, R. Ardon, M. Perrot, and I. Bloch, "Hyperparameter optimization of deep neural networks: Combining hyperband with Bayesian model selection," in *Conférence sur l'Apprentissage Automatique*, 2017.

[92] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[93] S. Kamada and T. Ichimura, "An object detection by using adaptive structural learning of deep belief network," in *2019 international joint conference on neural networks (IJCNN)*, 2019, pp. 1-8.

[94] C. Huang, S. Lucey, and D. Ramanan, "Learning policies for adaptive tracking with deep feature cascades," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 105-114.

[95] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE transactions on pattern analysis and machine intelligence,* vol. 41, pp. 3071-3085, 2018.

[96] N. Q. K. Le, T.-T. Huynh, E. K. Y. Yapp, and H.-Y. Yeh, "Identification of clathrin proteins by incorporating hyperparameter optimization in deep learning and PSSM profiles," *Computer methods and programs in biomedicine,* vol. 177, pp. 81-88, 2019.

[97] A. Klein, S. Falkner, S. Bartels, P. Hennig, and F. Hutter, "Fast bayesian optimization of machine learning hyperparameters on large datasets," *arXiv preprint arXiv:1605.07079,* 2016.

[98] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram*, et al.*, "Scalable bayesian optimization using deep neural networks," in *International conference on machine learning*, 2015, pp. 2171-2180.

[99] M. Claesen and B. De Moor, "Hyperparameter search in machine learning," *arXiv preprint arXiv:1502.02127,* 2015.

[100] S. C. Smithson, G. Yang, W. J. Gross, and B. H. Meyer, "Neural networks designing neural networks: multi-objective hyper-parameter optimization," in *Proceedings of the 35th International Conference on Computer-Aided Design*, 2016, pp. 1-8.

[101] E. Bochinski, T. Senst, and T. Sikora, "Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3924-3928.

[102] W.-Y. Lee, K.-E. Ko, Z.-W. Geem, and K.-B. Sim, "Method that determining the Hyperparameter of CNN using HS algorithm," *Journal of Korean institute of intelligent systems,* vol. 27, pp. 22-28, 2017.

[103] A.-C. Florea and R. Andonie, "Weighted random search for hyperparameter optimization," *arXiv preprint arXiv:2004.01628,* 2020.

[104] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA," *The Journal of Machine Learning Research,* vol. 18, pp. 826-830, 2017.

[105] X. Zeng and G. Luo, "Progressive sampling-based Bayesian optimization for efficient and automatic machine learning model selection," *Health information science and systems,* vol. 5, p. 2, 2017.

[106] G. Dikov, P. van der Smagt, and J. Bayer, "Bayesian learning of neural network architectures," *arXiv preprint arXiv:1901.04436,* 2019.

[107] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE transactions on pattern analysis and machine intelligence,* vol. 36, pp. 1532-1545, 2014.

[108] K. He, X. Zhang, S. Ren, and J. Sun, *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification* vol. 1502, 2015.

[109] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599,* 2010.

[110] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE,* vol. 104, pp. 148-175, 2015.

[111] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *25th annual conference on neural information processing systems (NIPS 2011)*, 2011.