DESIGN AND DEVELOPMENT OF EMBEDDED VISION BASED OBJECT DETECTORS FOR AUTONOMOUS DRIVING APPLICATION

Submitted in partial fulfilment of the requirements
for the award of the degree of

Doctor of Philosophy

by

Chintakindi Balaram Murthy
(Roll No. 718143)

Supervisor
Dr. Mohammad Farukh Hashmi
Assistant Professor, Dept. of ECE

Department of Electronics & Communication Engineering
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL – 506004, T.S, INDIA
June-2022

APPROVAL SHEET

This thesis entitled "**Design and Development of Embedded Vision Based Object Detectors for Autonomous Driving Application**" by Mr. Ch. Balaram Murthy is approved for the degree of **Doctor of Philosophy**.

Examiners

Supervisor

Dr. Mohammad Farukh Hashmi
Assistant Professor, Electronics and Communication Engineering Department,
NIT WARANGAL

Chairman

Prof. P. Sreehari Rao
Professor, Head,
Electronics and Communication Engineering Department,
NIT WARANGAL

Date: 26/06/2023

DECLARATION

This is to certify that the work presented in the thesis entitled "**Design and Development of Embedded Vision Based Object Detectors for Autonomous Driving Application**" is a bonafide work done by me under the supervision of **Dr. Mohammad Farukh Hashmi**, Department of Electronics and Communication Engineering, National Institute of Technology Warangal, and was not submitted elsewhere for the award of any degree.

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/date/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Ch. Balaram Murthy

Roll No: 718143

Date: 26/06/2023

Place: WARANGAL

Department of Electronics and Communication Engineering
National Institute of Technology
Warangal – 506 004, Telangana, India

CERTIFICATE

This is to certify that the dissertation work entitled **"Design and Development of Embedded Vision Based Object Detectors for Autonomous Driving Application"**, which is being submitted by Mr. Ch. Balaram Murthy (Roll No.718143), is a bonafide work submitted to National Institute of Technology Warangal in partial fulfilment of the requirement for the award of the degree of *Doctor of Philosophy in Electronics and Communication Engineering.*

To the best of our knowledge, the work incorporated in this thesis has not been submitted elsewhere for the award of any degree.

Dr. Mohammad Farukh Hashmi
(Research Supervisor)
Assistant Professor
Department of ECE
National Institute of Technology
Warangal – 506004

Dedicated to My

Family & Beloved Teachers

Contents

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my supervisor **Dr. Mohammad Farukh Hashmi** for continuous support of my research, for his patience, motivation, and guidance. My sincere thanks to him for providing me with an opportunity to join the institute as a research scholar and giving me access to research facilities. I find words inadequate to thank him for enabling me to complete this work in spite of all obstacles. The thesis would not have seen the light of day without his continued support and cooperation. Sir, I think you deserve an extra note of thanks for your ongoing leadership and support. Finally, I just wanted to thank you for being such a great advisor in my personal life too. I have enjoyed working with you so much over the past few years.

My special word of thanks should also go to **Prof. P. Sreehari Rao**, Associate Professor, Head of the department, Electronics and Communication Engineering Department, NIT Warangal for his valuable suggestions and support that he offered during my research.

Here, I also would like to take this privilege to thank my DSC members, Dr.P. Muralidhar, Associate Professor, Department of Electronics and Communication Engineering, NIT Warangal, Dr. B.K.N.Srinivasarao, Assistant Professor, Department of Electronics and Communication Engineering, NIT Warangal and Dr.M.Srinivas, Assistant Professor, Department of Computer Science Engineering, NIT Warangal for their continuous support, suggestions and advice during my research period whenever required.

I am grateful to the former Heads of the ECE department Prof. L.Anjaneyelu and Prof. N. Bheema Rao for their continuous support and encouragement. I would also appreciate the encouragement from teaching, non-teaching members and fraternity of Dept. of E.C.E. of N.I.T. Warangal. They have always been encouraging and supportive. I am also thankful to Dr. G. Arun Kumar and Dr. M. Raja Vishwanathan provided much help in proof reading my research papers.

It is my pleasure to show my indebtedness to my co-scholars at NITW Mr. B. Tulsai Naik, Miss. S. Priyadarsini, Dr. Prathap Soma, Mr. M. Vijay Kumar, Mrs. V. Kiranmai Mr. K. Sagar, Mr.Ravi Shankar and friends for their help during the course of this work. Their timely help and friendship shall always be remembered during my stay.

I owe my deepest gratitude towards my better half (Ragini Chintakindi) for her continuous support, love and understanding of my goals and aspirations. Her patience and sacrifice will remain my inspiration throughout my life. Without her help, I would not have been able to complete much of what I have done and become who I am. Also, I love my daughter (Hamsika Chintakindi) for giving me happiness and support during research. I appreciate my daughter for the patience she showed during my research. I consider myself the luckiest in the world to have such a lovely and caring family, standing beside me with their love and unconditional support.

My heartfelt thanks to my family members (parents Sathyamma-Thirumalaiah (Late), brother's family (Santhosh - Gouthami) for supporting me spiritually throughout my life in general. Their unconditional love, support, self-sacrifices and patience has always been the source of my strength. I shall be grateful forever because of their kindness and love.

Finally, I thank God, for filling me every day with new hopes, strength, purpose and faith.

Ch. Balaram Murthy

List of Publications

International Journals

1. Murthy, Chintakindi Balaram, Mohammad Farukh Hashmi, Neeraj Dhanraj Bokde, and Zong Woo Geem. "Investigations of object detection in images/videos using various deep learning techniques and embedded platforms - A comprehensive review." Applied sciences 10, no. 9 (2020): 3280**.** https://doi.org/10.3390/app10093280. **I.F- 2.838. (SCI).**

2. Murthy, Chintakindi Balaram, Mohammad Farukh Hashmi, Ghulam Muhammad, and Salman A. AlQahtani. "YOLOv2PD: An Efficient Pedestrian Detection Algorithm Using Improved YOLOv2 Model." CMC-COMPUTERS MATERIALS & CONTINUA 69, no. 3 (2021): 3015-3031. DOI:10.32604/cmc.2021.018781. **I.F- 3.772. (SCI).**

3. Chintakindi, Balaram Murthy, and Mohammad Farukh Hashmi. "SSAD: Single-Shot Multi-Scale Attentive Detector for Autonomous Driving." IETE Technical Review (2023): 1-13. **I.F-1.932. (SCI).**

4. Murthy, Chintakindi Balaram, Mohammad Farukh Hashmi, and Avinash G. Keskar. "Optimized MobileNet+SSD: a real-time pedestrian detection on a low-end edge device." International Journal of Multimedia Information Retrieval 10, no. 3 (2021): 171-184. https://doi.org/10.1007/s13735-021-00212-7. **I.F- 3.205. (SCI, DBLP).**

5. Murthy, Chintakindi Balaram, Mohammad Farukh Hashmi, and Avinash G. Keskar. "EfficientLiteDet: a real-time pedestrian and vehicle detection algorithm." Machine Vision and Applications 33, no. 3 (2022): 1-15. **I.F- 2.983. (SCI, DBLP).**

International Conferences

1. C. B. Murthy and M. F. Hashmi, "Real Time Pedestrian Detection Using Robust Enhanced YOLOv3+," 2020 21st International Arab Conference on Information Technology (ACIT), 2020, pp. 1-5, doi: 10.1109/ACIT50332.2020.9300053. **(Scopus Indexed, DBLP).**

2. C. B. Murthy and M. Farukh Hashmi, "Real Time Pedestrian Detection Using Robust Enhanced Tiny-YOLOv3," 2020 IEEE 17th India Council International Conference (INDICON), 2020, pp. 1-5, doi: 10.1109/INDICON49873.2020.9342082**. (Scopus Indexed).**

3. Chintakindi, Balaram Murthy, and Farukh Hashmi Mohammad. "A Lightweight Network for Detecting Pedestrians in Hazy Weather." In Machine Vision and Augmented Intelligence: Select Proceedings of MAI 2022, pp. 37-44. Singapore: Springer Nature Singapore, 2023. **(Scopus Indexed).**

ABSTRACT

Object detection is a crucial and challenging task in the field of autonomous or semi-autonomous driving systems, embedded vision, smart transportation and robotics. Over the last decade, many researchers have proposed different object detection algorithms. Most of the research is devoted to improve either detection accuracy or inference speed, as these two-evaluation metrics play a crucial role while evaluating the performance of object detection algorithm. However deep neural networks are computationally expensive and are difficult to deploy on low-end edge devices because of low memory footprint storage space, power hungry devices and limited computational power. Therefore, in this dissertation, a series of object detection algorithms have been presented, to develop a faster, light weight and more efficient object detector which can detect multi-scale objects, smaller and occluded targets accurately without losing any detection accuracy and be able to run on any low-end edge device in near real-time.

An important application area of object detection is pedestrian detection and vehicle detection. They are used extensively in complex applications, including video surveillance, self-driving cars, etc. In this dissertation, a series of works on object detection (especially detecting pedestrians and vehicles) is presented, starting from heavy YOLOv2PD network and moves towards light-weight EfficientLiteDet network. The main work of this dissertation was divided into two parts: algorithms employing traditional approach and efficient approach. In the first part traditional approach was employed for designing object detection algorithms.

In this work, YOLOv2 ("YOU ONLY LOOK ONCE Version 2")-based pedestrian detection (referred to as YOLOv2PD) algorithm was developed which is more suitable for detecting smaller and densely distributed pedestrians. This architecture adopts a Multi-layer Feature Fusion (MLFF) strategy, which helps to improve the model's feature extraction ability and, when one repeated convolution layer is removed from the final layer, computational complexity was reduced. Both the network structure and loss function were improved to make the model more accurate and faster. A novel Single shot Multi-scale detection network with feature fusion and multi-scale attention mechanism was adopted in this work in order to achieve optimal trade-off balance between detection accuracy and speed while detecting small-scale and occluded objects for autonomous driving. This network, is referred to as Single shot Multi-scale Attentive Detector, in short (SSAD) and it would build feature relations of the feature map in spatial space. The designed model learns to highlight pedestrian and vehicle regions on the extracted feature map and suppress irrelevant regions, from the global relation information and thus provide reliable guidance for autonomous driving. SSAD design is very simple, highly accurate and computationally efficient.

In the second part, efficient approach was employed for designing object detection algorithms. In this part, our goal was to design a light weight object detector which can run in near real-time on any low-end edge device. To achieve real-time pedestrian detection without any loss in detection accuracy, an Optimized MobileNet+SSD network was developed. This network lets the components work in coordination in such a manner that their strengths are improved and the number of parameters is decreased compared to recent detection architectures. A concatenation feature fusion module is introduced for adding contextual information in our network to improve detection accuracy of pedestrians. The designed network when deployed on low-end edge device Jetson Nano runs with 34.01 FPS. To improve detection accuracy and speed while detecting occluded, denser and tiny objects a novel light-weight detector was proposed and which is referred to as EfficientLiteDet. Based on Tiny-YOLOv4, one more prediction head was introduced in this network to detect multi-scale targets effectively. In order to detect tiny and occluded denser targets, Transformer Prediction Heads (TPH) were deployed by replacing original YOLOv4 anchor detection heads. To explore the potential of self-attention mechanism in TPH, this model integrates “convolutional block attention model (CBAM)” to locate attention region on scenarios with denser targets. Besides efficiency, our qualitative and quantitative analysis show that EfficientLiteDet network is more efficient, faster and light-weight and it can be deployed on any low-end edge device to achieve real-time detection performance.

List of Figures

List of Tables

Nomenclature

AP	Average Precision
ADAS	Advanced Driver Assistance System
ADAM	Adaptive Moment Estimation
ADS	Assisted Driving System
BB	Bounding Box
BN	Batch Normalization
BFLOPS	Billion Floating Point Operations per Second
CNN	Convolutional Neural Network
CV	Computer Vision
CBAM	Convolutional Block Attention Module
CUDA	Compute Unified Device Architecture
CSPDarkNet	Cross-stage Partial DarkNet
CAWS	Collision Avoidance Warning System
DPM	Deformable Part Model
DCNN	Deep Convolutional Neural Network
DSC	Depthwise Separable Convolution
DW	Depth-wise
EfficientLiteDet	Efficient Light-weight detector
FC	Fully Connected
FP	False Positive
FN	False Negative
FM	Feature Map
FPN	Feature Pyramid Network
FPS	Frames per second
FLOPS	Floating Point Operations per Second
GFLOPS	Giga Floating Point Operations per Second
GPU	Graphics Processing Unit
Fast RCNN	Fast Region based Convolutional Neural Network
Faster RCNN	Faster Region based Convolutional Neural Network
FSSD	Feature-fused Single-shot Detector

HOG	Histogram of Oriented Gradients
HV	Human Vision
IoU	Intersection over Union
ITS	Intelligent Transportation System
LAMR	Log Average Miss Rate
LBP	Local Binary Patterns
LSTM	Long Short-Term Memory
LPDDR4	Lower Power Double Data rate RAM4
MAU	Multi-scale Attention Unit
MR	Miss-rate
MLFF	Multi-layer Feature Fusion
NLP	Natural Language Processing
PASCAL VOC	Pascal Visual Object Classes
PM	Prediction Module
PR	Precision vs. Recall
PW	Point wise
PANet	Path Aggregation Network
ReLU6	Rectified Linear Unit 6
ResNet	Residual Neural Network
RNN	Recurrent Neural Network
RCNN	Region based Convolutional Neural Network
MLP	Multi-layer Perception
MLFF	Multi-layer Feature Fusion
Mask-RCNN	Mask-Region based Convolutional Neural Network
NMS	Non-max Suppression
Soft-NMS	Soft Non-max Suppression
SPPNet	Spatial Pyramid Pooling Network
SAM	Self-attention Mechanism
SGD	Stochastic Gradient Descent
SSD	Single Shot Multi-Box Detector
SSAD	Single Shot Multi-scale Attentive Detector
SIFT	Scale-Invariant Feature Transform
SOTA	State-of-the-Art

TP	True Positive
TE	Transformer Encoder
TPH	Transformer Prediction Head
TN	True Negative
VA	Visual Attention
VJ	Voila Jones
WBF	Weighted Box Fusion
YOLO	You Only Look Once
YOLOv2	You Only Look Once Version 2
YOLOv2PD	You Only Look Once Version 2 based Pedestrian Detection
YOLOv3	You Only Look Once Version 3
YOLOv4	You Only Look Once Version 4

Chapter 1

Introduction

1.1 Introduction

Recently, computer vision (CV) has been extensively researched in the fields of object detection for industrial automation, consumer electronics, medical imaging, military, and video surveillance [1]. It is predicted that the computer vision market will be worth $50 billion by the end of 2020. Since the beginning, the objective of computer vision systems has been automatic processing, analysis and interpretation of images, using classical algorithms including: cascade Speeded up robust features (SURF) [2], Haar like features [3], Scale-Invariant Feature Transform (SIFT) [4], shape contexts [5], Deformable Part-Based Model (DPM) [6], Histogram of Gradients (HOG) [7] and Local Binary Patterns (LBP) [8]. In 2012, significant advances were made in image processing methods, one of which was the use of deep learning techniques. This has led to further research and applications, the results of which have shown progress in the majority of computer vision challenges. For example, there are many research activities that are conducted in the area of computer vision, and one of the main aims is to determine whether or not there are any instances of objects from the specified varieties (e.g., animals, vehicles, and pedestrians) in an image, and if present, return the spatial location and extent of a single object (by bounding box). In contrast to significant progress in object detection focusing on still images, video object detection has received scant attention.

Murthy et al. [9] stated that the goals of object detection are to achieve both high accuracy and high efficiency by developing robust object detection algorithms. They are discussed in the form of tree diagram shown in Figure 1.1. Two of the main sub-branches of Object detection are pedestrian detection and vehicle detection areas, which deals with detecting specific pedestrian and vehicle classes.

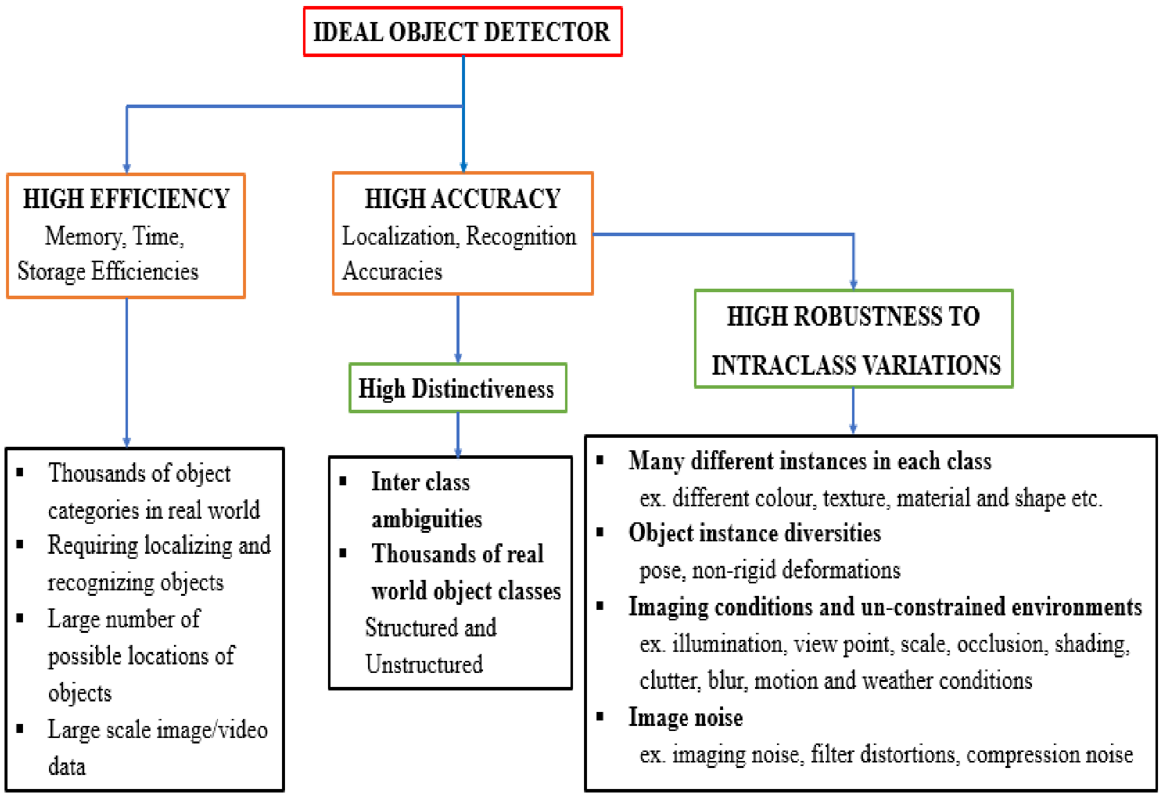
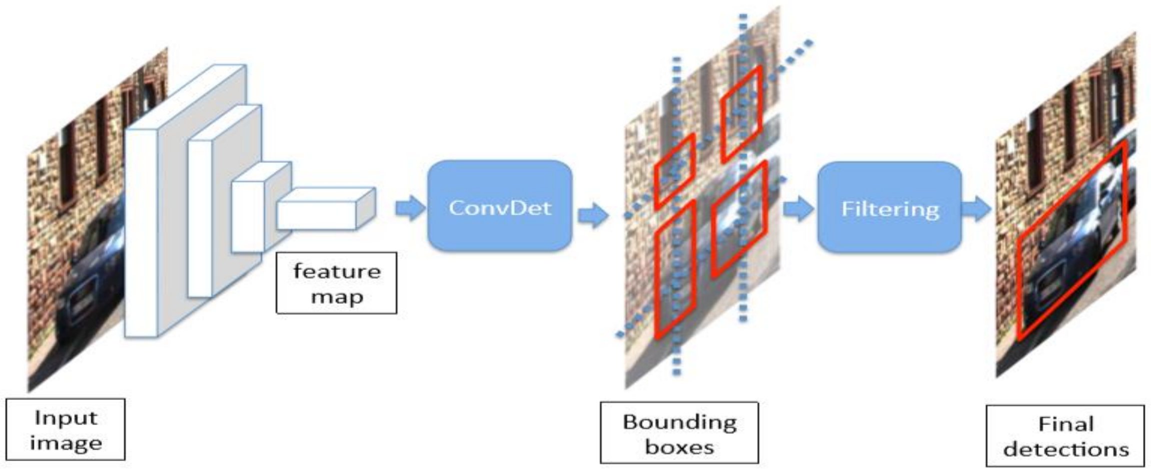

IDEAL OBJECT DETECTOR
HIGH EFFICIENCY
Memory, Time, Storage Efficiencies
HIGH ACCURACY
Localization, Recognition Accuracies
HIGH ROBUSTNESS TO INTRACLASS VARIATIONS
High Distinctiveness
Thousands of object categories in real world
Requiring localizing and recognizing objects
Large number of possible locations of objects
Large scale image/video data
Inter class ambiguities
Thousands of real world object classes
Structured and Unstructured
Many different instances in each class
ex. different colour, texture, material and shape etc.
Object instance diversities
pose, non-rigid deformations
Imaging conditions and un-constrained environments
ex. illumination, view point, scale, occlusion, shading, clutter, blur, motion and weather conditions
Image noise
ex. imaging noise, filter distortions, compression noise

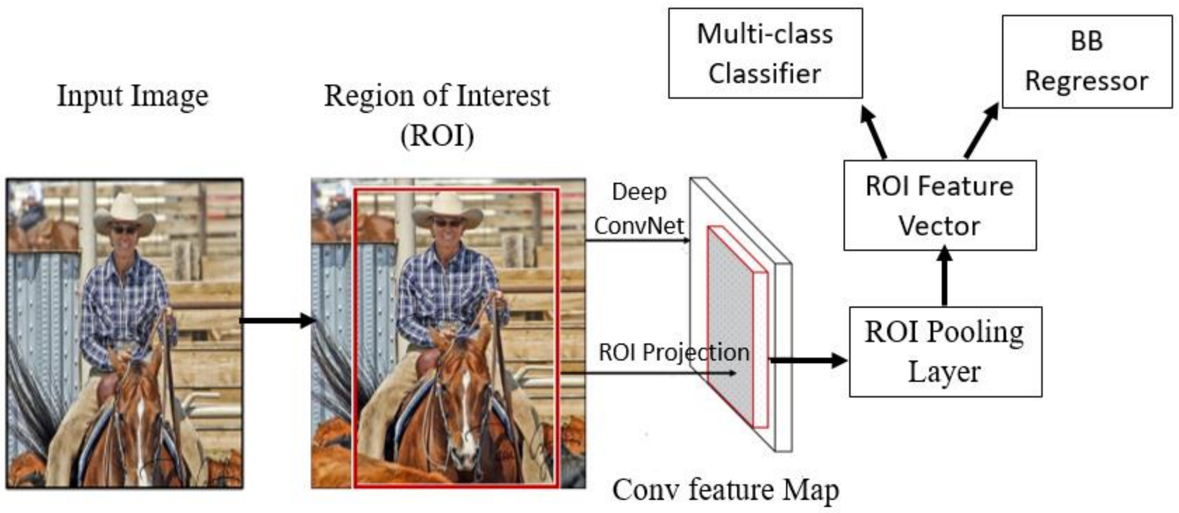
Figure 1.1 Goals of ideal Object Detector.

The embedded vision system of autonomous vehicle technology was initially very difficult to develop in the field of CV; however, owing to continuous improvements of hardware computational power, many researchers have attempted to develop reliable vision systems for autonomous driving cars. Compared to generic object detection, pedestrian detection and vehicle detection have their own differentiated characteristics, and from a practical and landing space perspectives. For example, driverless car systems, intelligent robots, intelligent video surveillance and intelligent transportation system (ITS) use both pedestrian and vehicle detection systems. Classical machine learning based methods employed for pedestrian and vehicle detection provide faster detection speed. However, such classical methods could only achieve good detection results under specific conditions, although they lose detection accuracy greatly in the presence of poor lighting conditions or occlusion situations and fail to satisfy real-time requirements.

Pedestrian and vehicle detection methods based on convolutional neural networks (CNN’s) have achieved significant improvements both in terms of speed and accuracy and have become current mainstream methods. Deep convolutional neural network-based detection methods are basically classified into two types. They are region proposal based frameworks i.e. two-stage detector such as Region based Convolutional Neural Network (RCNN) series [10-13] and Regression/Classification i.e. one-stage detector such as You Only Look Once (YOLO) series [14-18], Single-shot Multi-box Detector (SSD) series [19-21]. Figure 1.2 shows the block diagram of general object detection algorithm of one stage and two-stage detectors. Despite two-stage detectors achieving higher detection accuracy, they run slower and are not suitable to deploy on low-end edge devices. One-stage detectors lose some part of detection accuracy but run at a higher speed. Therefore, we can achieve a better trade-off balance between detection accuracy and speed using one-stage detectors when deployed on edge devices for autonomous driving application.



(a) One-stage Object Detector.



(b) Two-stage Object Detector.

Figure 1.2 Block diagram of general Object Detection.

1.2 Motivation & Problem Statement

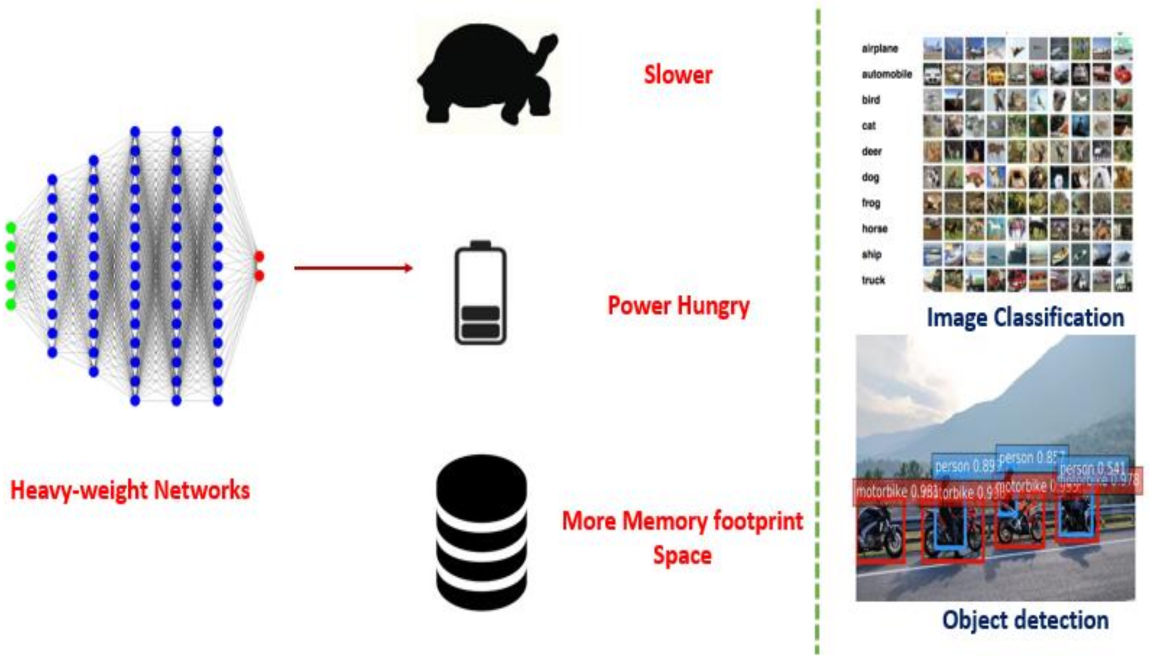
1.2.1 Motivation

Recently, in the field of automobile industry, there has been a great increase in technological innovation though the number of accidents has grown greatly due to various factors. So, to resolve this challenge, all automobile companies have been moving towards developing advanced unmanned or driver-assistance driving systems and they include many vision sensors and deploy complex algorithms. Since safety is top priority and plays a crucial role in vision-based systems, there is a need to develop faster, accurate and better real-time object detection algorithm. Therefore, great efforts have been devoted to improve both detection accuracy and speed in order to achieve real-time detection performance. Many applications of autonomous driving run on edge devices and demand online processing of data with low latency. Light-weight models have fewer operations; allowing them to process input with low latency. Also, these networks are small, so they can easily fit onto on-chip memory and help to reduce read/write latency. This further improves the inference speed.

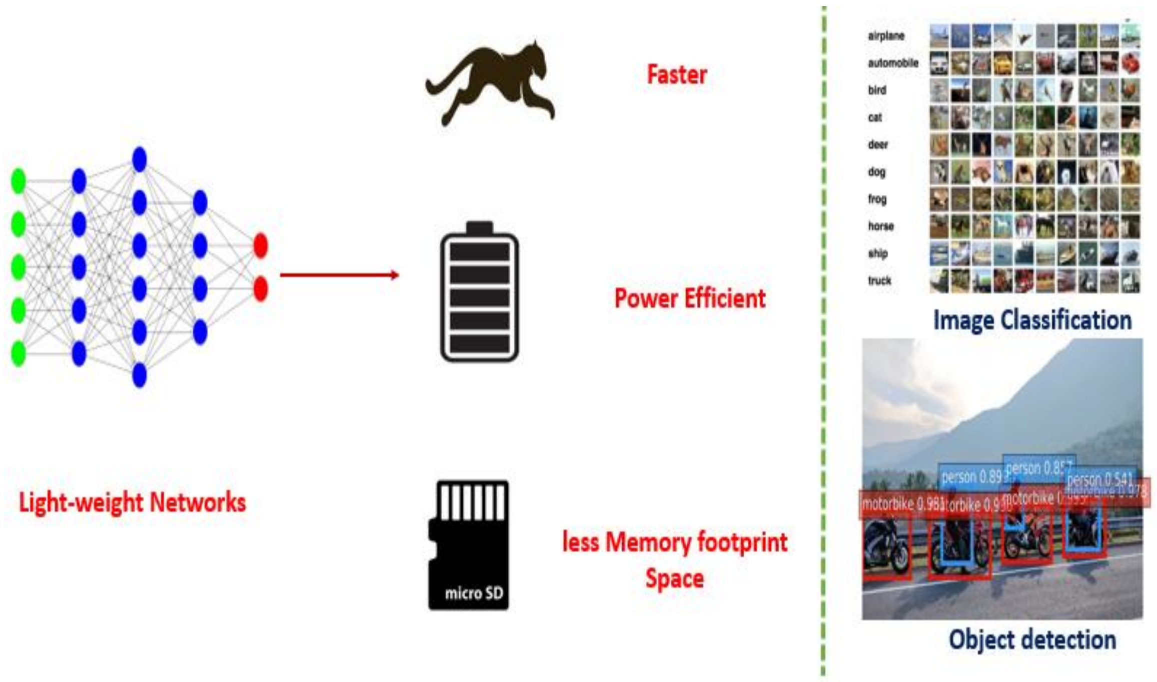
However large-scale Deep CNN based algorithms require high computational and large memory footprint resources and need a huge number of fine-tuning parameters. To realize real-time object detection, it requires powerful Graphics Processing Unit (GPU) computing power. However, it is difficult to deploy these heavy models on any low-end edge device due to limited computational power and memory constraints. Therefore, there is a need to develop a faster, efficient and light-weight object detection algorithms to operate on low-end edge devices for autonomous driving applications.

1.2.2 Problem Statement

The goal of this work is to design and develop faster, light weight and more efficient object detection algorithms which can detect multi-scale objects, smaller and occluded targets accurately without losing detection accuracy and be able to run on any low-end edge device in near real-time. Therefore, the developed algorithm should be light-weight, smaller, have higher inference speed, lower memory footprint and consume low power. So, the problem can be stated as "Design and Development of efficient Embedded Vision based Object Detectors". The outline of the thesis is shown in Figure 1.3. Chapters 3 and 4 employ traditional approach and Chapters 5 and 6 employ efficient approach.



(a) Traditional Approach.



(b) Efficient Approach.

Figure 1.3. Thesis Contribution.

1.3 Research Objectives

The objectives of the proposed work are to design faster, light-weight and more efficient embedded vision-based object detection networks for autonomous driving application, while detecting multi-scale, tiny and occluded targets and also to achieve a trade-off balance between detection accuracy and speed.

1. Develop a robust detection algorithm to detect smaller and densely distributed pedestrians accurately in real-time road scenes without losing any detection accuracy.

2. To design and develop an efficient object detection algorithm in order to achieve optimal trade-off balance between detection accuracy and speed while detecting small-scale and occluded objects for autonomous driving system.

3. To design and develop light-weight object detector for detecting smaller pedestrians accurately without losing any detection accuracy, and implement the model in real-time on a low-end edge device.

4. To design and develop an efficient, light-weight and rapid object detector which is particularly good at detecting multi-scale targets, tiny and occluded denser targets (pedestrians and vehicles) accurately and effectively in real-time on a low-end edge device Jetson TX2.

1.4 Thesis Organization

The thesis is organized into seven chapters. This section gives a summary of all chapters.

Chapter 1: Gives the introduction, background, and rationale for research.

Chapter 2: The second chapter begins with an overview of traditional and deep learning-based object detection, pedestrian detection and vehicle detection techniques. It also puts forth the idea of deploying deep learning models into various embedded platforms for real-time object detection. Then, some techniques and evaluation methods used in the literature to mitigate object detectors for autonomous driving application are discussed.

Chapter 3: In this chapter, we introduced YOLOv2PD network, since state-of-the-art detectors such as SSD and Yolov3 achieve higher detection accuracy, but they fail to detect smaller and denser objects accurately. Two networks were proposed and validated the robustness of YOLOv2PD network. The performance of the proposed detector was validated on multiple datasets and compared with state-of-the-art pedestrian detectors.

Chapter 4: In this chapter, to handle and detect multi-objects effectively, Single shot Multi-scale Detector (SSD) performs detection on the extracted multi-scale feature maps but fails to detect smaller and occluded objects accurately, since shallow layers lack semantic information.

To overcome this problem, the concept of Single-shot Multi-scale Attentive Detector (SSAD) was introduced which incorporates pixel-wise feature relations and follows human vision mechanism. To improve detection accuracy of occluded objects, a multi-scale attention unit (MAU) is embedded into SSAD model. The performance of SSAD model was validated on multiple datasets and compared with state-of-the-art object detectors in terms of efficiency and detection accuracy.

Chapter 5: In this chapter, Optimized MobileNet+SSD was developed for detecting smaller pedestrians accurately in real-time and the same model was implemented effectively on low-end edge device Jetson Nano evaluation board. This model lets the components work in coordination in such a manner that their strengths are improved and the number of parameters decreased compared to SOTA detection architectures. A concatenation feature fusion module was inserted for adding contextual information in the network to improve detection accuracy of smaller pedestrians. The performance of optimized MobileNet+SSD was validated on multiple datasets and compared with state-of-the-art pedestrian detectors.

Chapter 6: Tackling real-time pedestrian and vehicle detection tasks captured by high-speed moving vehicle scenarios has two problems. One is target scale varies drastically and the captured images contain both tiny targets and high-density targets, which bring occlusion between targets. To solve the two issues, an efficient light weight real-time detection algorithm was introduced, which is referred to as EfficientLiteDet. One more prediction head is introduced in this model to detect multi-scale targets effectively and three transformer encoders were introduced in the neck part to improve detection of tiny and occluded denser targets. To focus only on particular targets, Convolutional block attention module (CBAM) was introduced. The performance of EfficientLiteDet was validated on multiple datasets and compared with state-of-the-art object detectors. The proposed model achieves real-time inference on edge device Jetson TX2, which ensures that EfficientLiteDet model can be used in real-world scenarios for autonomous driving application.

Chapter 7: This chapter presents a summary of the results and conclusions from work carried out in the earlier chapters. The limitations and future work research directions are also indicated.

Chapter 2

2. Literature Survey

This chapter is devoted to the review of research work reported in literature in the area of object detection, pedestrian detection, vehicle detection and its low-end embedded vision design. The consolidated findings and results of various researchers for different methods and techniques are discussed. The literature review was carried out in four parts. In the first part, review of various object detection techniques along its pros and cons was discussed. In the second part, review of Pedestrian detection techniques along its pros and cons was discussed. In the third part, a review of Vehicle detection techniques along its pros and cons was done. At last, deployment of CNN models on low-end edge devices was discussed.

2.1. Object Detection

Object detection is applied in wide areas of computer vision (CV), including defence (surveillance), iris recognition, face detection, Human-computer interaction (HCI), robot vision, security, medical imaging, smart transportation, automated vehicle systems, image retrieval system, machine inspection, etc. For continuous video surveillance for a few hours, sensors generate petabytes of image data. The generated data is further reduced to geospatial data and then integrated with other collected data to generate a clear-cut picture of the current situation. One of the most important tasks involving object detection is to track vehicles/suspicious people from collected raw data [1]. Crucial applications of object detection include detecting faulty electric wires, detecting unattended baggage, detecting driver drowsiness on highways, detecting vehicles parked in restricted areas, detecting objects present or coming onto the road (for self-driving vehicles), and also detecting stray animals in industrial areas.

Generally, Object detection is classified into two types: traditional learning-based object detectors and Deep learning-based object detectors.

2.1.1. Traditional learning-based object detectors

P. Viola and M. Jones (VJ) performed facial detection without any restrictions (skin colour segmentation) [11, 12]. The implementation of VJ detector is simple and straight forward; i.e., a sliding window is moved along all possible locations and the image is scaled; then it checks whether a human face is present in any of the windows. Using VJ detector, detection speed improved drastically by including three techniques; integral image, detection cascades and feature selection. N. Dalal and B. Triggs [7] first implemented the “histogram of oriented gradients” (HOG) feature descriptor which was an improved version of “scale-invariant feature transform” (SIFT) [4], [24] and shape contexts [5]. HOG descriptor computes on dense grid cells. To balance both feature invariance (which includes translation and illumination) and linearity (on discriminating different objects classes), overlap local contrast normalization (on blocks) is applied, which improves detection accuracy. HOG detector is not only used in pedestrian detection but also to detect multiple object classes.

The deformable part-based model (DPM) [6] was at its peak for traditional object detection methods and was the winner of VOC detection challenges in 2008 and 2009. DPM is an improved version of HOG detector. Initially, DPM was implemented by P. Felzenszwalb [15], but later R. Girshick [10, 25, 26, 27] made refinements to DPM detector. The main theme of DPM is “divide and conquer”, where the training period is the proper way of learning, and decomposition of objects and the inference period are considered an ensemble of different parts in object detection.

2.1.2. Deep learning-based object detectors

Deep learning-based Object detectors is further classified into two groups. They are “one-stage detection” that completes in one step and “two-stage detection”, which completes with "coarse to fine" stages. RCNN [10] architecture, using selective search method [28] extracts a set of object region proposals. Each object region proposal is transformed into a fixed image size by rescaling it, and then applied to convolutional neural network model which is pre-trained on ImageNet, i.e., AlexNet [29], for feature extraction. SVM classifier [30] predicts the object presence within each region proposal and also recognizes object classes. RCNN method, drawbacks are: It consumes more time to train the network, as we need to classify 2000 object region proposals per image, and it cannot be implemented in real-time.Fast RCNN detector [11] was implemented by R. Girshick and is an improvement on RCNN [10]. Fast RCNN allows us to train simultaneously both detector and bounding box regressor; mean average precision (mAP) accuracy increased from 58.5% (RCNN) to 70.0%

(Fast RCNN) on PASCAL VOC-2007 dataset [100]. All the advantages of RCNN and SPPNet are successfully integrated with Fast RCNN, but still, the detection speed is limited. A Faster RCNN detector [12] was implemented shortly after the introduction of Fast RCNN by S. Ren et al. To overcome the drawbacks of Fast RCNN, a network referred to as region proposal network (RPN) was introduced in Faster RCNN. Fast RCNN performs both region proposal generation and detection tasks. Except for RPN, Faster RCNN and Fast RCNN are very similar. Initially, first ROI pooling is performed, and then the pooled area is fed to CNN and two fully connected layers are used for softmax classification and bounding box regressor. It is the first near-real-time object detector tested on MS-COCO dataset [101]; it achieved mAP = 42.7%, VOC-2012, mAP = 70.4%, and 17 fps with ZFNet [31]. Despite Faster RCNN being much faster than Fast RCNN, there is computational redundancy at the final stage.

He et al. [13] introduced Mask R-CNN, and it is the extension of Faster RCNN. The main aim of Mask RCNN is to solve instance segmentation problems in CV applications; i.e., to separate different objects in an image or a video. Additionally, a mask branch on each region of interest (ROI) is included in Mask R-CNN for predicting an object works in parallel with class label and bounding box (BB) regression branches. It is conceptually simple to train, flexible, and is a general framework for instance segmentation of objects. But the main drawback is it adds small computational overhead on the network and runs with a speed of nearly 5 Fps.

R. Joseph et al. [14] implemented YOLO architecture. YOLO is the strongest, fastest, and simplest object detection algorithm used in real-time object detection. All previous object detection algorithms use regions to localize objects within the image, but the YOLO approach is entirely different; the entire image is applied to a single CNN. YOLO network splits the entire image into regions, and for each region, it predicts bounding boxes and class probabilities. The main drawbacks of YOLO object detector are: detection of small objects in an image, where localization accuracy drops off when compared to two-stage detectors. To the basic YOLO detector, R. Joseph [14] later made improvements and implemented YOLOv2 and YOLOv3 [15, 16] which have achieved better detection accuracy without scarifying detection speed.

Liu et al. [19-21] implemented a series of single shot multi-box detector (SSD) for detecting objects. This is designed purely for real-time object detection in the deep learning era. Instead of taking two shots as in RCNN series, one for generating region proposals and another for detecting the object of each proposal, it uses only a single shot to detect multiple objects within an image. To improve the detection accuracy of SSD, particularly in detecting small objects, they introduced “multi-reference and multi-resolution detection” techniques. To improve Fast RCNN’s real-time speed detection accuracy, SSD eliminated region proposal network (RPN). The main drawback of SSD is at the cost of speed, detection accuracy increases with increase in number of default boundary boxes. SSD detector has more classification errors when compared to RCNN but low localization errors while dealing with similar categories.

Wu et al. [32] implemented SqueezeDet, a lightweight, single shot, extremely fast, fully-CNN for detecting objects in an autonomous driving system. To deploy Deep CNN for real-time object detection, the model should address some important problems, such as speed, accuracy, model size, and power efficiency. It is a single forward pass object detector, used to extract a high dimensional, low-resolution feature maps for the applied input images; it uses stacked convolution filters. Second, it uses ConvDet, a convolutional layer fed with a feature map as input that produces a large number of bounding boxes and also predicts the object’s category. Finally, by applying filtering to these bounding boxes, it outputs final object detections. The backbone model of SqueezeDet is SqueezeNet [33], and the model size is less than 8 MB, which is very small compared to AlexNet [29] without losing any accuracy. For an input image of size 1242x375, this model achieved 57.2 frames per second (FPS) on the KITTI dataset [106] and consumed only 1.4 joules energy per image.

Law et al. [35] implemented CornerNet for object detection, wherein the object is detected by a pair of key points using a CNN instead of drawing an anchor box around the detected object. So, the need for designing anchor boxes which are usually used in one stage detectors is eliminated by detecting objects as paired key points; i.e., top-left and top-right corners respectively. They introduced a new type of pooling layer referred to as corner pooling, which helps the network to localize corners better. The CNN outputs the heat map for all top-left corners and bottom-right corners, along with an embedded vector map for each detected corner. On MS-COCO dataset [101], CornerNet achieved 42.2% average precision (AP) which outperforms existing one-stage detectors. So to overcome this drawback, Duan et al. [36] implemented CenterNet by introducing a third key point at the center to detect each object. CenterNet achieved 47% AP, and inference speed is slower than CornerNet.

2.2. Pedestrian Detection

An important application area of object detection is pedestrian detection. Pedestrian detection is indispensable both in unmanned and advanced driver-assisted driving systems, and researchers' efforts have been directed to this area. It is used extensively in complex applications, including video surveillance, self-driving cars, etc. Earlier pedestrian detection methods used were HOG detector [7] and the integral channel features (ICF) detector [36], which rely purely on feature representation, classifier design [37], and acceleration detection [38]; others, such as "detection by components" [40], gradient-based representation [41], and deformable part-based model (DPM) [6,25,26], are time-consuming, require complex steps, are expensive and require a high level of human interference.

One of the direct applications of real-time pedestrian detection is that it should automatically locate pedestrians accurately with on-shelf cameras, since it plays a crucial role in robotics and unmanned driving systems. Despite tremendous progress having been achieved recently, this task still remains challenging due to the complexity of road scenes, such as them being crowded, occluded, containing deformations and exhibiting lighting changes. Currently, unmanned driving systems are among the major fields of research in computer vision, for which the real-time detection of pedestrians is essential to avoid possible accidents. Although deep learning-based techniques improve detection accuracy, there is still a huge gap between human and machine perception. Therefore, when detecting objects in a shadowy environment or objects captured at night, lower detection accuracy is achieved. This is the major drawback of reliable vision-based detection systems since self-driving cars in real-time extremely complex environments should be able to detect objects in daytime or at night. Nevertheless, current state-of-the-art (SOTA) real-time pedestrian detection still falls short of the fast and accurate human perception levels.

The first computer vision task that applied deep learning was pedestrian detection [42]. Real-time pedestrian detection using the Improved Tiny Yolov3 network [43] was proposed by Yi. This network applies the K-means clustering algorithm prior to the dataset training dataset to find the best prior bounding boxes in order to achieve better detection accuracy. The network effectiveness is validated only on the extracted pedestrian related images from PASCAL VOC-2007 [100] dataset. But this network fails to achieve better detection accuracy while detecting smaller and dense pedestrians. R Benson et al. [44] implemented the fastest technique to achieve a frame rate of 100 frames per second (FPS) for pedestrian detection. After 2012, the deep learning era started, which has greatly improved the detection accuracy of pedestrian detection. However, their run time on each image is slightly or markedly slower, taking a few seconds.

S. Paisitkriangkrai et al. [45] proposed, constructed a network based on low-level vision features and incorporated spatial pooling to improve translational invariance which in turn improves the robustness of pedestrian detection process. They optimize directly the area under the ROC curve and which helps to focus on detection performance. The ConvNet [46] method uses convents for detecting pedestrians. It employs convolutional sparse coding to initialize each layer at the start and later performs fine-tuning to perform object detection. RPN-BF [48] is a perfect fusion of Region Proposal Networks (RPN) and Boosted Random Forest Classifier. RPN proposed in Faster RCNN [12] generates candidate bounding boxes, high-resolution feature maps, and confidence scores. To shape Boosted Forest Classifier, it also employs real-boost algorithm for using information obtained from RPN. The two-stage detector showed good performance results on pedestrian test datasets.

Li Z et al. [47] proposed a network structure which integrates both region generation and prediction modules for accurate localization and speed up parallel processing for real-time small-scale pedestrian detection. They adopted feature pyramid strategy in the generation module to extract useful features, at the other end, higher level contextual features were extracted using deconvolution layers. The proposed network performance was validated on INRIA [102], CALTECH [103] and ETH [105] pedestrian datasets.

Li J et al. [49] proposed scale-aware Fast-RCNN method for detecting pedestrians of various scales, and applied anchor box mechanism onto multiple feature layers. They proposed multi-sub networks which are used to detect multi-scale pedestrians effectively and outputs from all sub-networks were combined adaptively to produce final detected results that are more robust to large variance in instance scales. This algorithm selects weights of large-size or small-size sub-network based on the proposal height. However, this method is not accurate, if a large proposal contains very small targets. The effectiveness of this network was validated on the widely used pedestrian datasets such as INRIA, ETH and CALTECH. In addition, Ouyang et al. [50] proposed a unified deep neural network for jointly learning four key components, namely, feature extraction + deformation + occlusion and classification for pedestrian detection This network performance was validated on CALTECH old and new datasets [103].

Y Pang et al. [51] introduced a mask-guided attention network for detecting occluded pedestrians, which emphasizes only visible regions and suppresses occluded regions by modulating full body features. The proposed network performance was validated on both CALTECH and CITY PERSONs datasets. However, this method fails to achieve satisfactory results on heavily occluded pedestrians. Detecting occluded pedestrians is very hard as their appearance varies greatly depending upon the occlusion rate. To handle occlusion, Zhang et al. [52] proposed a simple and compact method by incorporating a channel-wise attention network on Faster RCNN detector while detecting occluded pedestrians. The proposed network performance was validated on two pedestrian datasets CALTECH and CITY PERSONs.

Song et al. [53] proposed a novel method by integrating both somatic topological line localization and temporal feature aggregation for detecting small-scale pedestrians, who are relatively far from the camera. This method also eliminates ambiguities in occluded pedestrians by introducing a post-processing scheme based on Markov Random Field (MRF). This network achieved competitive performance on both CALTECH and CITY PERSONs pedestrian datasets.

Y Zhang et al. [54] proposed "key-point-guided super-resolution network" (KGSNet) for detecting small-scale and heavily occluded pedestrians. Initially, this network is trained to generate a super-resolution pedestrian image and then a part estimation module encodes the semantic information of four human body parts. In this network, super resolution and classification networks were optimized alternatively and trained in end-to-end fashion. The proposed network performance is validated on two widely used pedestrian datasets CALTECH and CITY PERSONs. This network was able to achieve best results on small-scale and heavily occluded pedestrians over existing state-of-the-art (SOTA) pedestrian detectors.

C Lin et al. [55] proposed a graininess-aware feature learning method for detecting small-scale and occluded pedestrians. Attention mechanism was used to generate graininess-aware feature maps and then to enhance the features, a zoom-in-zoom-out module was introduced. By integrating the two modules into CNN, they formed an end-to-end trainable pedestrian detector. The proposed network performance was validated on three widely used pedestrian datasets CALTECH, INRIA and KITTI [106].

W. Y Hsu et al. [56] proposed a new ratio-and-scale-aware YOLO (RSA-YOLO) which achieves extremely good results while detecting small-scale pedestrians. They introduced ratio-aware mechanism for dynamically adjusting various hyperparameters of YOLOv3 network. In order to solve the misdetection of extremely small pedestrians, they introduced scale-aware mechanism in which they used multi-resolution fusion technique. The proposed network performance was validated on INRIA, ETH and PASCAL VOC-2012 pedestrian datasets.

Moreover, existing algorithms could achieve higher detection accuracy but they fail to detect small-scale pedestrians who are relatively far from the cameras. To handle this issue, B Han et al. [57] proposed a novel small-scale sense (SSN) network, which can generate some proposal regions and is effective when detecting small-scale pedestrians. Based on cross entropy loss they proposed novel loss function to increase the loss contribution of small-scale pedestrians which are hard to detect. This network performance was validated on two pedestrian datasets CALTECH and VIP [57].

Yang Fan et al. [58] proposed SSD for online pedestrian detection on input fed video using Kalman filter. They adopted SSD as baseline detector and fusion modules to improve the detection accuracy while detecting medium and far-scale pedestrians. This method performs post-processing together by combining the fusion module and Kalman filter so that it effectively reduces miss rate and provides better performance at faster speed. The proposed network performance was validated on CALTECH and Private [58] datasets.

Cheng. Y et al. [59] proposed an enhanced SSD for detecting small-scale pedestrians. The proposed pedestrian detection with enhanced SSD depicts small-scale objects and most of the missed pedestrian targets (dense targets). They adopted SSD as baseline detector and made dense connections among convolutional blocks in order to improve detection performance while detecting small-scale pedestrians. They also modified matching strategy of SSD and added extra loss that can keep more information on denser objects. This network has performed experimentation on CALTECH and PASCAL VOC datasets.

G Brazil et al. [60] proposed a pedestrian detection network to handle occlusion problem in urban autonomous driving scenario. They leverage semantic segmentation on their network to boost pedestrian detection accuracy. To enable joint supervision on both pedestrian detection and semantic segmentation they introduced semantic segmentation infusion network. The proposed network performance was validated on both CALTECH and KITTI pedestrian datasets.

Z.Cai et al. [61] designed a complexity-aware cascaded pedestrian detector by combining features of different complexities. They formulated cascaded learning as Lagrangian optimization that accounts for accuracy and complexity. To optimize this, they introduced complexity aware cascade training (CompACT) technique. This method is able to achieve optimal trade-off balance between detection accuracy and complexity by pushing higher features to later cascaded stages. Therefore, the proposed network enables to use high complexity features in a single detector. This network performance is validated on CALTECH and KITTI datasets and surpasses SOTA detector results.

Z Liu et al. [62] proposed an efficient pedestrian detector which would achieve good trade-off balance between detection accuracy and speed. They adopted YOLOv2 as a baseline detector and then modified the network structure and number of network parameters and made the network more suitable for detecting pedestrians. To achieve higher detection accuracy on CALTECH dataset, they employed weak semantic segmentation after shared layers and scale aware structure in the proposed network. They performed experiments on INRIA and CALTECH pedestrian datasets.

X Du et al [63] developed an efficient pedestrian detector by fusion of multiple deep neural networks. They adopted SSD as baseline detector to generate pedestrian candidates. Next, to provide majority ground truth pedestrian annotations, a candidate generator was designed. Then ensemble learning classifier was employed to improve detection accuracy. Finally, the classifier, classifies the generated candidates based on the opinion of fusion network and multiple deep verification networks by utilizing soft-rejection fusion technique. This network performance is evaluated on KITTI, INRIA, ETH and CALTECH datasets. Cao Jingwei et al. [64] proposed an intelligent pedestrian detection algorithm to work under complex scenarios. They adopted YOLOv3 and modified the grid size, and improved prediction head based on receptive field and applied soft-NMS algorithm. This proposed network detection performance was validated on INRIA and PASCAL VOC 2012 datasets.

To maintain a trade-off balance between detection accuracy and speed, one stage pedestrian detector was proposed by Ma J [65]. First, they designed multi-scale convolution module to extract corresponding features at multiple scales. Second, they applied attention mechanism module across channels to represent various occlusion patterns. Finally, the enhanced features were passed through classification and regression modules to perform object localization and anchor box regression. The designed method was extensively performed on CALTECH and CITYPERSONs pedestrian datasets.

In pedestrian detection, the main difficulties and challenges faced are (a) small pedestrian detection; (b) hard negatives; (c) real-time pedestrian detection from HD video; (d) dense and occluded pedestrians. The widely used datasets for evaluating the pedestrian detection performance are PASCAL VOC, INRIA, CALTECH, ETH, KITTI and CITY PERSONs.

2.3. Vehicle Detection

Like pedestrian detection, vehicle detection is very necessary both in unmanned and advanced driver-assisted driving systems. Recently traditional handcrafted feature methods such as HOG [7] and Haar-wavelet [66] have been used to extract local features. Initially HOG features were used only for pedestrian detection, and then modified and used for vehicle-detection applications as well. Relevant study of both hand-crafted and deep CNN's based techniques are discussed in [9]. ACF [67] was refined and applied to vehicle detection area but it continues to produce unsatisfactory results. This is due to intra-class interferences because of different perspectives, which becomes a negative factor for vehicle detection.

Wu J et al. [68] constructed improved YOLOv2 algorithm to solve issues such as lack of vehicle-type recognition, low speed and low detection accuracy. They applied k-means++ algorithm to figure out optimal k anchor boxes prior to network training on vehicle dataset. This model achieved 94.78% mAP when validated on Beijing Institute of Technology (BIT)-Vehicle dataset [107]. Song H et al. [69] constructed a highway vehicle dataset which has many small annotated vehicles, and provided a complete dataset for vehicle detection. The proposed network employed new segmentation method to extract image features and divide the image into remote area and proximal area and then deploy in YOLOv3 network. They trained and tested the constructed highway vehicle dataset [69] on modified YOLOv3 network.

X Wang et al [70] introduced SPP module in Tiny-YOLOv3 and modified grid size to improve the detection accuracy of dense vehicles. They combined context feature information and increased two scale detections to three in the proposed network. In order to improve denser vehicle detection, they modified grid size and figure out the best number of anchor boxes by applying k-means clustering algorithm on KITTI dataset, prior to network training. From the experimental results, this network achieves 91.03% detection accuracy and 144 FPS on KITTI dataset. Limited computational power and memory constraints on an autonomous vehicle limit the deployment of deep neural networks on edge device. Therefore, Chen L et al. [71] proposed a light weight and faster vehicle detector by applying group convolution on DenseNet. By applying group convolution in the proposed network, they could reduce floating point operations and in turn make the detector as light weight and fast as possible. They designed certain guidelines to optimally select the group number in group convolution which maximizes the overall detection speed. They performed experiments on both PASCAL VOC 2007+12 and vehicle [71] datasets.

Liu W et al. [72] proposed a two-stage vehicle detector especially for detecting tiny vehicles in traffic complex scenes. The proposed network has two stages, backward feature enhancement network (BFEN) stage and a spatial layout preserving network (SLPN) stage. In the first stage, high quality region proposals for multi-scales were generated, and in the next stage, it progressively integrates ROI features, while preserving spatial layouts. This network performance was validated on KITTI and DETRAC [108] vehicle datasets. Jamiya S et al. [39] proposed a lightweight deep-CNN model named Little-YOLO-SPP to predict vehicles in darker environments and achieve real-time performance. To extract best features from the dataset they improved feature extraction layers using SPPNet. In order to improve the detection performance, they proposed Generalized IoU loss function for anchor box regression. This network robustness was verified and achieves 77.44% mAP on PASCAL VOC 2007, 2012 [100] and 52.95% mAP on MS COCO 2014 [101] datasets. This network fails to detect some vehicles and few false labelling of detected vehicles.

X. Hu et al. [74] proposed a scale-insensitive CNN model i.e SINet to overcome scale-sensitivity problem for fast vehicle detection task. This network proposes, context-aware ROI pooling to maintain contextual information and retain the original structure of small-scale objects, and to minimize intra-class distance of features they constructed a multi-branch decision network. This network detection performance was validated on two datasets, namely KITTI and HIGHWAY [69] datasets. Hoanh Nguyen et al. [75] proposed an improved Faster RCNN for faster vehicle detection to overcome large vehicle scale-variation, heavy occlusion, or truncation of the vehicles in the captured images. This model adopted MobileNet as backbone network, and adopted soft NMS algorithm on Faster RCNN to solve the issue of duplicate region proposals. The model performance was validated on KITTI vehicle dataset and achieved higher detection accuracy and speed compared to Fast RCNN. Xueru Dai et al

[76] proposed HybridNet, a faster two-stage cascaded vehicle detection system. This network solves the issue of large intra-class differences caused by occlusion, truncation and different viewpoints. This network has regression modes in each stage and incorporates a transitional stage to map generated proposals from the first stage to the next stage for refinement of extracted features. This model is validated only on KITTI and PASCAL VOC datasets.

The widely used datasets for evaluating the vehicle detection performance are PASCAL-VOC [100], Highway [69], KITTI [106] and Udacity [109].

2.4 Deployment of CNN models on low-end Edge devices

The primary requirements to be fulfilled for real-time object detection using deep learning on any embedded platforms are the following: high accuracy, high speed, small model size, and better energy efficiency. The various embedded platforms available for real-time object detection using deep learning are FPGA and Nvidia Jetson Variants such as Zynq, Pynq, Jetson Nano, Jetson TX1, TX2 and Xavier evaluation boards.

Subarna Tripathi et al. [151] proposed LcDet, a fully-convolutional neural network for generic object detection that aims to work in embedded systems. They designed and develop an end-to-end TensorFlow (TF)-based model and additionally, employed 8-bit quantization on the learned weights. LcDet achieved better accuracy compared with state-of-the-art CNN-based face detection methods, while reducing the model size by 12x and memory-BW by 16x compared to one of the best real-time CNN-based object detectors YOLO. Huizi Mao et al. [152] proposed a pipelined object detection implementation on the embedded platform. They made some additional modifications on Fast R-CNN method to fit the specific platform and achieve trade-off between speed and accuracy on embedded systems in order to make full use of limited computation resources. This detection system can run Fast R-CNN at 1.85 FPS. Chen L et al. [71] proposed a light weight and faster vehicle detector by applying group convolution on DenseNet. By applying group convolution in the proposed network, they could reduce floating point operations and in turn make the detector light weight and faster. They performed experiments on both PASCAL VOC 2007+12 and vehicle [71] datasets and implemented the model in real-time on Jetson TX2 and achieved 10FPS.

Chapter 3

YOLOv2PD: An Efficient Pedestrian Detection Algorithm

This chapter focuses on design and development of an efficient pedestrian detection algorithm for accurate smaller and densely distributed pedestrians in real-time road scenes without losing any detection accuracy. To show the effectiveness of the proposed model, yet another model named YOLOv2 Model A was proposed and the robustness of YOLOv2PD, and YOLOv2 Model A models verified, particularly while detecting small-scale and densely distributed pedestrians. Finally, the proposed network performance was validated on multiple pedestrian datasets and compared with state-of-the-art (SOTA) algorithms.

3.1 Introduction

Commonly, most of the object detection algorithm were aiming to detect one category of objects, such as animals, faces, humans or vehicles. In this work, pedestrian is employed as one object to be detected. Real-time pedestrian detection is an important task for unmanned driving systems and video surveillance. One of the direct applications of real-time pedestrian detection is that it should automatically locate pedestrians accurately with on-shelf cameras, since it plays a crucial role in robotics and unmanned driving systems. Despite tremendous progress having been achieved recently, this task still remains challenging due to the complexity of road scenes, such as them being crowded, occluded, containing deformations and exhibiting lighting changes. Currently, unmanned driving systems are among the major fields of research in CV, for which real-time detection of pedestrians is essential to avoid possible accidents. Although deep learning-based techniques improve detection accuracy, there is still a huge gap between human and machine perception [47]. A complex background, low-resolution images, lighting conditions, and occluded and distant smaller objects reduces the model accuracy. To date, most researchers in this field have focused only on colour-image-based object detection. Therefore, when detecting objects in a shadowy environment or objects captured at night, lower detection accuracy is achieved.

This is the major drawback of reliable vision-based detection systems since self-driving cars in real-time extremely complex environments should be able to detect objects in daytime or at night. Nevertheless, current state-of-the-art (SOTA) real-time pedestrian detection still falls short of fast and accurate human perception levels [77].

Currently, pedestrian detection methods are classified into two categories: traditional and deep learning-based methods. Traditional methods cover various traditional machine learning algorithms such as Voila Jones (VJ) detector [22], Deformable part model (DPM) [25], Histogram of oriented gradient (HOG) [7] and multi-scale gradient histograms [78]. These methods are time-consuming, require complex steps, are expensive, and require a high level of human interference. In the recent evolution of deep learning techniques since 2012, such techniques have become very popular and deep convolutional neural network (CNN)-based pedestrian detection methods have achieved better performance than traditional time slot methods [79-80]. The first deep learning-based object detection model was RCNN [10]. This method generates a region of interest by using a selective search window for deep learning-based object detection, implemented in all RCNN series. Deep learning time slot methods cover both two-stage detectors such as RCNN [10], Fast-RCNN [11], Faster RCNN [12], Mask-RCNN [13], SPPNet [86] and single-stage detectors such as YOLO [14-17] and SSD [19-21] series. Therefore, in the current scenario for real-time pedestrian detection, these methods are not quite suitable.

Generally, the speed of deep learning-based object detection methods is low, with these methods being unable to meet real-time requirements of self-driving cars. Therefore, to improve both speed and detection accuracy, YOLO network was proposed [14], as a single end-to-end object regression detector. A year later, YOLOv2 [15] was developed to overcome the drawbacks of the YOLO [14] network. To improve the speed of the detection algorithm without losing any part of detection accuracy, YOLOv2 [15] was proposed. However, when detecting smaller objects in complex environments, it achieves low detection accuracy. Therefore, in this chapter, YOLOv2 (“You Only Look Once Version 2”) based pedestrian detection algorithm (referred to as YOLOv2PD) was proposed, which would be more suitable for detecting smaller and densely distributed pedestrians in real-time complex road scenes.

The contributions of this work are summarized as follows:

- YOLOv2PD model adopts the multi-layer feature fusion (MLFF) strategy to improve the model's feature extraction ability and, at the higher end, one convolution layer is eliminated.
- Moreover, intuitively, to test the effectiveness of YOLOv2PD model, another model referred to as YOLOv2 Model A is implemented and compared.
- The loss function of YOLOv2PD model is improved by applying normalization, which reduces the effect of different pedestrian sizes in an image, and which potentially optimizes the detected bounding boxes.
- Through qualitative and quantitative experiments conducted on Pascal Voc-2007+2012 Pedestrian, INRIA, Caltech pedestrian and City Persons datasets, we validate the effectiveness of YOLOv2PD algorithm, showing that it has better detection performance on smaller pedestrians.

3.2. Overview of YOLO and YOLOv2

The YOLOv2 [15] model performs an end-to-end training CNN to detect smaller and densely distributed pedestrians in near real time. YOLO [14] is a fast one-stage object detector that directly predicts the class probability and bounding box for detected objects. Initially, the applied input image is divided into S × S grid cells and every grid cell can predict only one object, as shown in Figure 3.1. If the object center falls into a grid cell, then that grid cell is solely responsible for detecting that object. Each grid cell directly predicts class probabilities, confidence scores, and B bounding boxes. Each bounding box (BB) has five components: (x, y, w, h, confidence). (x, y) coordinates refer to the box center with respect to grid cell location. (w, h) coordinates refer to the BB width and height dimensions. These (x, y, w, h) coordinate values are normalized in the range of 0 to 1. C refers to the box confidence score, which reflects how likely the BB contains an object and how accurate the BB is. The box confidence score will be zero if the object does not exist in that grid cell. Otherwise, Intersection over union (IoU) between the ground truth and predicted BB should be equal to the confidence score. Then, under a given threshold, most of the boxes with low confidence scores are removed. To eliminate redundant BB, non-maximum suppression is finally applied.

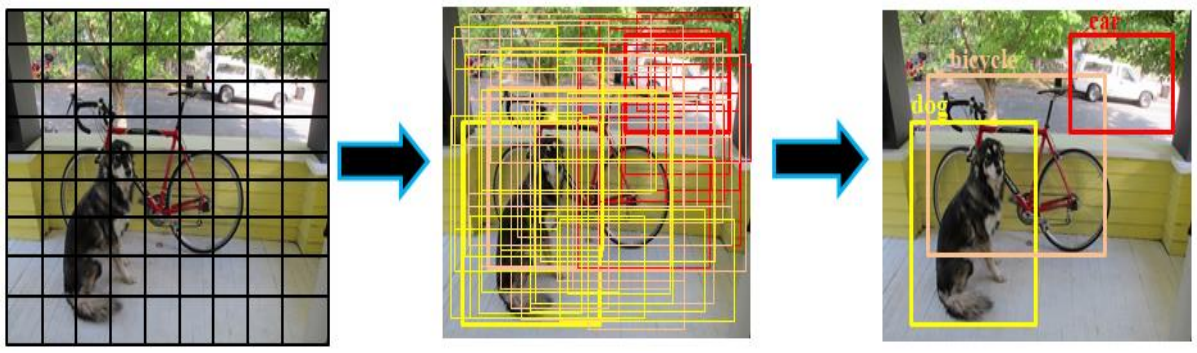


Figure 3.1 Flow of YOLO detection algorithm.

Since each cell generates B bounding box predictions, total outputs relative to BB predictions are $S \times S \times B \times 5$. There is also the possibility of multiple objects existing in a single cell. To overcome this problem, the anchor box mechanism is introduced. Therefore, the anchor box mechanism makes it possible for YOLOv2 to detect multiple objects falling in a single cell. By predefining the number of anchor boxes to be used in the model, and as a result of the anchor box, one more dimension is added in the output labels, one object is assigned to each anchor box.

Table.3.1 shows YOLOv2 [15] network architecture, which consists of 30 layers (L0 to L29), including 22 convolutional layers, 5 max. Pooling layers, and two route and one reorg layers. The 25th and 27th layers are referred to as route layers. The aim of the route layer is to merge two different convolutional layers; for example, L27 (Layer 27) is formed by merging two layers: L26 and L24. Finally, the extracted features from the previous layer are reorganized to predict the BB and probability of the pedestrian at the detection layer.

An input image of fixed size (416×416) is fed into the detection algorithm. Table 3.1 shows the flow of YOLOv2 algorithm in terms of how the image size changes its dimensions. To the end at L29 (layer 29), the output image size is $13 \times 13 \times 30$, which is finally reduced to a 13×13 grid size. Each cell generates 30 output values (i.e., 5×6), where 5 represents 5 predictive borders generated by every 13×13 grid cell and the remaining 25 (i.e., 30-5) correspond to each predictive border outputting 25. Of these 6 values, one value corresponds to the probability of a pedestrian. The remaining five values correspond to the box center position (t_x, t_y), width and height of BB (t_w, t_h) and box confidence score.

Table 3.1 YOLOv2 Network Architecture.

Layer No.	Conv.Type & Input	Filters	Conv.Output
L0	Conv_3*3_416*416*3	32	416*416*32
L1	Max.pool/2		208*208*32
L2	Conv_3*3_208*208*32	64	208*208*64
L3	Max.pool/2		104*104*128
L4	Conv_3*3_104*104*64	128	104*104*128
L5	Conv_1*1_104*104*128	64	104*104*64
L6	Conv_3*1_104*104*64	128	104*104*128
L7	Max.pool/2		52*52*128
L8	Conv_3*3_52*52*128	256	52*52*256
L9	Conv_1*1_52*52*256	128	52*52*128
L10	Conv_3*3_52*52*128	256	52*52*256
L11	Max.pool/2		26*26*256
L12	Conv_3*3_26*26*256	512	26*26*512
L13	Conv_1*1_26*26*512	256	26*26*256
L14	Conv_3*3_26*26*256	512	26*26*512
L15	Conv_1*1_26*26*512	256	26*26*256
L16	Conv_3*3_26*26*256	512	26*26*512
L17	Max.pool/2		13*13*512
L18	Conv_3*3_13*13*512	1024	13*13*1024
L19	Conv_1*1_13*13*1024	512	13*13*512
L20	Conv_3*3_13*13*512	1024	13*13*1024
L21	Conv_1*1_13*13*1024	512	13*13*512
L22	Conv_3*3_13*13*512	1024	13*13*1024
L23	Conv_3*3_13*13*1024	1024	13*13*1024
L24	Conv_3*3_13*13*1024	1024	13*13*1024
L25	Route (L16)		26*26*512
L26	Reorg		13*13*2048
L27	Route (L26–L24)		13*13*3072
L28	Conv_3*3_13*13*3072	1024	13*13*1024
L29	Conv_1*1_13*13*1024	30	13*13*30

To improve object detection performance further, a pre-trained model was adopted. In many CNN versions, VGG-16 [87] is generally preferred as a pre-trained model. To improve both the accuracy and speed, another pre-trained model called Darknet-19 was adopted in Yolov2 algorithm. The detection speed of Yolov2 is 4× faster while it maintained the same accuracy as VGG-16.

For an input image of size 224 × 224, the detection performances of the models VGG-16 and Yolov2 were compared by Simonyan et al. [87]. From their comparisons, it was apparent that GoogleNet [88]+Yolov2 uses only 8.52 billion Flops (Floating Point Operations per second), VGG-16[87]+Yolov2 requires 30.69 billon Flops, whereas Darknet-19 [89]+YOLOv2 needs only 5.58 billion Flops. Since it is lightweight and has a smaller network, it is more suitable for real-time pedestrian detection.

3.3. Proposed Methodology

3.3.1 Anchor Boxes Selected Based on K-means Clustering

The proposed YOLOv2PD model applies K-means clustering [90] algorithm on Pascal Voc-2007+2012 pedestrian dataset during training and selects the optimal number of anchor boxes of different sizes. It works by replacing traditional Euclidean distance with the distance function of YOLOv2 while implementing K-means clustering algorithm. Therefore, the error obtained is made irrelevant with respect to anchor box sizes by adopting intersection over union (IoU) as an evaluation metric, as shown in equation 3.1.

$$d\ (\text{box, centroid}) = 1 - \text{IOU}\ (\text{box, centroid}) \qquad (3.1)$$

where box is the sample; centroid is cluster center point;

IoU (box, centroid) is the overlap ratio between cluster and center boxes.

Based on the clustering results analysis, the K value was chosen to be 6; therefore, six different anchor box sizes would be applied in order to improve the positioning accuracy. Finally, by implementing the K-means clustering algorithm on the training dataset, a suitable number of different anchor box sizes are selected for pedestrian detection, which in turn improves positioning accuracy.

3.3.2 Improved Loss Function

Since images are captured using a video surveillance camera, some of the pedestrian images might be bigger, with pedestrians being nearer the camera, while some pedestrian images might be smaller, with pedestrians being located far away from the camera during detection. Therefore, pedestrians would appear smaller in the image when they are far from the camera, and vice versa. As such, sizes may vary in the captured images, even though the pedestrian is identical.

During YOLOv2 training, objects of different sizes show different effects on the network and produce large errors, particularly for images with smaller and densely distributed objects. To overcome this drawback, loss calculation for bounding box (BB) width and height was improved by applying normalization. Equation (3.2) shows the improved loss function as:

$$\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \prod_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \\
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \prod_{ij}^{obj} [(\frac{w_i - \hat{w}_i}{\hat{w}_i})^2 + (\frac{h_i - \hat{h}_i}{\hat{h}_i})^2] + \\
&\sum_{i=0}^{S^2} \sum_{j=0}^{B} \prod_{ij}^{obj} [(c_i - \hat{c}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \prod_{ij}^{noobj} (c_i - \hat{c}_i)^2] + \\
&\sum_{i=0}^{S^2} \prod_{i}^{obj} \sum_{c \in classes} [p_i(c) - \hat{p}_i(c)]^2
\end{aligned} \tag{3.2}$$

where (x_i, y_i) coordinates represent the center of the box,

(w_i, h_i) coordinates are the width and height of the box,

c_i is confidence prediction, and $p_i(c)$ is the conditional class probability for class c in cell *i.*

$\hat{x}_i$, $\hat{y}_i$, $\hat{w}_i$, $\hat{h}_i$, $\hat{c}_i$, and $\hat{p}_i(c)$ are the corresponding prediction values of x_i, y_i, w_i, h_i, c_i, and $p_i(c)$,

λ_{coord} Corresponds to the weight of position loss, with a value of 5,

λ_{noobj} Corresponds to the weight of the classification loss, with a value of 0.5,

S^2: S x S grid cells, B: Bounding Boxes (BBs),

$\prod_{ij}^{obj}$ = 1, Corresponds to the j^{th} BB in cell *i* that is responsible for detecting the pedestrian, else 0,

$\prod_{i}^{obj}$ = 1, if the pedestrian is located in the cell *i,* else 0.

From Equation (3.2), the first term determines the BB localization loss error, the second term determines the BB confidence loss error with objects and without objects, and the third term determines the classification loss error. Equation (3.2) in the proposed method was compared with original YOLOv2 [16] $\frac{w_i - \hat{w}_i}{\hat{w}_i}$ and $\frac{h_i - \hat{h}_i}{\hat{h}_i}$ term is used instead of $w_i - \hat{w}_i$ and $h_i - \hat{h}_i$,which would reduce the effect of different pedestrian sizes in an image, and which in turn potentially optimizes the detected bounding box.

3.3.3 Network Design

Multi-layer Feature Fusion (MLFF) Approach: In pedestrian detection, variations among pedestrians include occlusion, illumination changes, colour, height, and contour, whereas local features exist only in the lower layers of CNN. Therefore, to use local features fully, an MLFF approach was implemented in YOLOv2PD. The Reorg aim is to keep feature maps of layers the same. Part (a) passes through the following 3 × 3 and 1 × 1 convolution layers and then a down-sampling factor of Reorg/8 is applied, as shown in Figure. 3.2. Similarly, part (b) and part (c) perform the same operations, but with down-sampling factors of 4 and 2, respectively. Part (a), (b) local features, and part (c) global features of one layer are fused. This is done so that the network would distinguish tiny differences among pedestrians and also improve the network understanding of local features.

YOLOv2 is a fast and accurate object detection model. YOLOv2 network can detect 9000 classes and variations among multiple objects are wide such as vehicles, fruits, sofas and animals. There are three repeated 3 × 3 × 1024 convolutional layers in YOLOv2 network. Generally, at the higher end, repeated convolution operation deals with multiple classes and widely differing objects, such as fruits, animals, and vehicles. However, our main concern is only detecting the pedestrian class and feature differences among pedestrians are minute. Thus, the model performance may not improve due to repeated convolution layers at the higher end and, due to their presence, the model becomes more complex. Therefore, repeated convolution layers are removed from the higher end in the proposed models. This strategy would achieve almost competitive performance and reduce the time complexity of Yolov2 network. Thus, three repeated 3 × 3 × 1024 convolution layers are reduced to two in the proposed model, as shown in Figure. 3.2.

A novel YOLOv2PD network structure is designed by adopting MLFF approach and one unwanted convolutional layer is removed at the higher end. Moreover, intuitively, to test the effectiveness of the proposed model, another model, referred to as YOLOv2 Model A, was implemented and compared. YOLOv2 Model A removed two 3 × 3 × 1024 convolution layers and YOLOv2PD model removed only one 3 × 3 × 1024 convolution layer when compared with YOLOv2 network.

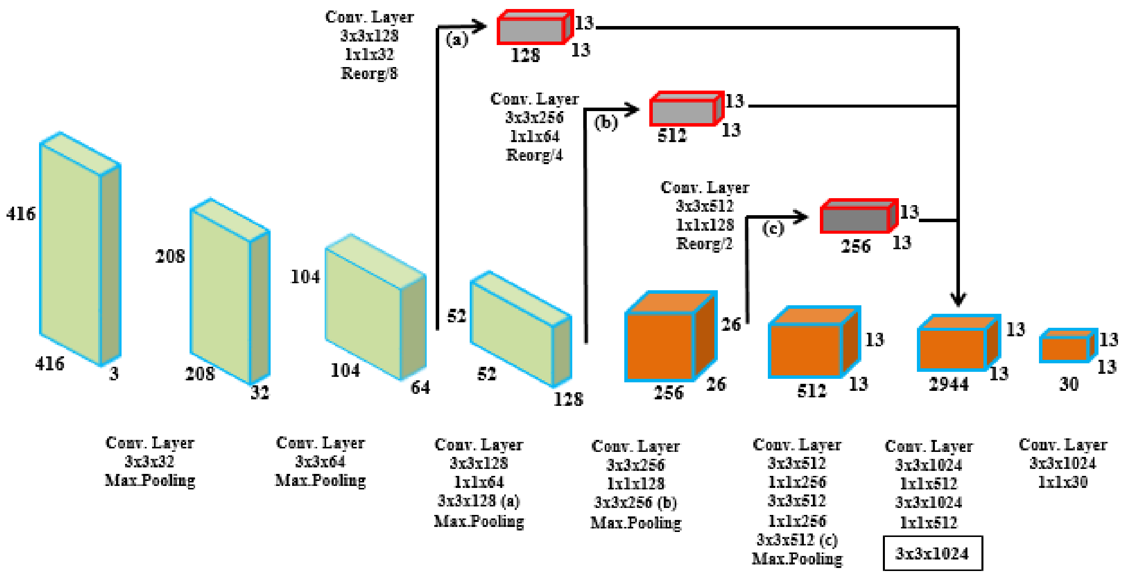


Figure 3.2 YOLOv2PD Network Architecture.

Table. 3.2 shows the comparison between YOLOv2, YOLOv2 Model A, and YOLOv2PD network architecture.

Table 3.2 YOLOv2, YOLOv2 Model A, and YOLOv2PD Network Architectures.

Layer No.	YOLOv2	YOLOv2 Model A	YOLOv2PD
L0	Conv_3*3_416*416*32	Conv_3*3_416*416*32	Conv_3*3_416*416*32
L1	Maxpool/2	Maxpool/2	Maxpool/2
L2	Conv_3*3_208*208*64	Conv_3*3_208*208*64	Conv_3*3_208*208*64
L3	Maxpool/2	Maxpool/2	Maxpool/2
L4	Conv_3*3_104*104*128	Conv_3*3_104*104*128	Conv_3*3_104*104*128
L5	Conv_1*1_104*104*64	Conv_1*1_104*104*64	Conv_1*1_104*104*64
L6	Conv_3*3_104*104*128	Conv_3*3_104*104*128	Conv_3*3_104*104*128
L7	Maxpool/2	Maxpool/2	Maxpool/2
L8	Conv_3*3_52*52*256	Conv_3*3_52*52*256	Conv_3*3_52*52*256
L9	Conv_1*1_52*52*128	Conv_1*1_52*52*128	Conv_1*1_52*52*128
L10	Conv_3*3_52*52*256	Conv_3*3_52*52*256	Conv_3*3_52*52*256
L11	Maxpool/2	Maxpool/2	Maxpool/2
L12	Conv_3*3_26*26*512	Conv_3*3_26*26*512	Conv_3*3_26*26*512
L13	Conv_1*1_26*26*256	Conv_1*1_26*26*256	Conv_1*1_26*26*256
L14	Conv_3*3_26*26*512	Conv_3*3_26*26*512	Conv_3*3_26*26*512
L15	Conv_1*1_26*26*256	Conv_1*1_26*26*256	Conv_1*1_26*26*256

L16	Conv_3*3_26*26*512	Conv_3*3_26*26*512	Conv_3*3_26*26*512
L17	Maxpool/2	Maxpool/2	Maxpool/2
L18	Conv_3*3_13*13*1024	Conv_3*3_13*13*1024	Conv_3*3_13*13*1024
L19	Conv_1*1_13*13*512	Conv_1*1_13*13*512	Conv_1*1_13*13*512
L20	Conv_3*3_13*13*1024	Conv_3*3_13*13*1024	Conv_3*3_13*13*1024
L21	Conv_1*1_13*13*512	Conv_1*1_13*13*512	Conv_1*1_13*13*512
L22	**Conv_3*3_13*13*1024**	**Conv_3*3_13*13*1024**	**Conv_3*3_13*13*1024**
L23	**Conv_3*3_13*13*1024**	Route-L16	**Conv_3*3_13*13*1024**
L24	**Conv_3*3_13*13*1024**	Conv_3*3_13*13*512	Route-L6
L25	Route-L16	Conv_1*1_13*13*64	Conv_3*3_13*13*128
L26	Conv_1*1*_13*13*64	Reorg	Conv_1*1_13*13*32
L27	Reorg	Route-L26 L22	Reorg
L28	Route-L27 L24	Conv_3*3_13*13*1024	Route-L10
L29	Conv_3*3_13*13*1024	Conv_1*1_13*13*30	Conv_3*3_13*13*256
L30	Conv_1*1_13*13*30	Detection	Conv_1*1_13*13*64
L31	Detection		Reorg
L32			Route-L16
L33			Conv_3*3_13*13*512
L34			Conv_1*1_13*13*64
L35			Reorg
L36			Route-L35 L31 L27 L23
L37			Conv_3*3_13*13*1024
L38			Conv_1*1_13*13*30
			Detection

3.4 Experimental Discussions

To showcase the power of YOLOv2PD model performance, it was evaluated on multiple datasets and the proposed model was compared with several state-of-the-art pedestrian detection models. The effectiveness of the proposed model was verified both quantitatively and qualitatively.

3.4.1. Datasets

(i) Pascal Voc-2007+2012 Pedestrian: This dataset contains 20 object classes and around 17,125 labeled images; it is a complete dataset generally used for object detection and classification. An unsupervised learning method (K-means clustering) is applied during training. Since manual annotation of a dataset is a complex and huge project, around 10,080 pedestrian and non-pedestrian images (referred to as the Pascal Voc-2007+2012 Pedestrian dataset) were extracted from Pascal dataset [100].

(ii) INRIA: The INRIA Pedestrian dataset [102] contains 1826 pedestrians, with image resolution 64 × 128. The pedestrian images captured in this dataset possess a complex background, illumination changes, various degrees of occlusion, variations in human posture, and individuals wearing different clothes.

(iii) Caltech Pedestrian: The Caltech pedestrian dataset [103] contains a set of video sequences of 640 × 480 in size captured from an urban environment. It includes training (set 00 to set 05) subsets and testing (set 06 to set 10) subsets. It contains 250k video frames, 350k bounding boxes and 2.3k pedestrians (“person” or “people” labels) are annotated. The training dataset is formed by extracting every image after every 30 frames from set 00 to set 05 and testing images are extracted from set 06 to set 10.

(iv) City Persons: This dataset [104] contains 2975 training set, 500 validation images and 1525 testing images. This dataset contains a large number of images under a variety of conditions. The image resolution used for training is 1024 × 2048. This model is trained on 3475 training images and performance is evaluated on 1525 testing images.

The summary of datasets used in the YOLOv2PD algorithm is shown in below Table.3.3

Table 3.3 Summary of datasets used for experimentation of YOLOv2PD network.

Datasets	Training Images	Testing Images
Pascal Voc-2007+2012 Pedestrian	9072	1008
INRIA	614	228
Caltech Pedestrian	4250	4024
City Persons	3475	1525

3.4.2. Training and Evaluation of Performance Parameters

The experiments were carried out on a workstation during the training phase; the testing phase was also performed on the same workstation. DarkNet was chosen as a feature extractor for all of the models, which was trained on a huge ImageNet dataset. The experimental setup of the workstation was: Windows 10 pro OS, Intel Xeon 64-bit CPU @3.60 GHz, 64 GB RAM, Nvidia Quadro P4000 GPU, CUDA 10.0 & CUDNN 7.4 GPU acceleration library and Tensorflow 1.x deep learning framework.

The model training was carried out on Pascal Voc-2007+2012 Pedestrian dataset (9072) training images and tested on 1008 testing images, since we are only concerned with pedestrian images. The input image size was resized to 416 × 416 resolution and various data augmentation techniques were applied such as colour shifting, flipping, cropping, and random sampling, in order to enhance the training process.

All the three models were trained for 100 epochs, with an initial learning rate of 0.001, and later learning rate is divided by 10 at 60 and 80 epochs respectively. During the model training, it randomly selects a new input image of different resolution after every 20 epochs. Since multi-scale training strategy improves model robustness, it can perform better prediction on images with different resolutions. While training Caltech dataset, the original images are up-sampled to 1024×1024 pixels, one mini-batch contains 16 images, learning rate is 10^{-4} and the model training is stopped after 85 epochs. Initially the learning rate was set to 0.005 for the first 50 epochs and 0.0005 for the rest 35 epochs. For City Persons dataset, we set a mini-batch that contains 2 images, and training was stopped after 85 epochs.

Average precision (AP) and inference speed (FPS-Frames per second) are standard techniques preferred to evaluate model performance. Intersection over union (IoU) is a good performance metric used to measure the accuracy of the designed model on a test dataset. IoU is simply computed as the area of intersection divided by the area of union. IoU helps to determine whether a predicted bounding box is a True Positive (TP), False positive (FP) or False Negative (FN) by defining a threshold of ≥ 0.5.

(i) Recall: A measure of how good the model is at finding all of the positives.

(ii) Precision: A measure of the accuracy of our predictions. These two terms are inversely proportional to each other.

$$\text{Re call} = \frac{\text{True Negative}}{\text{Predicted Results}} \text{ Or } \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{3.3}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \text{ Or } \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{3.4}$$

(iii) Average Precision (AP): This is the area under the precision- recall curve, which shows the correlation between precision and recall at different confidence scores. A higher AP value indicates better detection accuracy.

The performance of the model while validating INRIA, Caltech and City Persons test datasets was visualized using a plot between the number of false positives per image and the Miss Rate (MR).

(iv) Miss-Rate (MR): The ratio between the number of FNs and the total number of positive samples (N) is referred to as the MR.

Miss Rate (MR) = FN/N (3.5)

There is another relationship between the miss rate and recall expressed as:

Recall = 1 – Miss Rate (MR) (3.6)

3.5 Results & Discussions

The analysis of the training stage of all three models is depicted in Figure. 3.3. Average loss is indicated on y-axis, while x-axis indicates the number of iterations performed in training. It is clear from Figure.3.3 that the average loss curve is not stable up to approximately 10000 iterations. Compared with all other models, the average loss curve of YOLOv2PD model decreases faster initially, followed by that of YOLOv2 Model A. The reason for this is that both YOLOv2PD and YOLOv2 Model A adopted a multi-layered feature fusion strategy, so they obtained more local features, which accelerated the training convergence. During the training stage, initially YOLOv2PD model first reached a minimum average loss value (overall lowest value = 0.54), followed by YOLOv2 Model A and YOLOv2 models. Therefore, the YOLOv2PD model is more suitable for detecting small pedestrians on Pascal Voc-2007+2012 pedestrian dataset.

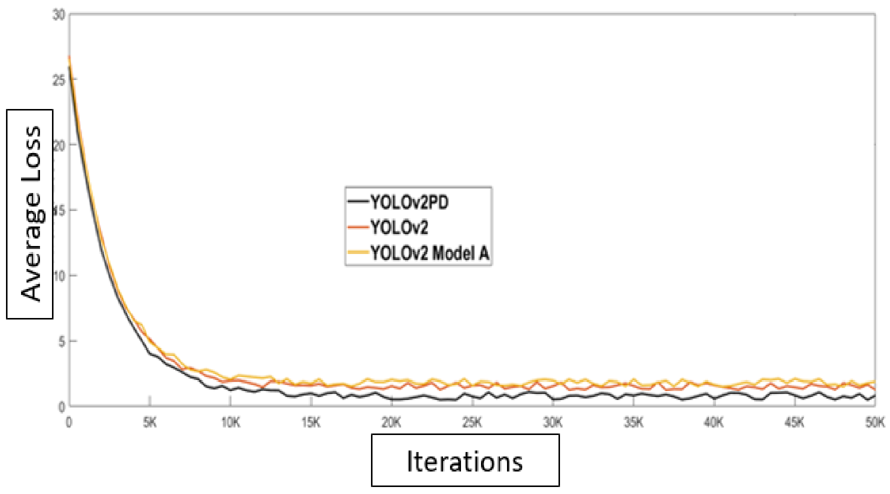


Figure 3.3 Analysis of training stage of all three models.

Figure. 3.4 shows the precision vs. recall (PR) curve obtained on the Pascal Voc-2007+2012 pedestrian dataset of all three models. The graph shows that, with increasing recall value at the convergence point, the precision gradually starts decreasing.

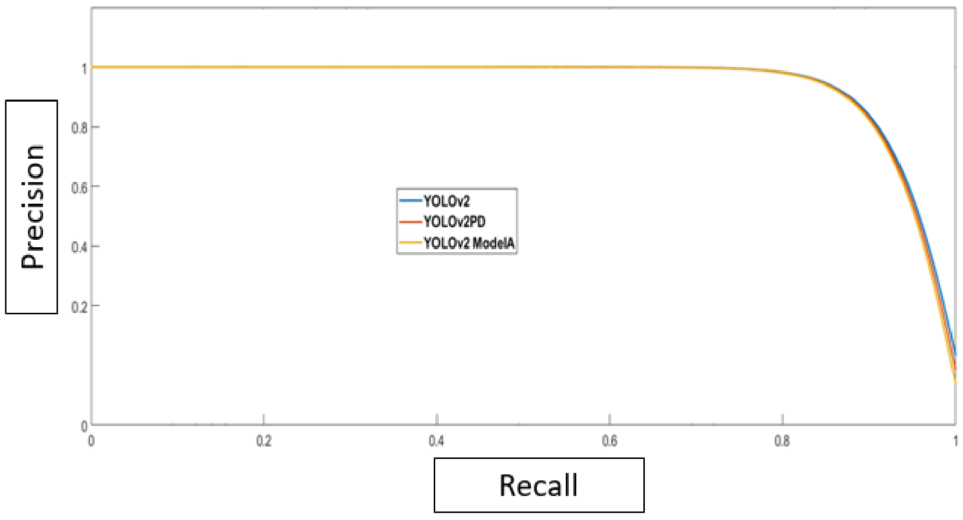


Figure 3.4 PR curves of all three models on Pascal Voc-2007+2012 pedestrian dataset.

3.5.1 Quantitative Analysis

To further validate the proposed method's effectiveness, the experimental results were examined using several quantitative performance metrics. With different input image resolutions of 416 × 416, 544 × 544, and 608 × 608, YOLOv2PD achieves comparable detection performance when compared with YOLOv2 Model A and YOLOv2. Table.3.4 compares the detection performance of all models for different image resolutions with respect to AP and inference speed (FPS) parameters. The proposed network YOLOv2PD achieves AP, that is, detection performance of 79.5, 80.7 and 82.3 respectively. From these results, it is clear that, as the applied input image resolution increases, the AP value increases but at the same time inference speed decreases.

To have a model that runs at higher inference speed, an image size of 416 × 416 is the best choice. As the input image size increases, inference speed decreases since these terms are directly proportional to each other. However, we are concerned with detecting smaller and densely distributed pedestrians, so 416 × 416 images are not quite suitable as they miss the detection of many smaller objects. Therefore, we consider selecting a 544 × 544 image size for detecting smaller and densely distributed pedestrians.

Table 3.4 Evaluation results of all three models on pedestrian test dataset (IoU@0.5).

Input Size	Model	Average Precision (AP)	Inference Speed (FPS)
416 × 416	YOLOv2	75.2	45.1
	YOLOv2 Model A	77.1	54
	YOLOv2PD	79.5	47.2
544 × 544	YOLOv2	76.5	32
	YOLOv2 Model A	78.3	38.2
	YOLOv2PD	**80.7**	**36.3**
608 × 608	YOLOv2	78.2	26.1
	YOLOv2 Model A	80.4	32.1
	YOLOv2PD	82.3	30.6

From the experimental results, our proposed algorithm runs at 36.3 FPS in real time on 544 × 544 image resolution. In this study, if AP is considered, then an image size of 544 × 544 would be the best choice as the proposed model achieves 80.7% detection accuracy, which is 2.1% higher than that of YOLOv2 [16]. The proposed model runs at 30.6 FPS for 608 × 608 image resolution, but the inference speed falls by 5.7 FPS compared to 544 × 544 image resolution.

The evaluation results of all three models on INRIA test dataset are expressed in terms of average precision and inference speed (milliseconds). Table.3.5 shows the detected results on INRIA test dataset for different image resolutions. At 544 × 544 test image resolution, the proposed model achieves 91.2% AP, which constitutes an improvement by 6.6% and 11.4% compared with YOLOv2 Model A and YOLOv2 models, respectively. This is because the proposed model uses MLFF strategy while detecting smaller pedestrians.

Table 3.5 Detection results of all three models on INRIA Test dataset. (IoU@0.5)

Input Size	Model	Average Precision (AP)	Inference Speed (FPS)
544 × 544	YOLOv2	79.8	32
	YOLOv2 Model A	84.6	38.2
	YOLOv2PD	**91.2**	**36.3**

608 × 608	YOLOv2	82.5	26.1
	YOLOv2 Model A	87.1	32.1
	YOLOv2PD	93.4	30.6

To test the robustness of the proposed model, testing was performed on INRIA pedestrian dataset and YOLOv2PD model performance was compared with several SOTA algorithms.

Table.3.6 shows a comparison of YOLOv2PD model performance with advanced existing algorithms evaluated in terms of average MR and runtime (FPS) on a reasonable test dataset. Our model achieves better detection performance than YOLOv2 [15], Spatial Pooling [86] and Y-PD [62] and improved by 4.7%, 3.4% and 1.3% respectively, but lags behind F-DNN [63] and YOLOv3 [16] by 1% and 0.6% respectively. Obviously, on INRIA pedestrian test dataset, the proposed model achieves a better trade-off balance between speed and accuracy when detecting pedestrians.

Table 3.6 Comparison of YOLOv2PD results with recent SOTA methods on INRIA.

Models/Avg.MR (%)	Reasonable	Runtime (FPS)
VJ [22]	72.5	<1
HOG [7]	46	<1
YOLOv2 [15]	12.5	32
Very Fast [44]	16	**>100**
Spatial Pooling [86]	11.2	<1
RPN + BF [47]	6.9	~4
Y-PD [62]	9.1	73
F-DNN [63]	**6.8**	~6
YOLOv3 [16]	7.2	20
Proposed	7.8	36.3

Table.3.7 shows a comparison of the proposed model performance with the advanced existing algorithms on the Caltech test dataset, evaluated in terms of MR, average precision, and detection speed.

Table 3.7 Comparison of YOLOv2PD detection results with recent SOTA methods on Caltech dataset (IoU@0.75).

Models/LAMR (%)	Reasonable	Average Precision (AP)	Runtime (sec)
RPN + BF [47]	9.58	0.324	0.5
SA-FastRCNN [49]	9.68	0.344	0.59
UDN + SS [50]	11.52	0.331	0.28
M-GAN [51]	**6.83**	-	-
Faster RCNN + ATT-Vbb [52]	10.33	-	-
TTL(MRF) + LSTM [53]	7.4	-	-
SSNet [57]	8.92	0.36	0.43
SDS-RCNN [60]	7.36	0.355	**0.21**
CompactACT + Deep [61]	11.75	0.334	1
Y-PD [62]	18.4	0.321	-
Proposed	7.48	**0.381**	0.29

From Table.3.7, it is clear that, on Caltech test dataset, the proposed model has better detection performance than RPN + BF [47], SA-FastRCNN [49], UDN + SS [50], Faster RCNN + ATT-Vbb [52], SSNet [57], CompactACT + Deep [61], and Y-PD [62] models on the reasonable subset [h ϵ (50, ∞)]. However, the proposed model, average miss rate falls behind those of M-GAN [51], TTL (MRF) + LSTM [52] and SDS-RCNN [60] models by 0.65%, 0.80% and 0.12% respectively.

The proposed model performance assessment is also done on City Persons pedestrian dataset [104]. Table.3.8 shows the experimental results of the proposed model and the results were compared with several state-of-the-art approaches. The proposed model was evaluated on reasonable subset only and considered IoU @0.5.

Table.3.8 Comparison of the metrics parameters and testing time of several SOTA relevant approaches with that of the City Persons dataset (Reasonable subset IoU@0.5).

Models/Avg.MR (%)	Reasonable	Testing time (sec/img)
RPN + BF [47]	15.5	0.5
DDFE [91]	12.9	0.22
YOLOv2 [15]	14.5	0.32
SA-FastRCNN [49]	12.7	0.59
TTL(MRF) + LSTM [53]	14.4	0.25
F-DNN [63]	13.6	**0.16**
YOLOv3 [16]	**12.1**	0.38
Proposed	13.8	0.29

Table.3.8 shows the experimental results of the proposed method comparison with several SOTA approaches, namely, RPN+BF [47], DDFE [91], YOLOv2 [15], SA-FastRCNN [49], TTL (MRF) + LSTM [53], FDNN [63] and YOLOv3 [16]. On Reasonable scenario, the proposed model achieves a comparative log-average Miss rate when compared to other SOTA detectors. However, this model fails to achieve low log-average miss rate and falls behind YOLOv3 [16] model.

3.5.2 Qualitative Analysis on multiple datasets and small-scale Pedestrian Detection

Pascal Voc-2007+2012 pedestrian dataset contains 20 different classes and every class may have small objects. Since our main aim is detecting smaller and densely distributed pedestrians in this dataset, a dataset was made manually by collecting 330 images that mainly include smaller and dense pedestrians to evaluate the proposed model's performance. Figure. 3.5 shows qualitative detected results of small-scale pedestrian detection on all three models and these were compared with YOLOv3 [47] SOTA detector. From these detection results, it is evident that the proposed model can produce better prediction on smaller and densely distributed pedestrians than other models.

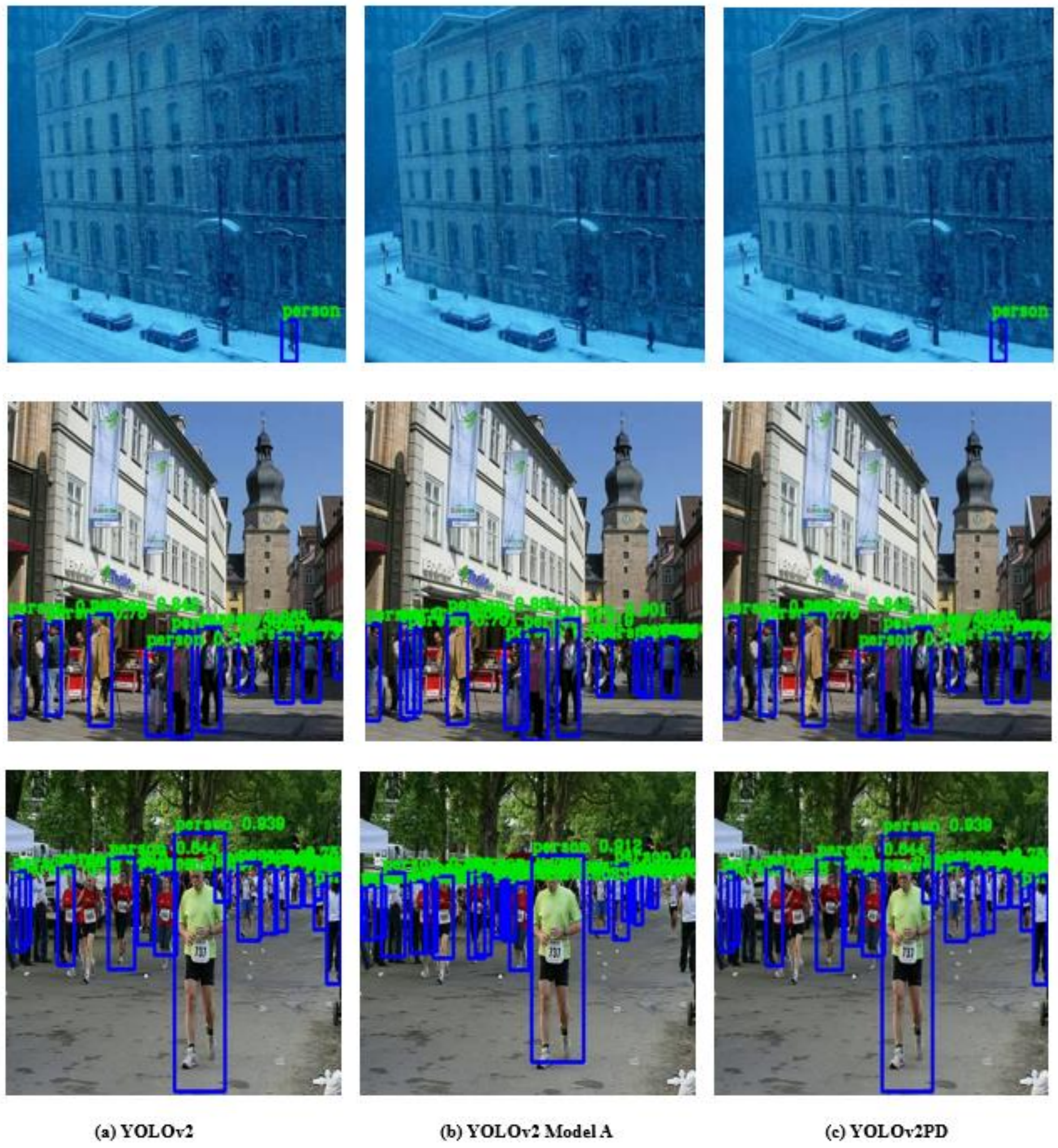


Figure 3.5 Detection results of YOLOv2, YOLOv2 Model A and YOLOv2PD Models (544×544).

Figure 3.6 shows qualitative detection results on both Pascal 2007+2012 pedestrian and INRIA test datasets. To show the findings more intuitively, regarding real-time performance of the proposed algorithm and to achieve a perfect balance between detection speed and accuracy, a real-time test video is fed to all three models. The detection results of the randomly selected 79th frame on all the models is shown in Figure.3.7.

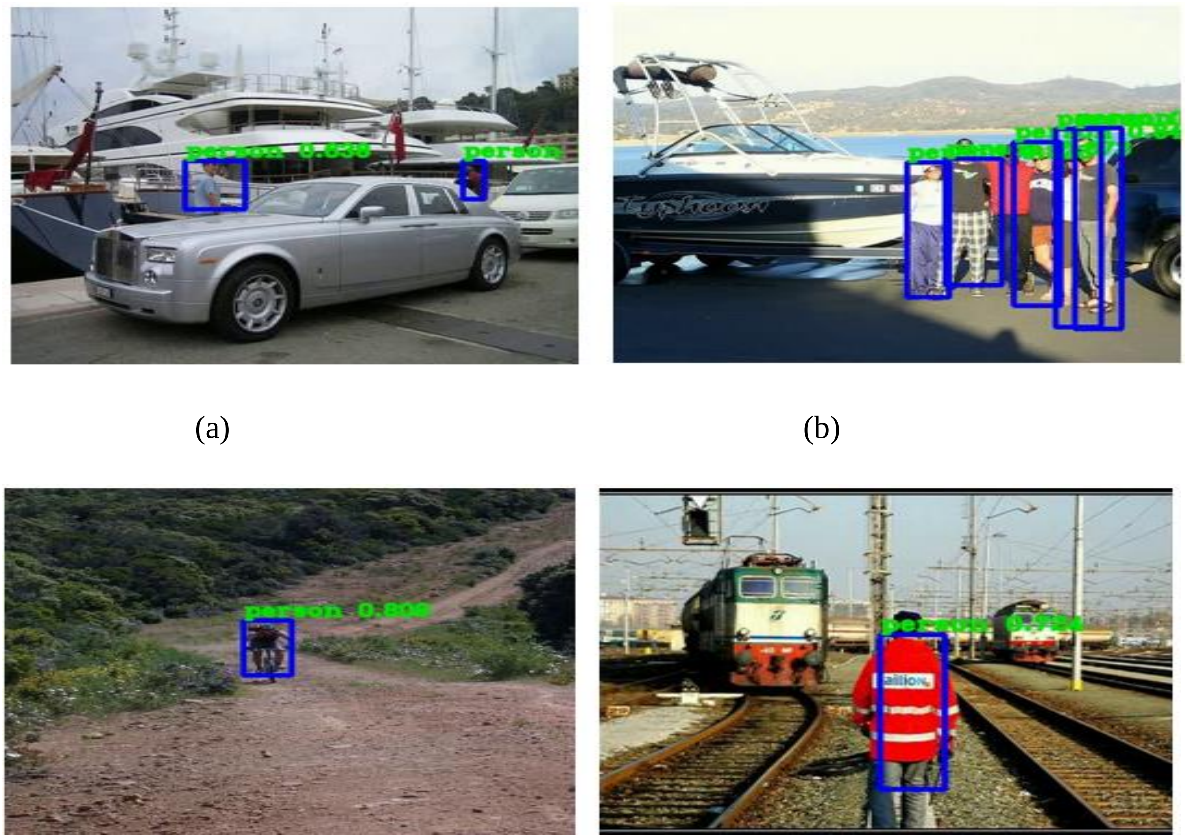


Figure 3.6 Sample detected results of YOLOv2PD on Pascal 2007+2012 Pedestrian and INRIA test datasets (544×544).

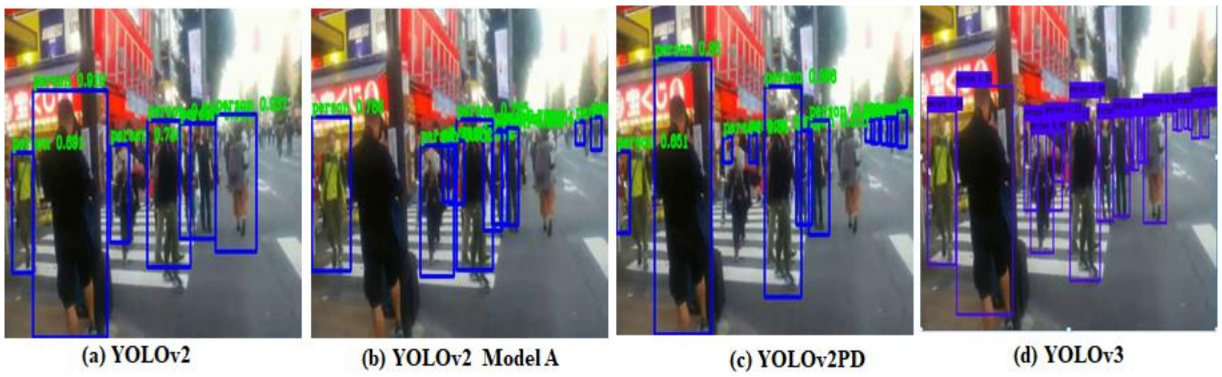


Figure 3.7 Real-time detection results of YOLOv2, YOLOv2 Model A, YOLOv2PD and YOLOv3 Models (544×544) on test video.

The running time of all the three models on a real-time input test video is evaluated. The detection speed on an input image of size 544 × 544 was 32 FPS for YOLOv2, 38.2 FPS for YOLOv2 Model A, 36.3 FPS for YOLOv2PD and 20 FPS for YOLOv3.

3.6 Summary

This chapter presented an accurate pedestrian detection algorithm while detecting smaller and densely distributed pedestrians. In this work three new contributions were presented. Our first contribution involved designing two pedestrian detection models YOLOv2PD and YOLOv2 Model A and prove the effectiveness of YOLOv2PD model through experimental results. The second contribution was introduction of Multi-layer Feature Fusion (MLFF) strategy in the proposed model to improve the model's feature extraction ability, and reduced network complexity by removing one convolution layer from the higher end. And the last contribution is that loss function is improved by applying normalization on the bounding box loss error. However, the proposed model runs in real time, there is still room for improvement of the speed, miss rate on INRIA, City Persons datasets and miss detection of small similar and occluded pedestrians.

Chapter 4

SSAD: Single Shot Multi-scale Attentive Detector

This chapter's main objective is to discuss the development of an efficient object detection algorithm and to achieve optimal trade-off balance between detection accuracy and speed while detecting small-scale and occluded objects. In this work, Single shot Multi-scale Detector (SSD) was adopted as baseline network and two modules named as feature fusion and Multi-scale Attention Unit (MAU) were introduced and the network is referred to as "Single shot Multi-scale Attentive Detector" (SSAD). Both qualitative and quantitative analysis of the proposed model were verified on multiple autonomous driving datasets and a comparative analysis was made with existing state of the art (SOTA) object detection algorithms.

4.1. Introduction

In order to improve safety driving and reduce driver fatigue, there is a need to develop intelligent driving technology [92]. The first task is to guarantee driver's safety in intelligent driving system, which is possible by assisted driving system (ADS) [93]. So, improving driver's safety is a key component of research in intelligent driving system. For assisted driving system in smart cars, a collision avoidance warning system (CAWS) [94] is required. The captured images of pedestrians and vehicles by car cameras are to be identified, classified, localized and detected by detection technology, which faces challenges due to complex scene information. Driver assistance systems not only require an extremely high accuracy of object detection but also cannot miss small targets that are difficult to detect in complex scenes. There are wide applications of detecting pedestrians and vehicles for autonomous driving, covers areas such as smart robots, smart intelligence video surveillance, smart cars, assisted smart driving system and intelligent transportation system (ITS).

Commonly object detection methods are classified into traditional learning methods and deep learning methods. Traditional machine learning based detection methods primarily focus only on manual feature extraction. The following sequence of steps such as post-processing, feature extraction, classifier learning and detection were carried out by traditional detection methods; however, these methods run with higher detection speed. Under specific conditions, these methods achieve good detection results, while detection accuracy is greatly sacrificed under poor lighting or occlusion situations. Therefore, traditional learning detection methods are not suitable practically and don't satisfy real-time requirements.

After 2012, deep learning-based detection methods use Convolutional Neural Networks (CNN's) for detecting pedestrians and vehicles. The features are extracted by traversing through the entire image, classified by a classifier and finally non-maximum suppression suppresses the output detected results. CNN based pedestrian and vehicle detection methods have achieved great success in terms of speed and detection accuracy and have become current state-of-the art mainstream methods. Recently, there has been a tremendous growth and rapid development using deep learning-based pedestrian and vehicle detection methods [47],[91], [95-97] however, some problems remain. One of the major problems is to maintain trade-off balance between detection accuracy and speed. In addition, research in autonomous driving system is in full swing, and the available detection technologies should accurately and quickly detect both pedestrians and vehicles. In order to improve both detection accuracy and efficiency, a multi-scale attention unit (MAU) module is introduced, in the proposed model.

Our main goal is to improve, one-stage SSD based detector for autonomous driving system from a different perspective. The proposed model focus only on the crucial regions of the feature map from which intrinsic feature relations are extracted and which are helpful for detecting occluded pedestrian and vehicles. This key idea came from the human vision (HV) system. At first glance, while perceiving a scene, we humans figure out the contents instantly through global dependency analysis. When our eyeball focus on a particular point, the intensity of the surrounding regions decreases. A MAU module is designed which is similar to human vision system. MAU has the capability to analyse the importance of features at different locations by using global feature relations. This MAU generates an attention map matrix which highlights only useful regions and suppresses unnecessary regions.

Driver assistance systems not only require an extremely high accuracy of object detection but also cannot miss small targets that are difficult to detect in complex scenes. So one of the major problem is in-balance of detection accuracy and speed and their trade-off. Therefore, in this chapter, a novel Single shot Multi-scale detection network was proposed to improve both detection accuracy and efficiency. To achieve optimal trade-off balance between detection accuracy and speed, SSD was adopted as baseline detector in the proposed model. A combination of SSD + Feature Fusion + Multi-scale Attention Unit (MAU) is referred to as 'Single shot Multi-scale Attentive Detector' (SSAD). The merits of the original SSD [19] structure are preserved in SSAD model and which is more robust and effective while learning object features.

The contributions of this chapter are summarized as follows:

- The proposed one-stage detector incorporates pixel-wise feature relations and follows the human vision mechanism. The SSAD model learns to highlight pedestrian and vehicle regions on the extracted feature map and also suppress irrelevant regions, from the global relation information and thereby provide reliable guidance for autonomous driving.
- SSAD model is more accurate and preserves the simplicity and efficiency of SSD.
- To verify the robustness of the SSAD model, it was validated on four challenging datasets for autonomous driving scenario. Evaluation results on multiple datasets show that SSAD model competes favourably with SOTA one-stage SSD based series detectors in terms of efficiency and detection accuracy.

4.2. Overview of Single Shot Detector (SSD) Series

SSD is referred as regression-based object detector. This model can solve the conflict caused by translational invariance and variability and can achieve better detection accuracy as well as speed. The structure of SSD + VGG backbone network for an input image 300x300 size is shown in Figure 4.1. Each selected feature map consists of K frames which varies in size and width to height ratio. Each frame is also referred to as anchor box. Figure 4.1 shows bounding boxes on feature maps of different convolution layers. B class score and 4 position parameters are predicted by default at every bounding box. B x K x w x h class score and 4 x K x w x h position parameters need to be detected for the w-h feature image. It requires a size 3 x 3 (B + 4) x K number of w x h convolution kernel for the processing of the feature map. The convolution result is considered to be the last feature for bounding box and classification regression. The sum of bounding box regression's position loss and classification regression's confidence loss is referred to as total loss function.

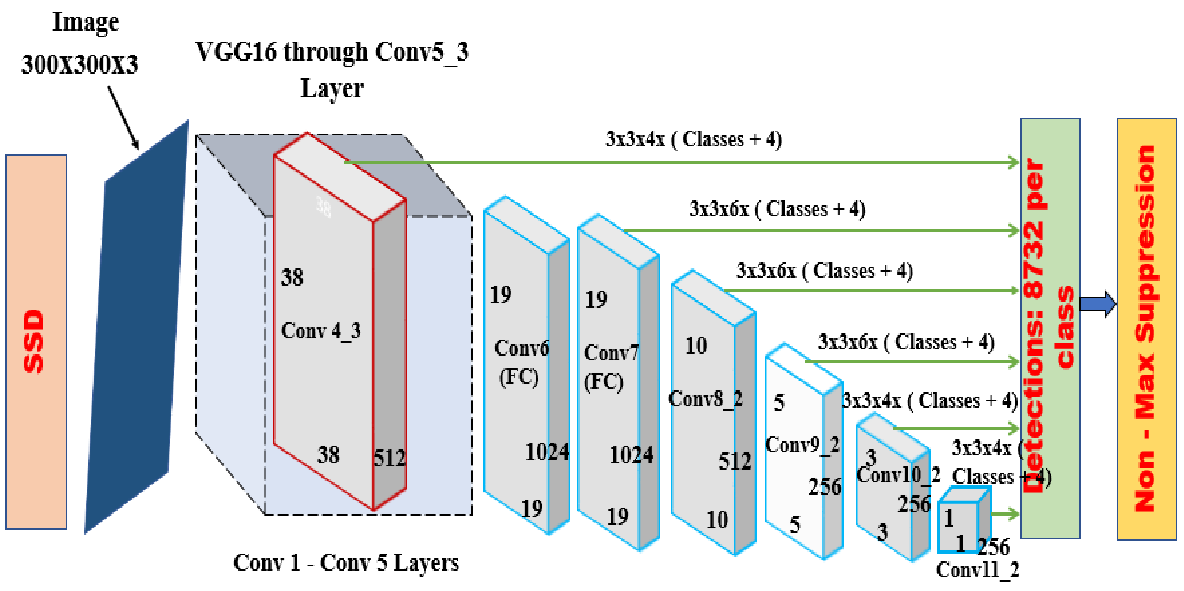
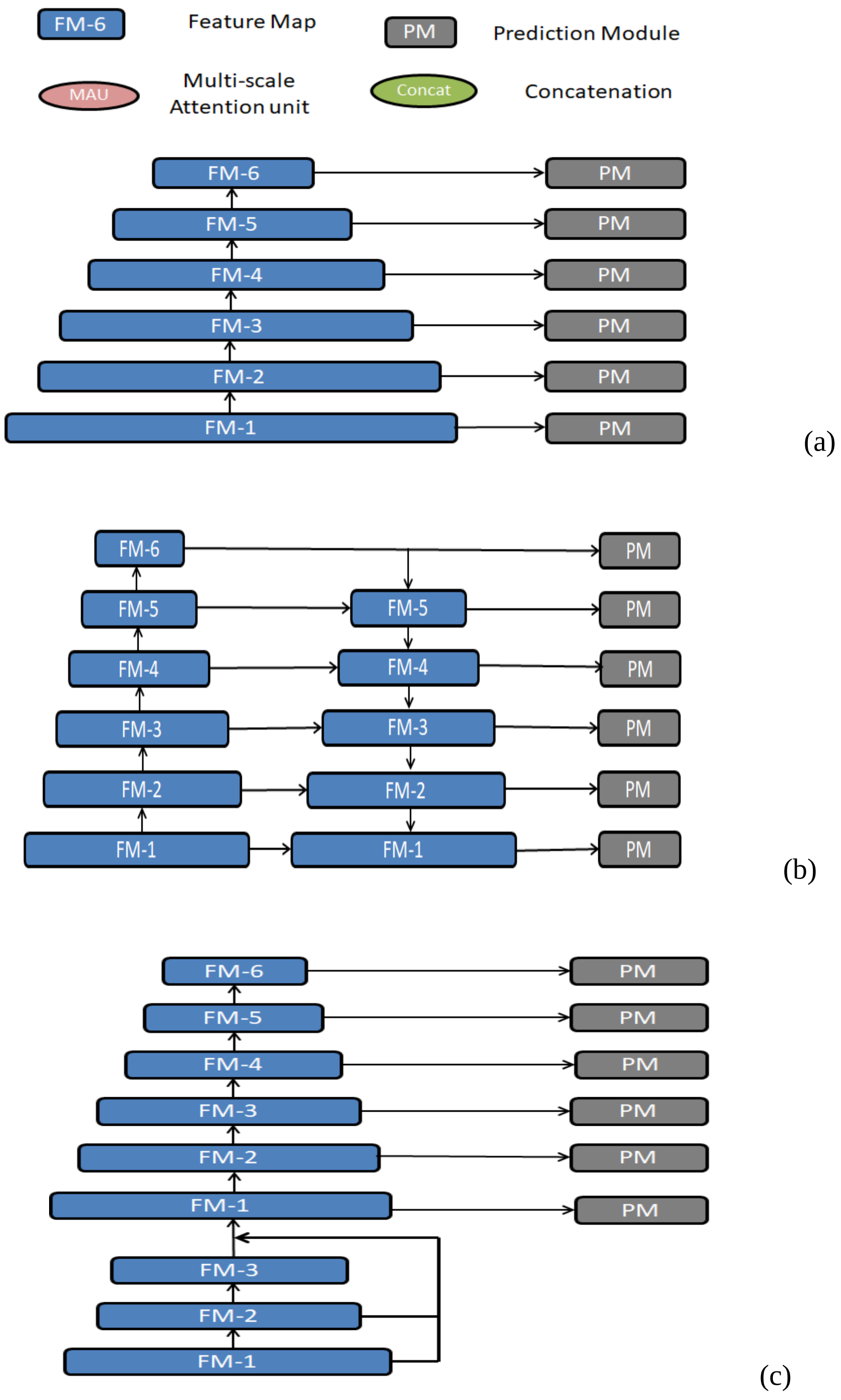
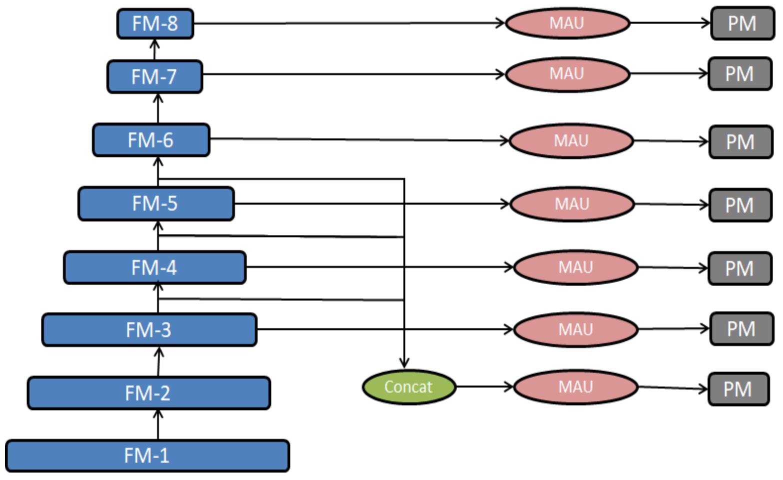


Figure 4.1 SSD300 with VGG-16 backbone network through Conv5_3 layer.

Figure 4.2 shows that the designed SSAD model is simple and more effective in refining the contextual semantics when compared to state-of-the-art (SOTA) one-stage SSD-based detectors. Fu et al. [21] proposed Deconvolutional SSD (DSSD), with a highly complex feature pyramid network (FPN) [98] and the information flows through various convolutional layers.

Compared to SSD [19], DSSD achieved higher detection accuracy but was computationally expensive and relatively more complex. Li et al. [20] proposed Feature-fused Single-shot Detector (FSSD) for multi-scale feature aggregation; it is a combination of SSD + Feature Fusion modules, and achieves only a marginal improvement in the detection but by scarifying the speed when compared to SSD [19]. Whereas proposed SSAD model refines the various targets information from each layer by employing a single efficient MAU and also retains the original SSD structure (see Figure 4.2(d)).


FM-6
Feature Map
PM
Prediction Module
MAU
Multi-scale Attention unit
Concat
Concatenation
FM-6
FM-5
FM-4
FM-3
FM-2
FM-1
PM
(a)
FM-6
FM-5
FM-4
FM-3
FM-2
FM-1
PM
(b)
FM-6
FM-5
FM-4
FM-3
FM-2
FM-1
PM
(c)



(d)

Figure 4.2 Various structures of one stage SSD-based detectors (a) SSD (b) DSSD (c) FSSD and (d) SSAD (Proposed).

4.2.1 Visual Attention Mechanism

In order to exploit the salient visual information and facilitate visual tasks in various object recognition applications, visual attention (VA) mechanism was applied. Saliency-based VA model [110] selects only specific locations from the extracted saliency maps. Recurrent Neural Network (RNN) is employed in RAM [111] model, and to discover specific targets reinforcement learning was applied. To classify real objects through CNN, classification was performed by Attention Net [112] and RA-CNN [113] models. The above discussed models focus only on single instance problems. To discover a global contextual guidance in multi-object recognition applications, AC-CNN [117] and Relation Net [114] were proposed. AC-CNN [115] uses stacked Long Short-Term Memory (LSTM) units to examine the global context. Practically Relation Net [114] is a two-stage detector, would generate more attentive features, since it correlates the geometrical features and information appearance between the region proposals and performs slightly better than AC-CNN [117]. For detecting pedestrians a dual attention network was proposed in CSANet [116]. This network models the attention mechanism of feature maps from both the channel and spatial dimensions. MSCMANet [65] model performs multi-scale feature extraction and uses an attention module on the extracted feature maps from both channel and spatial dimensions.

4.2.2 Self-attention

In the field of Natural Language Processing (NLP), to model long-range dependencies of a sentence, a self-attention mechanism is a widely adopted technique. An attention memory network was constructed in LSTMN [115] which discovers the relations between tokens and also improves memorization capability of LSTM network. To generate a two-dimensional embedded matrix, structured self-attentive [118] applies self-attention mechanism in bidirectional LSTM on each row part of the sentence. This technique also provides global dependencies between the input and output of a transformer [119] model. So in the proposed work, transformer [119] concept was adopted to develop long-range dependencies from the extracted feature maps itself. Therefore, our model was highly capable of focusing on different regions while detecting multi-scale pedestrians and vehicles effectively and accurately.

4.3 Proposed Methodology

To handle and detect multi-scale objects effectively, SSD [19] performs detection on the extracted multi-scale feature maps. Yet, SSD fails to detect smaller objects, since shallow layers lack semantic information. The solutions to the above problem are: inject semantic information from deeper layers to the shallower layers exhaustively or construct a network with more CNN layers to obtain refined feature maps. One major concern is one-stage detectors, should run in real-time, and improve SSD detection accuracy but with an extra computational cost. Prior to the prediction module (PM), MAU module is embedded into SSD [19] in order to improve the detection accuracy. Figure 4.3 shows SSAD architecture and used ResNet101 (conv1–5) [79] used as backbone (see Table 4.1) network. Conv. (6 to 9) (i.e., pyramidal convolutional) blocks follow SSD [19]. Conv-3 to 9 feature maps detect multi-objects with different scales. In SSAD model, an MAU module is inserted between extracted feature map and prediction module (PM).

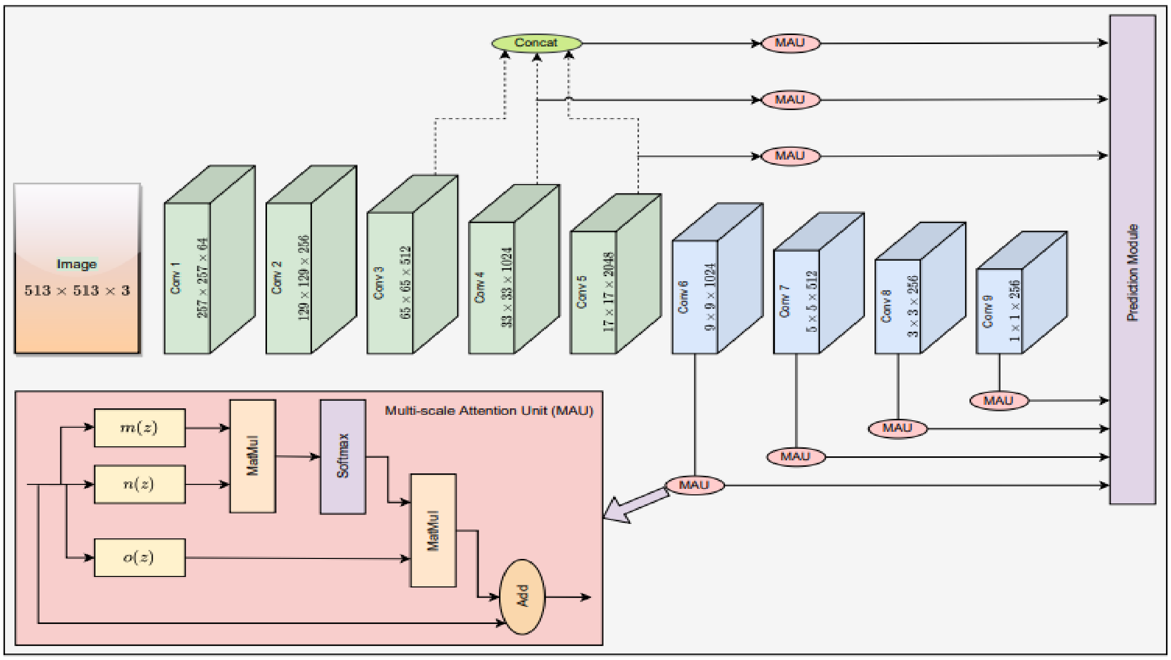


Figure.4.3 SSAD Architecture.

4.3.1 Multi-scale Attention Unit (MAU)

In the proposed model, Self-attention Mechanism (SAM) [24] was adapted for attention map visualization. By using an attention map, SAM produces global dependencies between the two sequences, which map a query and a set of key-value pairs to an output. The input features motivate the attention module in SAM and are much helpful to refine the features. So in order to construct global feature correlations among extracted pixels, we consider our problem as a similar query problem to estimate relevant information from the input feature maps.

Assume at a given scale s € {1, 2, …,S}, z^s € $\mathbb{R}^{Cs \text{ x } Ns}$ is the feature map, where C represents number of channel locations and N represents total spatial locations in the extracted feature map. Initially linear transformation is applied on the input feature map z^s and it is divided into three different feature spaces m, n, o i.e. $m(z^s) = w_m^T z^s$,

$$n\,(z^s) = w_n^T\, z^s \qquad \text{and}$$

$$o\,(z^s) = w_o^T\, z^s$$

where w_m^T, w_n^T, w_o^T € $\mathbb{R}^{Cs \text{ x } C'}$ with C' = C/8.

The final matrix is obtained after multiplying $m(z^s)$ and $n(z^s)$ matrices as shown in Figure 4.3, to generate an attention map score matrix A^s € $\mathbb{R}^{Ns \text{ x } Ns}$. By applying softmax operation, each row of the attention map score matrix is normalized:

$$\bar{A}_{ij}^{s} = \frac{e^{A_{ij}^{s}}}{\sum_{j}^{N^s} e^{A_{ij}^{s}}} \quad i, j = 1, 2 \ldots N^s$$

$$A^s = m\,(z^s)^T\, n\,(z^s) \tag{4.1}$$

Where, $\bar{A}_i^s$ is known as multi-scale attention map, which describes the pixel relations when querying the ith location of the feature map. To reduce the computational cost, the input feature map z^s is linearly transformed into $m(z^s)$ and $n(z^s)$. The resultant matrix after multiplying both $m(z^s)$ and $n(z^s)$ would compute the similarity features and then generate an N x N multi-scale attention map which reveals the feature global relations among extracted pixels.

Then matrix multiplication is performed between $o(z^s)$ and $\bar{A}^s$. So the refined feature map is determined efficiently at every possible location as the sum of the weighted individual features. Therefore, the resulting matrix multiplication is summed up to the input feature map z^s:

$$\bar{z}^s = z^s + (\bar{A}.\, O(Z^s)^T)^T \tag{4.2}$$

where $\bar{A}_i^s$ multi-scale attention map, which relates the long-range dependencies from the extracted feature maps from all locations. Therefore, it only highlights crucial regions of the feature map and guides the detector effectively with the modified information.

Table 4.1 Architecture of SSAD + ResNet-101backbone (Input image size is 513 x 513).

Layers	Output Size	Specifications
Conv 1	256x256	Conv_ 7x7x, s2, 64
Conv 2_x	128x128	{Conv_ 1x1, 64, conv_3x3, 128, conv_1x1, 256} x3
Conv 3_x	64x64	{Conv_ 1x1, 128, conv_3x3, 128, conv_1x1, 512} x4
Conv 4_x	32x32	{Conv_ 1x1, 256, conv_3x3, 256, conv_1x1, 1024} x23
Conv 5_x	8x8	{Conv_ 1x1, 512, conv_3x3, 512, conv_1x1, 2048} x3

4.3.2 Semantic feature fusion

FSSD [20], model fuses the contextual information from layer4 and layer5 into layer3 to enhance its semantics. From thorough experimentation, it is clear that feature fusion alone does not improve the detection accuracy much (refer Table.4.4). However it decreases the detection accuracy slightly, with extra computational cost, because all three layers possess different receptive fields and capabilities.

In this model, both concatenation and 1x1 conv. transformation were applied, which would neutralize the relative importance of these layers and important features in layer3 were lost. So to overcome this problem, an MAU is inserted after feature fusion operation, which gives considerable improvement (refer Table.4.4). So MAU uses semantics from deeper layers to discover crucial information which resides in original layer3. Inferior performance was observed by applying MAU alone, which was compared with a combination of feature fusion and attention mechanisms simultaneously. This clearly shows that these two operations are complementary to each other.

The semantic feature fusion process is expressed as:

$$z^3 = W^3 \text{ Concat } \{z^3, z^4, z^5\} + b^3 \qquad (4.3)$$

where $z^s \in \mathbb{R}^{Cs \times Ns}$ is the feature map at various layers, $W^3 \in \mathbb{R}^{C3 \times C'}$ and $b^3 \in \mathbb{R}^{C3}$. When concatenation is applied, three layers (from 3 to 5) are up sampled through bilinear interpolation which aligns their sizes with that of original layer 3.

4.4 Experimental Discussions

To showcase the power of SSAD model, it was tested and evaluated on multiple datasets and the performance was compared with existing state-of-the-art object detection models. The robustness of the proposed algorithm was verified both quantitatively and qualitatively.

4.4.1. Datasets

(i) Pascal Voc-2007+2012: The proposed model is trained on two mainstream datasets: PASCAL VOC 2007 and VOC 2012 [100]. The number of images in Pascal 2007 +12 has 9963 and 22,531 images respectively. The trained model is finally tested on PASCAL VOC-2007 test dataset particularly for three classes i.e. car, bus and pedestrian. SGD is the optimizer solution used during the model training.

(ii) INRIA: The INRIA Pedestrian dataset [102] contains 1826 pedestrians, with an image resolution 64 × 128. The pedestrian images captured in this dataset possess a complex background, illumination changes, various degrees of occlusion, variations in human posture, and with individuals wearing different clothes.

(iii) Caltech Pedestrian: The Caltech pedestrian dataset [103] contains a set of video sequences of 640 × 480 in size captured from an urban environment. It includes training (set 00 to set 05) subsets and testing (set 06 to set 10) subsets. It contains 250k video frames, 350k bounding boxes and 2.3k pedestrians ("person" or "people" labels) are annotated. The training dataset is formed by extracting every image after every 30 frames from set 00 to set 05 and testing images are extracted from set 06 to set 10.

(iv) City Persons: This dataset [104] contains 2975 training set, 500 validation images and 1525 testing images. This dataset contains a large number of images under a variety of conditions. The image resolution used for training is 1024 x 2048. This model is trained on 3475 training images and the performance is evaluated on 1525 images.

The summary of datasets used for experimentation of SSAD algorithm is shown in Table 4.2.

Table 4.2 Summary of datasets used in experimentation of SSAD Network.

Datasets	Training Images	Testing Images
Pascal Voc-2007+2012	32494	4098
INRIA	614	228
Caltech Pedestrian	4250	4024
City Persons	3475	1525

4.4.2. Training and Evaluation of Performance Parameters

The experiments were carried out on a workstation during the training phase; and the testing phase was also performed on the same workstation. The experimental setup of the workstation is as follows: Windows 10 pro OS, Intel Xeon 64-bit CPU @3.60 GHz, 64 GB RAM, Nvidia Quadro P4000 GPU, CUDA 10.0 & CUDNN 7.4 GPU acceleration library and Pytorch [120] deep learning framework.

The proposed SSAD network was evaluated on PASCAL-VOC 2007 test dataset specifically on three different classes: Car, Bus and Pedestrian. Stochastic Gradient Descent

(SGD) optimizer solution was adopted in the proposed model. Our proposed network is trained on NVIDIA Quadro P4000 GPU work station. SSAD321 network was trained for 160 epochs where the learning rate initially used is 0.001 and decreased by 1/10th after the 80th and 120th epochs respectively. SSAD513 network was trained for 110 epochs where the learning rate initially used was 0.01 and later on decreased by 1/10th after the 60th and 80th epochs respectively. The weights of ResNet-50 & 101 backbone model was pre-trained on ImageNet [4] dataset. Using the settings of SSD, DSSD and FSSD, SSAD model was trained and evaluated on PASCAL VOC-2007+12 dataset for two input resolutions 321x321 and 513x513.

While training SSAD model on Caltech dataset, one mini-batch would contain 8 images, and the learning rate was 10^{-4}, and the training was stopped after 100 epochs. During the SSAD model training on City Persons, 2975 training images and performance were evaluated on 500 validation images. Initially, the learning rate was set to 0.005 for the first 50k iterations and 0.0005 for the rest 35k iterations. For City Persons dataset, a mini-batch that contains 2 images, and training was stopped after 85k iterations.

The same anchor box mechanism was followed as in original SSD [19] structure. The various aspect ratio (a_r) {1, 2, 0.5} for anchor boxes on conv_3, 8 & 9 feature maps and {1, 2, 0.5, 3, .3} for anchor boxes on conv_4 to conv_7. Each anchor box has both minimum and maximum scale, where the s_{min} scale, is spaced regularly over the feature map layers and s_{max}, is the s_{min} of next layer. Anchor box normalization is calculated using:

$$w = s \,.\, \sqrt{a_r} \quad \text{and} \tag{4.4}$$

$$h = \frac{s}{\sqrt{a_r}} \text{, where} \tag{4.5}$$

$$s = \sqrt{s_{max}} \,.\, \sqrt{s_{min}} \tag{4.6}$$

for a_r =1, else s = s_{min}. To solve the class imbalance problem hard negative mining was applied during the network training. Adopted same loss function and data augmentation techniques as used in SSD [15].

The qualitative performance metrics such as Average Precision (AP), mean average precision (mAP), frames per second (FPS) or testing time (seconds/image) and parameters size (MB) were evaluated on the proposed model. Log-average miss rate of 2% is calculated for INRIA dataset.

4.5 Quantitative and Qualitative Analysis

In order to test the robustness of the proposed model, both quantitative and qualitative test results analysis were performed on four different challenging detection datasets: PASCAL VOC 2007 + 2012 [100], INRIA Pedestrian [102], Caltech pedestrian [103] and City Persons [104] for autonomous driving. Stochastic Gradient Descent (SGD) is the optimizer solution applied while training all the four datasets.

The proposed model was initially tested on PASCAL VOC-2007 test dataset, for detecting pedestrians and various types of vehicles. The primary goal of the proposed model was comparing speed and detection accuracy with one-stage state-of-the-art detectors. The evaluation metrics are Average Precision (AP) and Mean Average Precision (mAP) complying with the PASCAL challenge protocols. The performance parameters such as Average Precision (AP), Mean Average Precision (mAP), frames per second (FPS) and parameters size (MB) were calculated and compared with one-stage SOTA detectors. From Table 4.3 it is clear that the proposed model achieves large improvement in detection accuracy compared to YOLOv2PD, YOLOv3 one-stage SSD-based detectors while detecting pedestrians and vehicles.

Table 4.3 Comparison of speed and accuracy on Pascal Voc 2007 test dataset.

Models	Back bone	Bus	Car	Person	mAP	FPS	No.of Anchors	Test Image	Param. (MB)
Faster-RCNN [12]	ResNet 101	78.60	76.6	80.70	78.63	2.4	6000	1000*600	134.7M
Yolov2 [15]	Darknet 19	79.8	76.5	81.3	79.20	32	--	544*544	--
SSD300 [19]	VGG16	82.7	78.8	82.6	81.37	46	8732	300*300	56.8M
SSD321 [21]	Resnet101	81.4	75.6	81.5	79.50	11.2	17080	321*321	56.8M
RetinaNet300 [121]	ResNet 101	67.2	58.8	70	65.33	11.4	15354	300*300	55.7M
RetinaNet500 [121]	ResNet 101	73	71.2	78.9	74.37	7.1	35964	500*500	55.7M

SSD512 [19]	VGG16	85.1	84.3	85.5	84.97	19	24564	512*512	--
SSD513 [121]	ResNet 101	81.1	83.7	85.7	83.50	6.8	43688	513*513	57.5M
FSSD513 [20]	VGG16	86.6	86.9	80.4	84.63	**43**	24564	--	--
YOLOv2PD [122]	Darknet19	80.5	79.2	82.3	80.7	36.3	--	544*544	--
YOLOv3 [16]	Darknet53	85.1	84	82.1	83.73	30	--	544*544	--
SSAD 300	VGG16	82.6	78.2	83.7	81.50	11.8	8732	300*300	**29.4M**
SSAD 321	ResNet 101	81.7	77	82.1	80.27	27.4	10325	321*321	66.7M
SSAD 512	VGG16	85.3	84.7	86.5	85.50	3.4	24564	512*512	176.2M
SSAD 513	ResNet 101	85.5	84.8	86.5	**85.60**	16	25844	513*513	67.2M

4.5.1 Ablation study on PASCAL VOC 2007 test dataset

The proposed SSAD model was explored using various effects of MAU, semantic feature fusion in terms of both detection accuracy and speed. Here the evaluation results of four models, SSD513 + ResNet101 (only), SSD513 + Resnet101 + Feature Fusion, SSD513 + ResNet101+MAU, and SSD513 + ResNet101 + Feature Fusion + MAU on PASCAL-VOC 2007 test dataset were verified. From Table 4.4, it is observed that feature fusion module alone does not lead to much improvement in detection accuracy. On the other hand, it creates a little more computational overhead. Whereas, significant performance improvement was given by multi-scale attention unit alone. So combining MAU and feature fusion modules gave further boost in detection accuracy parameter. Therefore, finally MAU module has the ability to analyse contextual semantics at different levels and select crucial information for guiding a better and more accurate objects detection.

Table 4.4 Ablation study on Pascal Voc-2007 test dataset (test image 513x513 size).

Models	Method	Time (sec)	Mean Average Precision (mAP)
SSD + ResNet101	--	**0.1518**	79.86
SSD + ResNet101	Feature-Fusion	0.1576	79.42
SSD + ResNet101	Multi-scale attention	0.1592	82.17
SSD + ResNet101	Feature-Fusion + Multi-scale attention	0.1623	**85.6**

4.5.2 INRIA Pedestrian

The Quantitative test results analysis carried out on multiple autonomous driving datasets on SSAD model were discussed in detail. Table 4.5 shows the test results of INRIA person dataset of the proposed model and make a comparison with the state-of-the-art detection algorithms.

Table 4.5 Comparison of the Miss rate with several SOTA approaches on INRIA dataset.

Models	Miss rate (%)
LDCF [123]	14
Spatial Pooling [86]	11.6
MT-LDCF [124]	11
MCF [125]	9
NNNF [126]	10
YOLOv2PD [122]	7.8
Y-PD [62]	9.1
YOLOv3 [16]	7.2
YOLOv4 [17]	6.7
SSAD	**6.4**

For fair comparison of our model with several SOTA models were considered LDCF [123], Spatial Pooling [86], MT-LDCF [124], MCF [125], NNNF [126], Y-PD [62], YOLOv2PD [122], Yolov3 [16] and YOLOv4 [17]. For INRIA Person dataset, the proposed model achieved lowest log-average miss rate of 6.4%, which was superior to that of several SOTA detectors.

4.5.3 Caltech Pedestrian

The quantitative test results analysis of the proposed model was performed on Caltech Pedestrian dataset [103]. Table 4.6 shows the experimental results of the proposed model and these were compared with several SOTA approaches. The evaluation results of the proposed model was on Caltech reasonable subset only and considered IoU @0.5.

Table 4.6 Comparison of testing time performance metric with several SOTA relevant approaches with that of the Caltech pedestrian dataset (Reasonable subset IoU@0.5).

Models	Reasonable	Test time (s/img)
RPN+BF [47]	9.58	0.5
FDNN [63]	8.6	0.16
TLL-TFA [53]	7.4	0.25
RTPD [127]	8.9	0.13
SSNet [57]	8.92	0.43
SA-Fast RCNN [49]	9.68	0.59
YOLOv2PD [122]	7.48	0.29
MSCM-ANet [65]	7.1	0.06
SSAD	**6.91**	**0.05**

To test the effectiveness of proposed model, testing was performed on Caltech pedestrian dataset [19]. Table 4.6 shows the experiment results of the proposed method and the comparisons with several SOTA approaches, namely, RPN+BF [47], FDNN [63], TLL-TFA [53], RTPD [127], SSNet [57], SA-Fast RCNN [49], YOLOv2PD [122] and MSCM-ANet [65]. On Reasonable scenario, the proposed model achieves a decent low log-average miss rate and test time than compared to the other SOTA detectors.

4.5.4 City Persons

The quantitative assessments of the proposed model were performed on City person's pedestrian dataset [104]. Table 4.7 shows the experimental results of the proposed model which was compared with the results of several state-of-the-art approaches. The proposed model tested qualitative results on City person's reasonable subset only and considered IoU @0.5.

Table 4.7 Comparison of the metrics parameters and testing time of several SOTA relevant approaches with that of the City Persons dataset (Reasonable subset IoU@0.5).

Models	Back bone	Reasonable	Param. (MB)	Test time (s/img)
DDFE [91]	ResNet 50	12.9	168.8	0.22
FPN [98]	ResNet 50	15.4	–	–
Rep Loss [128]	ResNet 50	14.6	188.5	0.35
AMSNet [129]	ResNet 50	13.9	–	–
CSANet [116]	ResNet 50	12	–	0.32
TLL+MRF [53]	ResNet 50	14.4	230.5	0.41
ALFNet [130]	ResNet 50	13.1	191.1	0.27
MSCM-Anet [65]	ResNet 50	11.95	154.1	0.15
Cascade RCNN [69]	HRNet	11.2	–	–
Cascade RCNN [69]	SwinT	**9.2**	–	–
SSAD	ResNet 50	12.6	**97.7**	**0.11**

The robustness of the proposed model, was made on City person's dataset [104]. Table 4.7 shows the experimental results of the proposed method and the comparison with the several SOTA approaches, namely, DDFE [91], FPN [98], RepLoss [128], AMSNet [129], CSANet [116], TLL+MRF [53], ALFNet [130] MSCM-Net [65] and Cascade RCNN [69]. On reasonable scenario, the proposed model achieves a better log-average Miss rate and low testing time when compared to other SOTA detectors. The proposed model outperforms over other SOTA detectors in terms of Parameters (MB) size and testing time. However, SSAD model fails to achieve low log-average miss rate and falls behind MSCMNet [65] and

Cascade RCNN [69] models. Figure.4.4 shows the detected results of proposed network on City person's dataset.

4.5.5 Qualitative Analysis

Figure 4.4 shows qualitative test results of the proposed model on city persons test datasets. The applied test image size is 513x513.

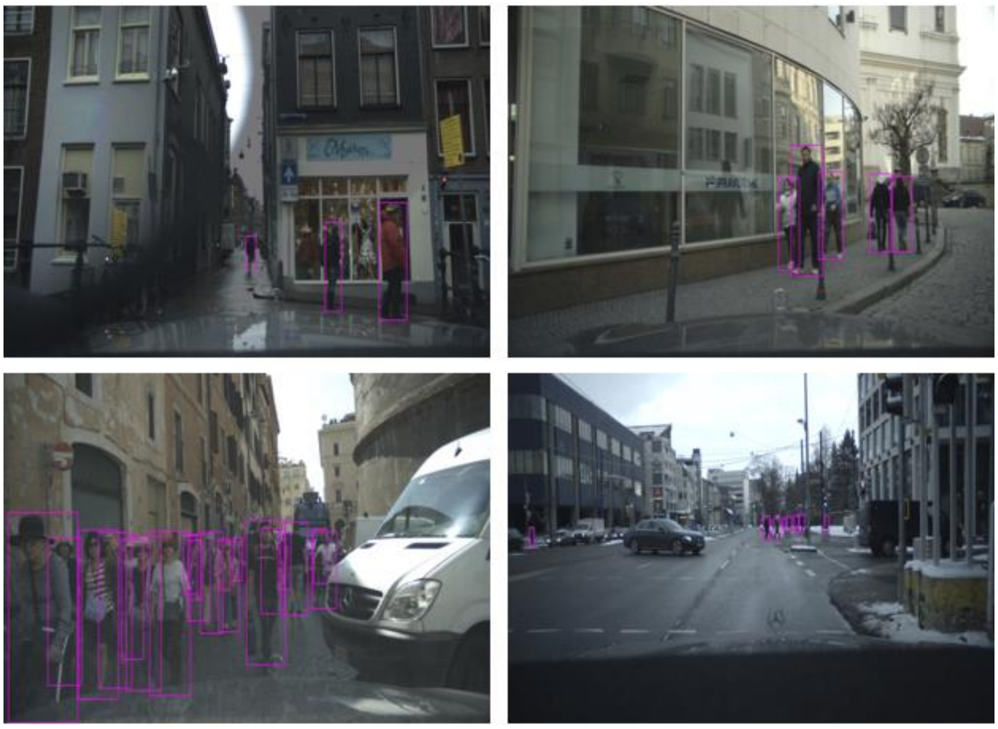


Figure 4.4 Detected results of City Persons dataset on SSAD Network.

4.5.6 Visualization of Multi-scale Attention map

To visualize the effectiveness of multi-scale attention map mechanism, on SSAD model at different scales is investigated. In this work multi-scale attention maps were projected onto original images. The sample visualization of attention map is shown on PASCAL VOC-2007 test dataset [100] and shown on three classes. i.e., Car, Bus and Pedestrian. The multi-scale attention map only highlights the required classes and their locations, which indicate the feature relations; it helps the proposed model to concentrate more on useful and important regions as shown in Figure 4.5. It also guides the model to focus on small targets in shallower layers, while in deeper layers, it highlights bigger targets. Finally, it is observed clearly that the multi-scale attention map rejects unwanted regions and focuses mainly on crucial regions and this would help in quick determination of negative anchor boxes.

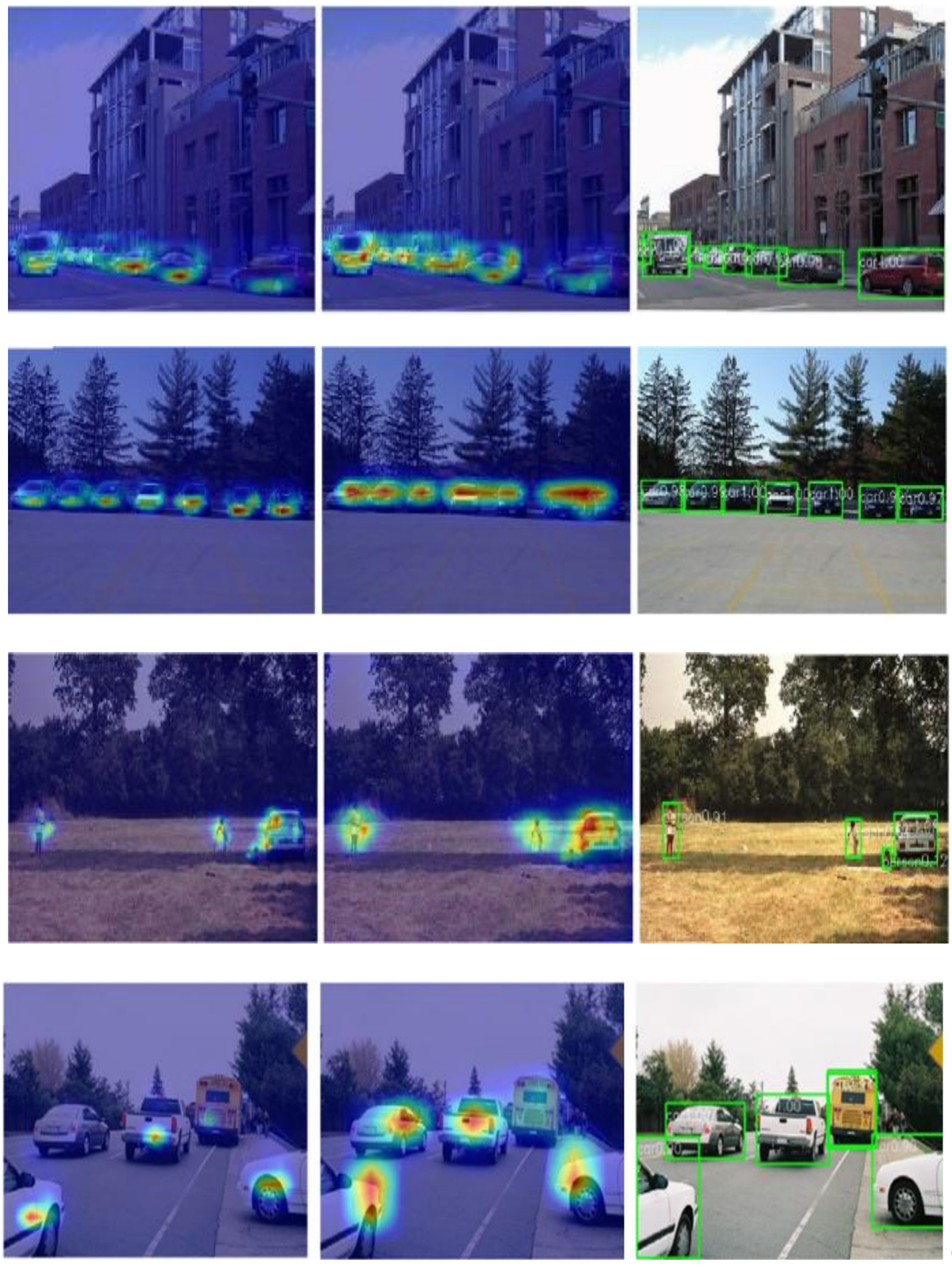


Figure 4.5 Visualization of PASCAL VOC 2007 test set using attention map.

4.6 Summary

This chapter proposed a new novel framework for effective pedestrian and vehicle detection for autonomous driving systems. In this work, two new contributions are presented. The first contribution in that the, proposed model learns to highlight pedestrian and vehicle regions on extracted feature maps and suppress irrelevant regions from global relation information and provide reliable guidance for autonomous driving. Therefore, SSAD network is highly capable of focusing on different regions for detecting multi-scale and occluded pedestrians and vehicles effectively and accurately. Second contribution is that prior to the prediction module (PM), a Multi-scale Attention Unit (MAU) module is embedded into SSD in order to improve detection accuracy and preserves the simplicity and efficiency of SSD. Therefore, SSAD network is simple, efficient and achieves higher detection accuracy while detecting smaller and occluded pedestrians and vehicles in different environmental conditions. However, SSAD network is heavy so it fails to achieve real-time performance when deployed on any low-end edge device for autonomous driving application.

Chapter 5

Optimized MobileNet+SSD: A Real-time Pedestrian Detection

The main goal of this chapter is to design and develop a real-time pedestrian detection algorithm without losing any detection accuracy while detecting smaller pedestrians. Secondly, the proposed algorithm is implemented in real-time on a low-end edge device Jetson Nano board. Both quantitative and qualitative analysis of the proposed model was verified on PASCAL VOC- 2007 test dataset and a comparative analysis was made with existing state of the art detection algorithms. Real-time performance of the proposed model was verified on Jetson Nano board and a comparative analysis was made with existing state-of-the-art pedestrian detection algorithms.

5.1 Introduction

Computer vision (CV) has had major growth in numerous fields such as robotics, medical imaging, microscopy, image retrieval, face recognition, and modern industrial applications. In recent years, pedestrian detection is a challenging problem in CV applications like semi-autonomous vehicles and self-driving cars. It is also an indispensable and remarkable task in an intelligent video surveillance system. It has a clear-cut uses in automotive applications because of its use in safety systems. This was offered by many car manufacturers (e.g. Ford, Nissan, GM, Volvo) as an advanced driver assistance system (ADAS) option in 2017.

In daily life, we often drive through busy environments, traffic, and challenging weather conditions. Road accidents are one of the major causes of death. So, there is a need to design safer automobiles by providing tools to inform and warn the driver about pedestrians as well as other relevant information. This can save many lives in road accidents i.e. the concept of self-driving cars. For pedestrian detection, the designed network should be able to differentiate between multi-scale pedestrians and other complicated objects that are also present in the image backgrounds. For that, we need to withdraw the features of pedestrians such as shape, colour, and behaviour. The intra-class variations of pedestrians such as clothing, backgrounds, lightning, articulation and occlusion are the main challenges. The detection is more precise if the expected and plausible features are extracted perfectly from the images.

A feature is an interesting part of an image and the main focus of feature extraction is to extract information from the image and make some decisions at every image point so that the given features of any object are present in that image. So the extraction of image information can be done by convolutional neural networks (CNNs) as at present they have very good capability of high-level feature extraction of an image and are also useful in low-level feature detections. Deep convolutional neural network (DCNN) is classified basically into two categories, base and detection network. AlexNet [29], VGGNet [87], Xception [131], ResNet [79], DenseNet [80] and MobileNet [81] are widely used baseline networks. High-level features are provided by classification or detection by base network. MobileNet uses convolution to produce high-level features just like other base networks as well to decrease the number of network parameters. For image classification, a fully connected (FC) layer is the final layer of CNN. Classification layers can be removed and replaced by detection networks.

Examples of detection layers include Feature-Fused [132], Feature Pyramid Network (FPN) [39], RCNN ("Region-based Convolutional Neural Network") series [10-13], YOLO ("You Only Look Once") series [14-16], SSD ("Single Shot Multi-Box Detector") series [19-21] etc. The use of SSD on the last convolutional layer results in a detection task.

Some studies on pedestrian detection [9] [133] use different types of targets as well as detection networks that are implemented using either traditional methods or deep learning-based methods. Traditional methods widely used in machine learning are "Histogram of Oriented Gradients" (HOG) [7], Haar-like features using patterns of motion and appearance [66], deformable models [25], etc. Deep learning-based methods include RCNN series [10-13], YOLO series [14-16], and SSD300/512 [19]. However, there is still a need for further optimization in these networks for real-time multi-scale pedestrian detection on low-end edge devices. Therefore, in this chapter, a real-time pedestrian detection algorithm was proposed without having any loss in detection accuracy and also verified the real-time implementation on Jetson Nano board.

The contributions of this work are summarized as follows:

- The SDD + VGG [19] model fails to detect smaller pedestrians, and to offset this, shallow convolution layers conv4_3 and conv5_3 are concatenated in the proposed Optimized MobileNet+SSD network which improves feature map information and this in-turn improves detection performance.
- A concatenation feature fusion module is introduced which can add contextual information in the proposed Optimized MobileNet+SSD network in order to improve the detection accuracy of pedestrians.
- The proposed network seeks to extract the best hyper-parameters because, fine-tuning hyper parameters such as depth, stride, filter, shape, optimizer, etc., plays a vital role in the optimization of the network and also helps in reducing computational power while execution.
- Finally, from experimental results, it is verified that the proposed model outperforms on Pascal-Voc 2007 test dataset, and shows better detection effect while detecting denser and small-scale pedestrians during low light and darker images and runs at 34.01 FPS on Jetson Nano board.

5.2. Methods and Materials

5.2.1. Overview of Single Shot Detector (SSD)

The architecture of SSD+VGG backbone network shown in Figure 5.1. SSD is referred to as regression-based object detector. The model can solve conflict caused by translational invariance and variability and can achieve better detection accuracy as well as speed. Each selected feature map which consists of K frames varies in size and width to height ratio. Each frame is termed as anchor box. Figure 5.1 shows bounding boxes on feature maps of different convolution layers. B class score and 4 position parameters are predicted by default at every bounding box. B x K x w x h class score and 4 x K x w x h position parameters need to be detected for w-h feature image. It requires a size 3 x 3 (B + 4) x K number of w x h convolution kernel for processing the feature map. The convolution result is considered to be the last feature for bounding box and classification + regression.

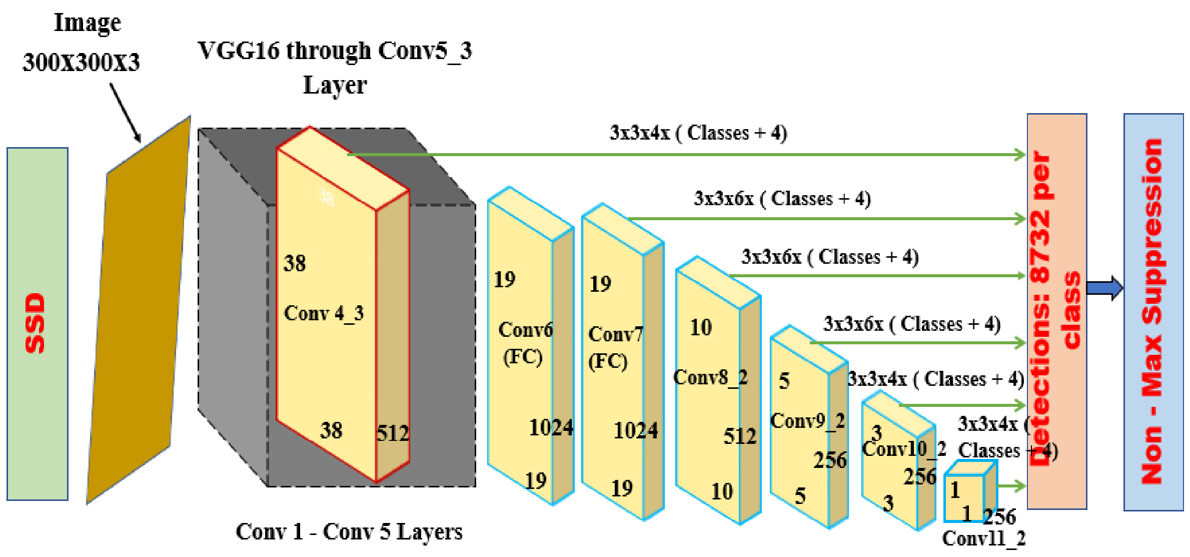


Figure 5.1 SSD300 with VGG-16 backbone network through Conv5_3 layer.

The mathematical expression for bounding boxes for every feature map is

$$S_K = \frac{S_{Min} + (S_{Max} - S_{Min})^* (K-1)}{(M-1)} \qquad \text{(K varies from [1, M])} \qquad (5.1)$$

M indicates the count of feature maps and S_{Min} and S_{Max} are settable parameters. The same five types of aspect ratios, a = {1, 2, 3, 0.5, 0.33} generate anchor boxes to tune the fairness of feature vectors in training as well as while testing experiments. So, each bounding box can be mathematically expressed as:

$$h^a_K = \frac{S_K}{\sqrt{a_r}} \qquad (5.2)$$

$$w^a_k = S_k \sqrt{a_r} \qquad (5.3)$$

Here h^a_k and w^a_k represent the height and width of the corresponding bounding box respectively. When the aspect ratio is one, a bounding box $s'_k = \sqrt{s_k} . \sqrt{s_{k+1}}$ should be added.

The centre of every bounding box varies from $\left(\frac{(i + 0.5)}{|f_k|}, \frac{(j+0.5)}{|f_k|}\right)$ and if $|f_k|$ represents the size of K^th^ feature unit, i, j € [0, $|f_k|$]. Figure 5.2 shows that a dog is perfectly matched to a bounding box in the 4×4 feature map while it is not matched to any of the bounding boxes in the 8×8 feature map. Since the bounding boxes with different scales do not match with the dog box, they were considered as negatives during training.

The intersection over union (IoU) can be mathematically expressed as:

$$\text{IoU} = \frac{(\text{AREA (C)} \cap \text{AREA (D)})}{(\text{AREA (C)} \cup \text{AREA (D)})} \tag{5.4}$$

If the calibration and bounding box Intersection over union value exceeds 0.5, it indicates that for the respective category, the bounding box matches the calibration box.

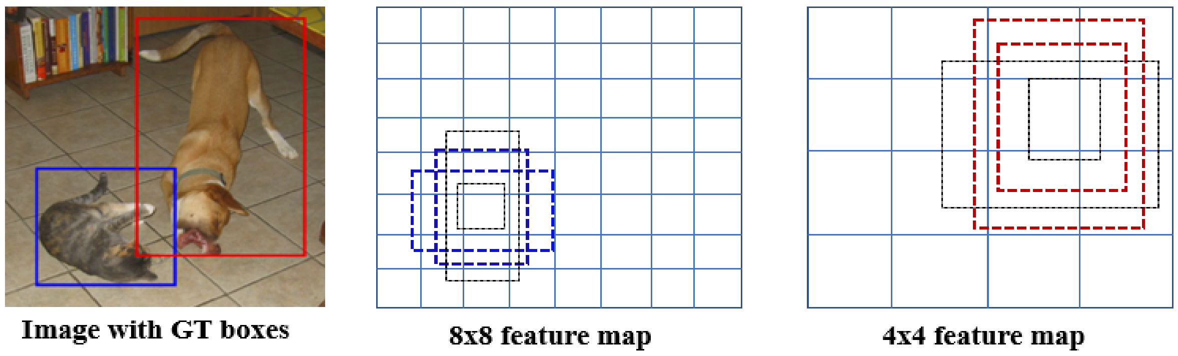


Figure 5.2 SSD Multiple Bounding Boxes for Localization and Confidence.

The sum of bounding box regression's position loss i.e. L_{loc} (r, l, g), and the classification regression confidence loss is referred to as total loss function and is expressed as shown below:

$$L(s,r,c,l,g) = \frac{1}{N}.(L_{conf}(s, c) + \alpha L_{loc}(r, l, g) \tag{5.5}$$

where 'r' and's' are eigenvectors of position loss and confident loss respectively, α is a parameter to tune both position and confidence loss; 'I' is the offset including the scaling offset of the height and width and translational offset of the centre of the predicted boxes, 'N' represents the number of bounding boxes matching the calibration box for a given category and 'g' is the calibration box of the target actual position.

5.2.2 Standard convolution and Depth wise convolution

Figure 5.3 shows MobileNet's standard convolution layers (left side) and depth-wise and point-wise separable convolutional layers (right side). Conv_Dw_Pw is a deep and separable convolution structure that consists of two layers, namely, Pointwise (Pw) and Depth-wise layers (Dw). The Dw layer uses 3x3 kernels and the Pw layer uses 1x1 kernels which are also called deep convolutional layers and common convolutional layers respectively. So, the result of each convolution is then processed by batch normalization (BN) algorithm and ReLU6 activation.

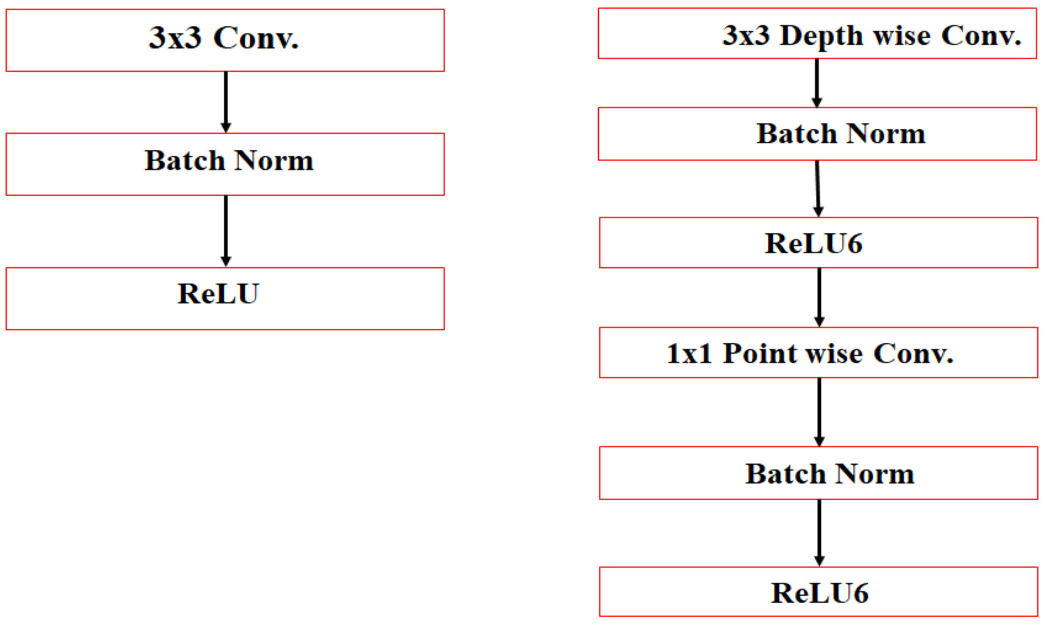


Figure 5.3 Standard convolution (left) and Depth wise convolution modules (right) with Batch Norm and ReLU6.

The Mobilenet architecture uses depthwise separable convolution (DSC). It uses the standard convolution but just once on the very first layer. After the first layer, all further layers have DSC.

Depthwise separable convolution (DSC) module is just a combination of depthwise + pointwise convolution. The main difference between standard and depthwise separable is that unlike the former, the latter performs convolution on each image channel separately. For an image having 3 image channels, it performs convolution separately and creates an output image that also has image channels. Each channel has then a separate set of weights. The main motive of depthwise operation is applied to a single channel at a time. This results in more precision in edge detection, color filtering, etc. In depthwise separable convolution, if the dimensions of the input feature map ($D_F \times D_F \times Q$), are applied to a filter of kernel size ($D_K \times D_K \times 1$), it produces an output feature map ($D_F \times D_F \times P$). This is then processed by pointwise convolution (1x1convolution) on P channels. The final output feature map generated is $D_k \times D_k \times Q \times D_F \times D_F$.

The objective of doing pointwise convolution is to combine separate channels in the output of depthwise convolution to create new features. Using the two methods results in separate filtering and combining unlike in standard convolution where the two are one process. Another advantage of depthwise separable over standard is that even though they both finally result in filtering data and making new features, standard convolution needs much more computational work to get the result and hence needs to learn more weights. Meanwhile, DSC implements convolution operations much more efficiently and uses fewer parameters. Table 5.1 shows the computation cost and the number of parameters required for both standard convolution and depthwise separable convolution.

Table 5.1 Computation cost and no. of parameters required for Standard convolution and DSC.

Layer	Parameter Size	Computation Cost
Standard Conv	$D_k \times D_k \times Q \times R$	$D_k \times D_k \times Q \times R \times D_F \times D_F$
Depthwise Conv	$D_k \times D_k \times Q$	$D_k \times D_k \times Q \times D_F \times D_F$
Depthwise Separable Conv	$D_k \times D_k \times Q + 1 \times 1 \times Q \times R$	$D_k \times D_k \times Q \times D_F \times D_F + Q \times R \times D_F \times D_F$

Therefore the reduction of computation cost obtained is :

$$= \frac{D_k x D_k x Q x D_F x D_F + Q x R x D_F x D_F}{D_K x D_K x Q x R x D_F x D_F} = \frac{1}{R} + \frac{1}{D^2_K} \tag{5.6}$$

Therefore the parameters reduce to :

$$= \frac{D_k x D_k x Q + 1 x 1 x Q x R}{D_K x D_K x Q xR} = \frac{1}{R} + \frac{1}{D^2_K} \tag{5.7}$$

When width multiplier “α” was applied, the computation cost of DSC is given by:

$$D_K x D_K x \alpha Q x D_F x D_F + \alpha Q x \alpha R x D_F x D_F \tag{5.8}$$

Experiments have proved that using a 3 x 3 kernel and saving computation time by the new approach is about 9 times faster without making any difference in detection. MobileNet uses up to 13 depthwise convolutions in a row.

5.2.3 MobileNet

MobileNet [81] model was proposed by Google and is a type of base architecture highly suited for embedded-based vision applications with low computing power. The MobileNet architecture uses depthwise separable convolutions instead of standard convolution. This reduces the number of parameters significantly compared to the network with normal convolution with the same amount of depth in the network, which results in lightweight deep neural networks. The "Batch Normalization" layer was included in each layer of the newly appended structure to prevent the gradient's disappearance. MobileNet is easy to train and takes relatively less time while training, which is highly desired for real-time implementation. This makes the network more reliable compared to VGG-16 [87] and other available architectures.

5.2.4 Jetson Nano Evaluation Board

. Nvidia Jetson Nano board is an embedded developer kit mainly meant for developing embedded systems that need high processing power for CV, deep learning, machine learning, and image/video processing applications. Figure 5.4 shows Nvidia Jetson Nano evaluation board.

The specifications of Jetson Nano are Maxwell architecture based graphics processing unit (GPU), 128 CUDA cores, quad-core ARM A57 central processing unit (CPU) works at 1.43 GHz with 4GB of LPDDR4 memory, data transfer at 25.6 giga bits per seconds (Gbps), operates with Linux operating system, 4K video encoding and decoding at 60 FPS, and has a processing power of 472 GFLOPS. Since the Jetson Nano board consumes low power i.e. less than 5 watts, has in-built GPU cores, and is low cost compared to other embedded boards, it is a popular board for implementing real-time pedestrian detection algorithms.

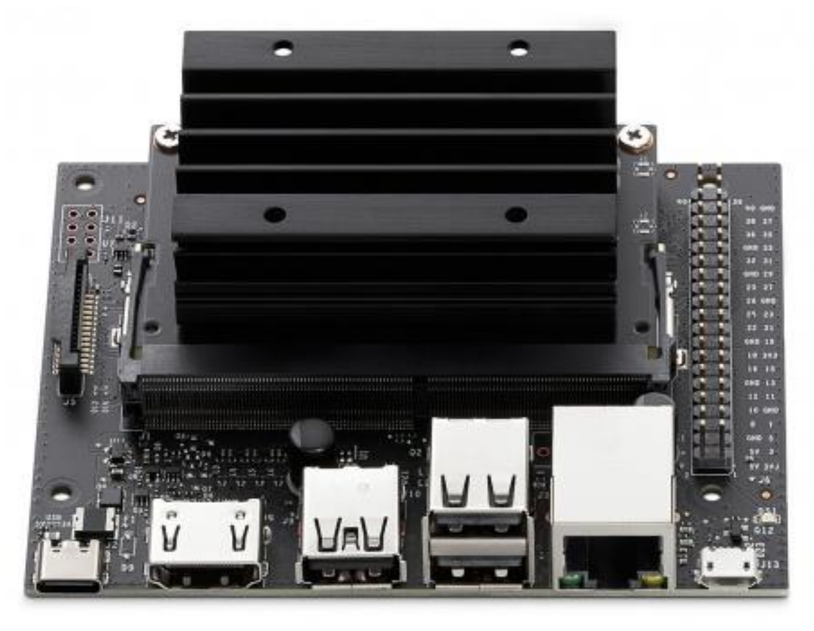


Figure 5.4 Nvidia Jetson Nano Evaluation Board.

5.3 Proposed Methodology

Figure 5.5 shows that the Optimized MobileNet+SSD network is composed of 21 convolutional layers. The target feature layers for detection are Conv 4_3, Conv 13, Conv 14_2, Conv 15_2, Conv 16_2 and Conv 17_2. This network enhances the information of the newly added shallow convolutional layer Conv 5_3 to detect smaller and denser objects. Since Conv 4_3 and Conv 5_3 feature maps are different in terms of size, to make the same size, the Conv 5_3 layer is followed by a deconvolution layer (2x up-sampled). On both layers, 3x3x256 convolution layer and applied normalization with 10, 20 different scales were used incorporating better features to fuse. Finally, both convolutional layers were concatenated before applying a 1x1x256 convolutional layer which reduces dimensions and feature recombination to generate the final fusion feature map as shown in Figure 5.6.

A feature fusion concatenation module was introduced in the proposed network, which would inject contextual information into shallower layer Conv 4_3 since this layer lacks semantic information and is an important supplement for detecting small-scale and dense pedestrians. Therefore, the detection performance of small-scale pedestrians was improved by passing the captured semantic information in convolutional forward computation back towards shallower layers.

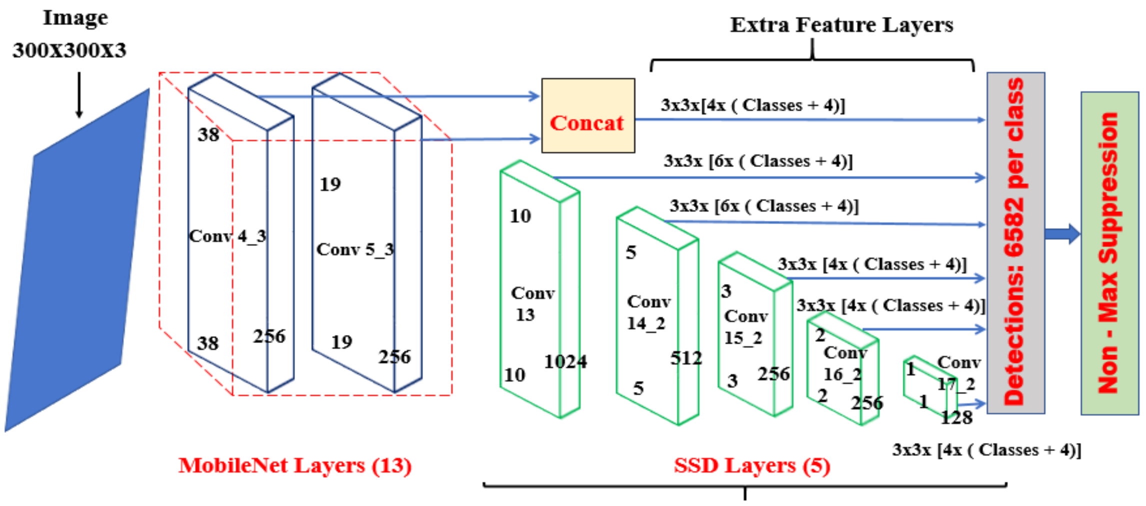


Figure 5.5 Optimized MobileNet+SSD backbone network through Conv4_3 layer.

While designing the most effective feature fusion concatenation module, we explored different layer feature fusion trials. Finally, Conv 4_3 and Conv 5_3 shallow layers were selected and fused as these layers would introduce less background noise while detecting small-scale pedestrians. Higher shallower layers after Conv 5_3 possess large receptive fields and would introduce more background noises while detecting small-scale pedestrians. The flow of the proposed detection algorithm is shown in Figure 5.7.

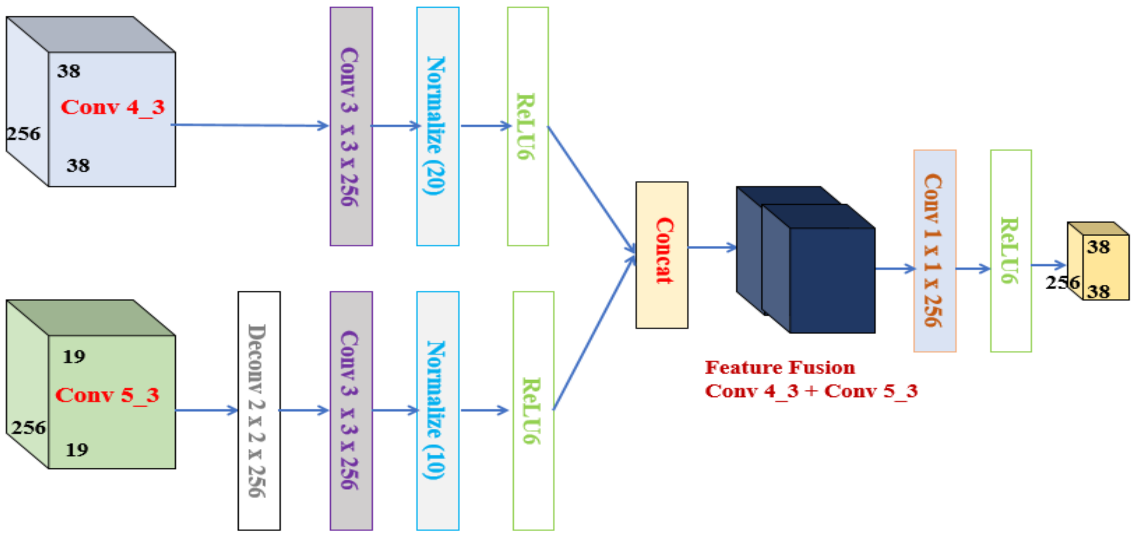


Figure 5.6 Feature Fusion Concatenation Module.

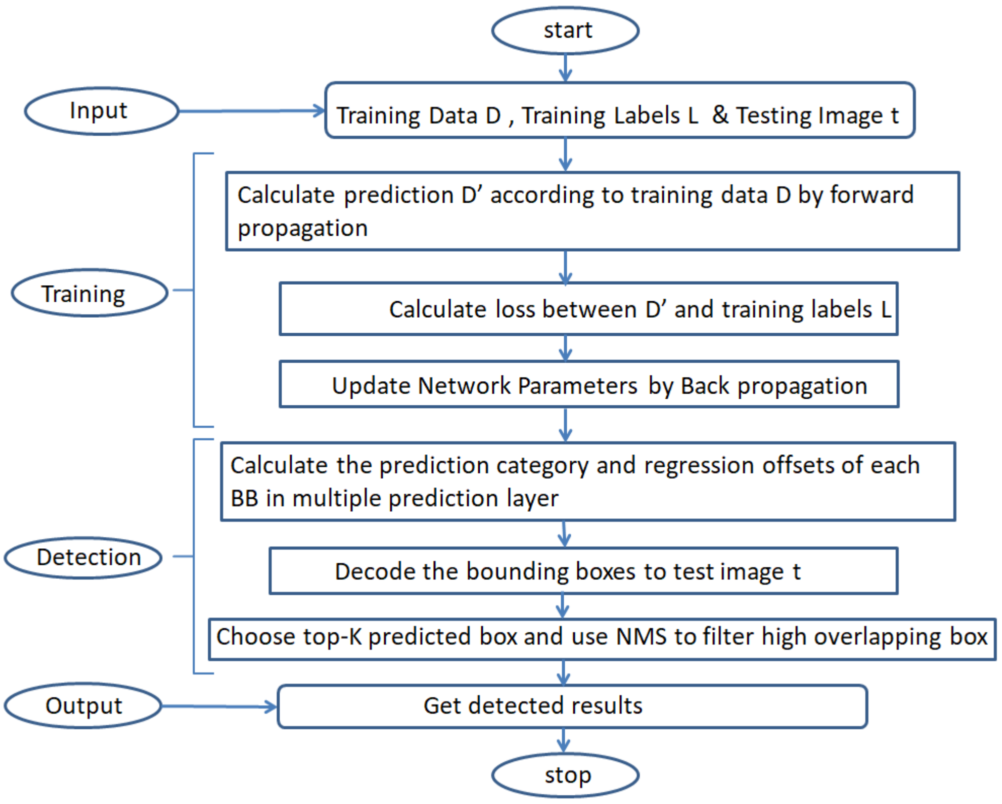

start
Input
Training Data D , Training Labels L & Testing Image t
Calculate prediction D' according to training data D by forward propagation
Training
Calculate loss between D' and training labels L
Update Network Parameters by Back propagation
Calculate the prediction category and regression offsets of each BB in multiple prediction layer
Detection
Decode the bounding boxes to test image t
Choose top-K predicted box and use NMS to filter high overlapping box
Output
Get detected results
stop

Figure 5.7 Flowchart of the proposed algorithm.

5.4 Experimental Discussions

To showcase the power of the Optimized MobileNet+SSD model, it was evaluated on two datasets and made comparative analysis with the existing state-of-the-art pedestrian detection models. The effectiveness of the proposed model is verified both quantitatively and qualitatively.

5.4.1. Datasets

The proposed methodology was trained on Pascal Voc-2007 [100] trainVal dataset and tested on Pascal Voc-2007 dataset.

(i) Pascal VOC 2007: The "Pascal Visual Object Classes Dataset" is Pascal Voc-2007 test dataset, and a collection of images of around 20 classes. During the training, the model used Pascal Voc-2007 trainVal (5011 images). The 'Test' image set contains around 4952 images. Since the Pascal dataset background is more complicated, the degree of occlusion and human postures is different, and the size of the humans is not the same. Therefore, several images were used to improve the generalization of the trained network to meet complex real-time traffic scene environment.

(ii) Caltech Pedestrian [103]: This dataset contains a set of video sequences of 640×480 size. It includes some train (set 00 to set 05) subsets and test (set 06 to set 10) subsets. There are about 350,000 bounding boxes in 250,000 frames and 2300 pedestrians were annotated and only “person” and “people” were used in our experiments. Finally, the proposed model was only tested on Caltech pedestrian dataset.

5.4.2. Experimental Setup

The experiments were carried out on a workstation during the training phase and finally, the testing phase was performed both on the workstation and Jetson Nano evaluation board. Table 5.2 shows the experimentation configuration setup. Figure 5.8 shows the experimental setup and real-time pedestrian detection captured on the Jetson Nano evaluation board. In order to know or monitor the performance of core components such as GPU, CPU, memory and other used by Jetson Nano during the testing the dataset. First install jetson stats package and then in terminal window run sudo jtop and which will display a dashboard and to get individual stats of each core components click on the bottom tab-menu.

On system we need to integrate CUDA then GPU will be enabled. During the network testing, when GPU=1 then model would run on GPU and it delivers higher inference speed. Whereas when GPU=0 then the network would run on CPU and it delivers very low inference speed.

Table 5.2 Experimental configuration setup.

Names	Experimental Configuration
OS	Windows 10 Pro
CPU/GHz	Intel Xeon 64 bit CPU @3.60
RAM/GB	64
GPU	NVIDIA Quadro P4000, 8 GB, 1792 CUDA cores
GPU acceleration library	CUDA10.0, CUDNN7.4
Tensor flow [134]	2.x
Keras	2.2.x

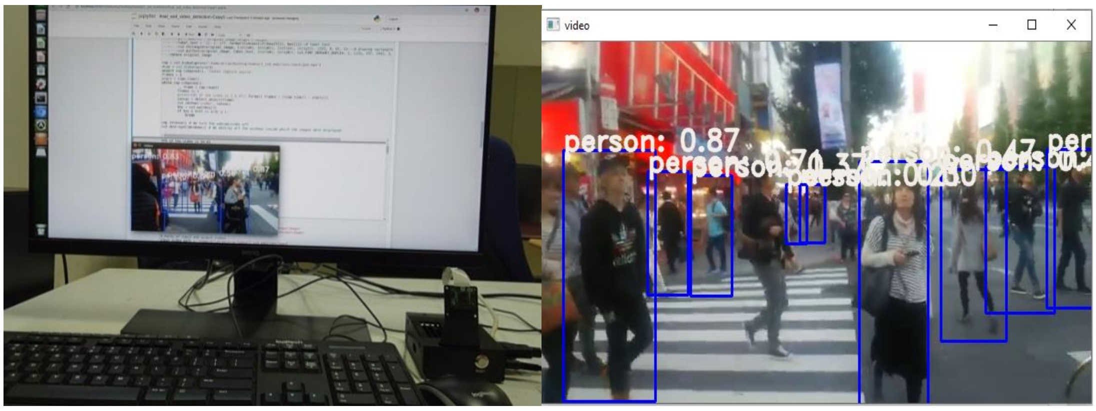


Figure 5.8 Experimental setup and captured results on Jetson Nano Evaluation board.

5.4.3. Training and Evaluation of Performance Parameters

The model training was performed on trainVal (5011) images of Pascal VOC-2007 dataset and tested on Pascal Voc-2007 test (4952) images. The input image size was set to 300 x 300 and various data augmentation techniques were applied such as flipping, cropping, and random sampling to enhance the training process. We followed the standard evaluation methods used in [19]. The proposed method used ADAM (Adaptive Moment Estimation) optimizer instead of SGD optimizer while training Pascal Voc-2007 dataset. The hyper-parameter values used while training the proposed model are batch size 8, weight decay 0.005, epsilon 10^{-9}, beta1 and beta2 0.9, 0.999, iteration steps 150, learning rate .001 and 110 epochs. To evaluate the robustness of the proposed network, the commonly used performance parameters for pedestrian detection such as Recall, Precision, Average Precision (AP), speed (Frames per second-FPS) and memory footprint were employed.

(i) Recall: Recall is defined as the % of total relevant results correctly classified by the algorithm.

$$\text{Recall} = \frac{\text{True Negative}}{\text{Predicted Results}} \text{ Or } \frac{\text{True Positive}}{\text{True Positive + False Negative}} \quad (5.9)$$

(ii) Precision: This is defined as % of results that are relevant i.e. represents the accuracy of prediction.

$$\text{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \text{ Or } \frac{\text{True Positive}}{\text{True Positive + False Positive}} \quad (5.10)$$

(iii) Average Precision (AP): It is the area under the precision-recall curve and it shows the correlation between precision and recall at a different level of confidence scores.

$$\text{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}} \quad (5.11)$$

5.5 Results & Discussions

The proposed model's test results on Pascal Voc-2007 test dataset were drawn between Precision versus Recall as in Figure 5.9. This graph depicts that with an increase in recall value at the convergence point, the precision starts gradually decreasing. The predictor built on Keras runs predictions over the entire dataset, matches predictions to ground-truth boxes, computes precision-recall curves for pedestrian class, and samples 11 equidistant points from the precision-recall curves to compute the average precision for the pedestrian class, giving an AP value.

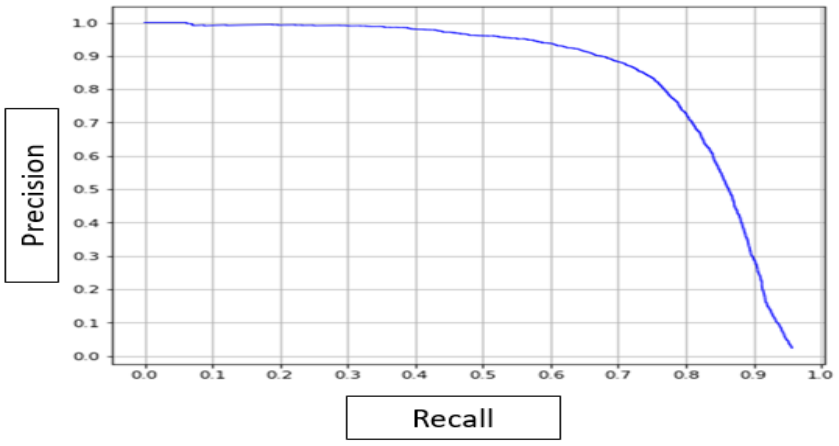


Figure 5.9 Precision-Recall curve of Optimized MobileNet+SSD Network.

5.5.1 Quantitative Analysis on Pascal Voc dataset

The robustness of the proposed model robustness was verified both quantitatively and qualitatively on the widely used pedestrian dataset. Table 5.3 shows the comparison of average precision (AP), speed (FPS) and weight file of the proposed model with the state-of-the-art (SOTA) models on Pascal Voc-2007 test dataset.

Table 5.3 Comparison of quantitative metrics of the proposed model with the existing SOTA models. (IoU@0.5).

Models	AP (%)	Speed (Quadro P4000)	Weight (MB)
Fast RCNN [11]	68.4	0.5	513
Faster RCNN [12]	70.4	7	522
YOLO [14]	57.9	45	753

YOLOv2 [15]	76.8	38	203
SSD 300 (VGG16) [19]	71.4	46	100
SSD 512 (VGG16) [19]	76.8	19	103
100Hz DPM [22]	16	100	-
30Hz DPM [22]	26.1	33	-
Yolov3 [16]	78.12	35	247
Tiny Yolov3 [16]	68.54	**220**	33.9
Improved Tiny Yolov3 [43]	73.98	206	33.1
Proposed method	**80.04**	155	**23.6**

Comparing the results of Table 5.3 with existing SOTA models, the proposed model achieves better detection performance; the AP value reaches 80.04% on Pascal Voc-2007 test dataset, which is +3.24%, +1.92%, +11.5%, and +6.06% higher compared to SSD 512 [19], Yolov3 [16], Tiny Yolov3 [16] and Improved Tiny Yolov3 [43], respectively.

From the evaluation results, it is clear that the speed of Tiny-Yolov3 [43] is 220 fps, while that of the proposed model is only 155 fps. At the same time, the proposed model file is 23.6 MB, which is much smaller compared to Tiny Yolov3 [16] model file. The proposed model surpasses Improved Tiny Yolov3 [43] model both in terms of accuracy and weight file. To test the robustness of the proposed model, it was also tested on the Caltech pedestrian test dataset and achieved competitive results.

For real-time implementation, the proposed model was tested on a low-cost edge device Jetson Nano board. After training the proposed model on Quadro P4000 GPU, the whole model was tested on Nvidia Jetson Nano evaluation board with the same system environment. Generally, more CUDA cores represent higher computational power, with the same memory and frequency conditions. The number of cores on Jetson Nano is 256 which is only 1/7th of Quadro P4000 (1792) GPU. But Jetson Nano consumes less energy and has much lower computational power.

To test the validity of the proposed algorithm more intuitively, a captured real-time road video under low light conditions was fed for model detection verification. Figure 5.10 shows a comparison of the proposed model detection speed (FPS) with existing SOTA models on the Pascal Voc-2007 test dataset.

By using the same video for verification, the detection speed of all SOTA detectors on Jetson Nano was far slower compared to on Quadro P4000 GPU. From Figure 5.10, it is clear that the proposed model runs with a speed of 34.01 fps on Jetson Nano which is quite higher compared to SSD 512 [19], Yolov3 [16], Tiny Yolov3 [16], and Improved Tiny Yolov3 [43] SOTA models.

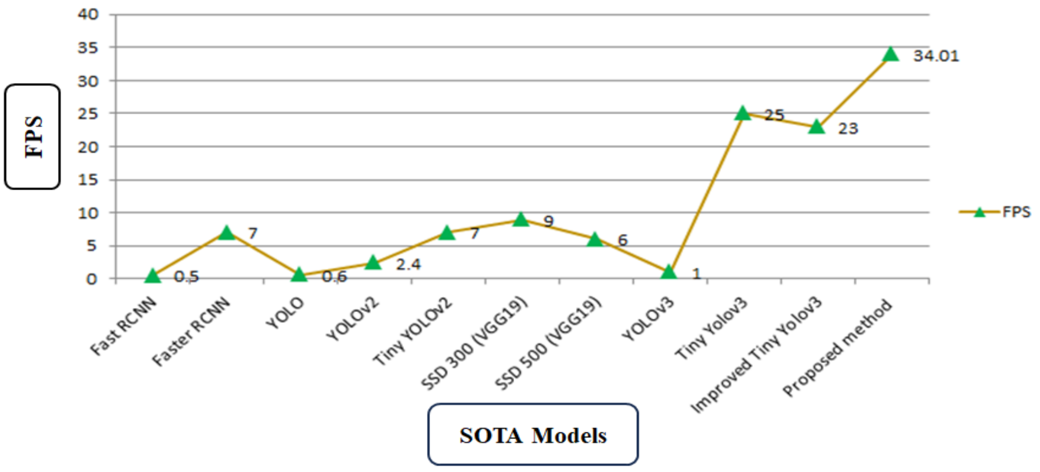


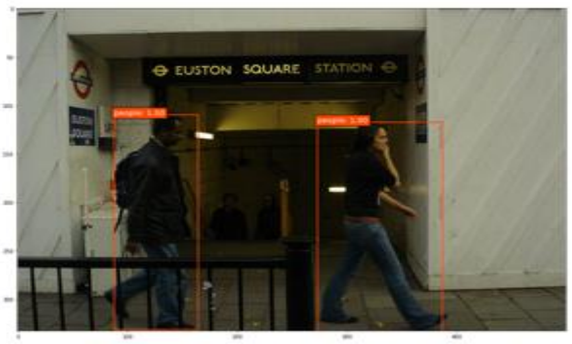
Figure 5.10 Comparison of detection speed of the proposed model with the existing SOTA models. (On Jetson Nano board).

5.5.2 Qualitative Analysis on Pascal Voc and Caltech datasets

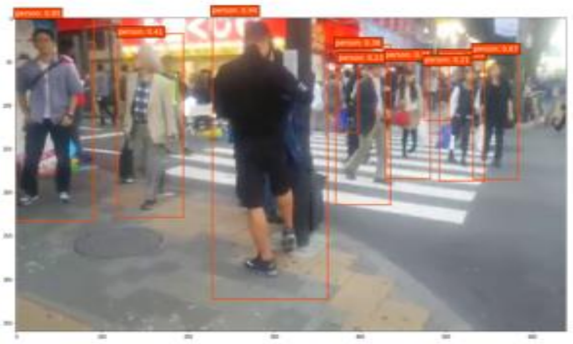
The model was trained on low-resolution images with a size of 300 x 300. This results in a fall in detection accuracy on low-resolution images. Nevertheless, with optimized MobileNet as a backbone model, the proposed model can detect and identify pedestrian class with an appreciable amount of accuracy.

Figure 5.11 shows qualitative sample detected images of both Pascal Voc-2007 and Caltech pedestrian test datasets. This model accurately detected different samples varying from a few people to several. The model perfectly segregates different persons without intermixing them, giving precise detection results.

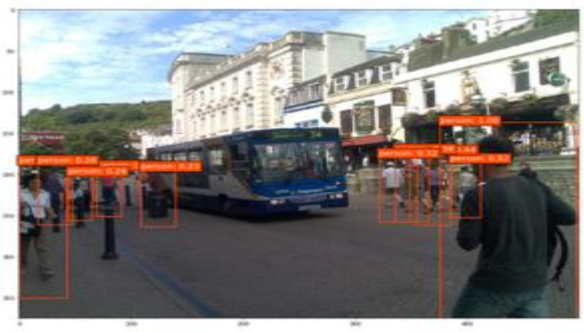
The proposed model was tested on low-resolution images and the detected results are shown in Figure 5.12. So, this model has a better detection effect while detecting dense and smaller pedestrians during low light and darker pictures while tiny-yolov3 [43] fails to detect pedestrians in darker pictures.



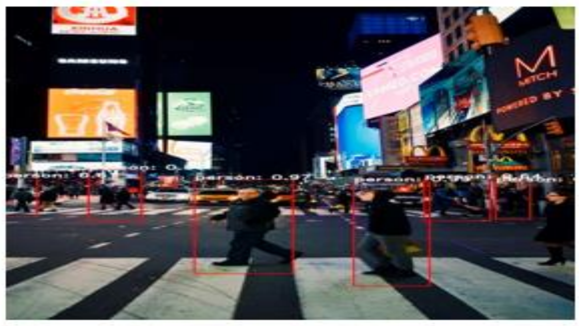
(a)



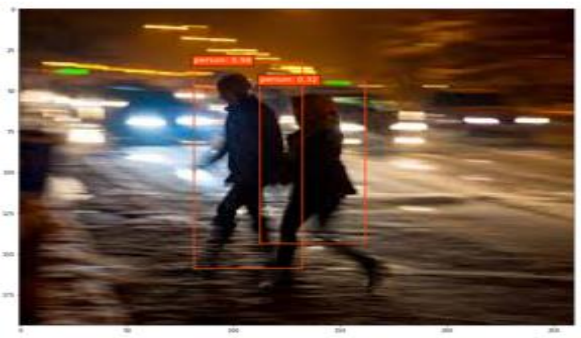
(b)



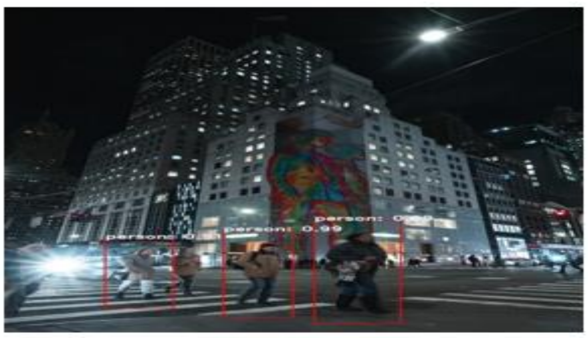
(c)



(d)



(e)



(f)

Figure 5.11 Detected sample images from Pascal Voc-2007 and Caltech datasets (512x512).

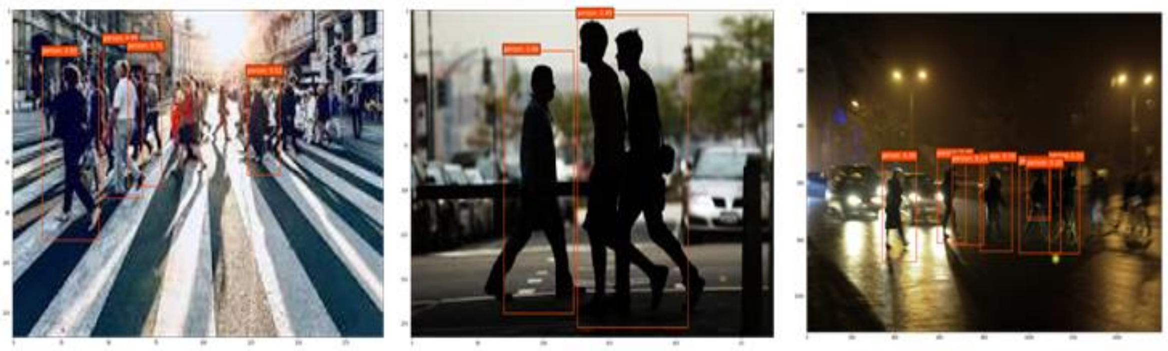
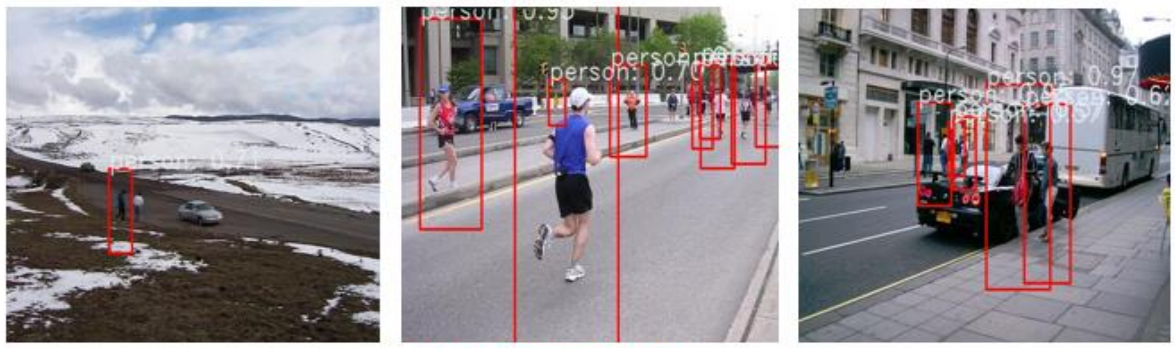
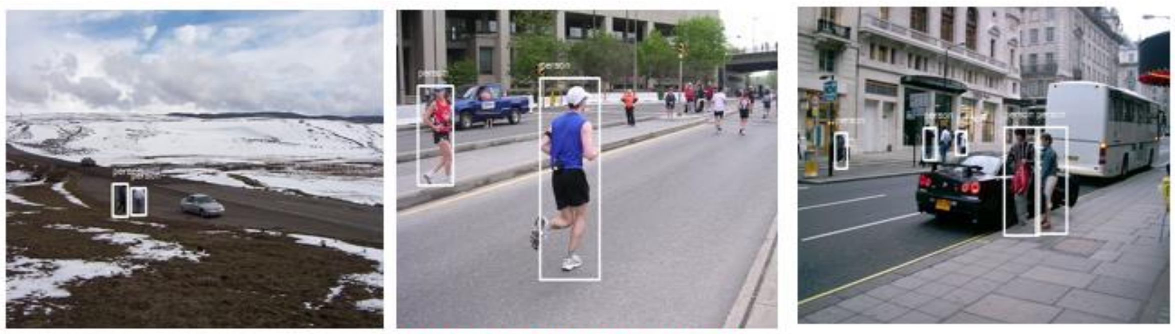


Figure 5.12 Detection examples on sample images from Pascal Voc-2007 and Caltech datasets on low-resolution images (512x512).

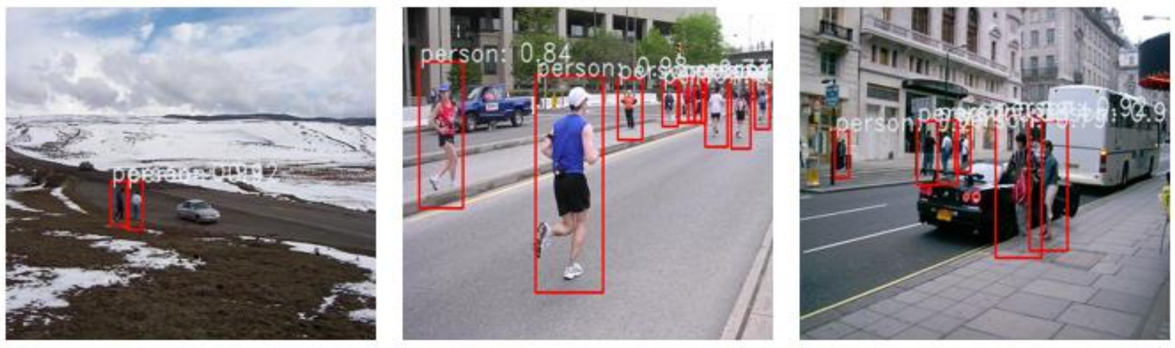
Since Pascal Voc-2007 dataset contains many small objects, our concern is only the pedestrian class, and thus around 300 images were manually gathered that cover mainly smaller pedestrians for testing the performance of the proposed model. Detection results of the original SSD, YOLOv3 and Optimized MobileNet+SSD models are shown in Figure 5.13.



(a) SSD detection results



(b) YOLOv3 detection results



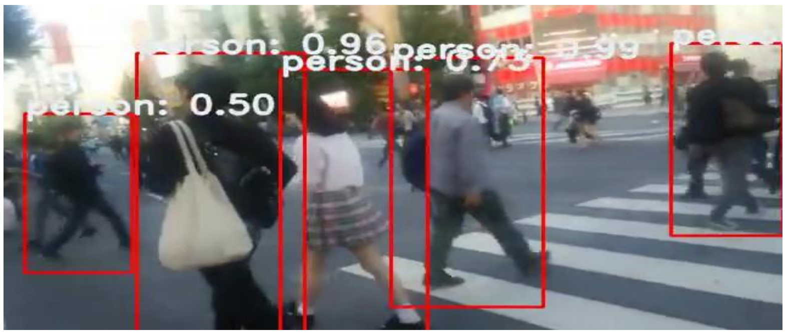
(c) Optimized MobileNet+SSD detection results

Figure 5.13 Detection results of (a) Original-SSD (b) YOLOv3 (c) Optimized MobileNet +SSD (512x512).

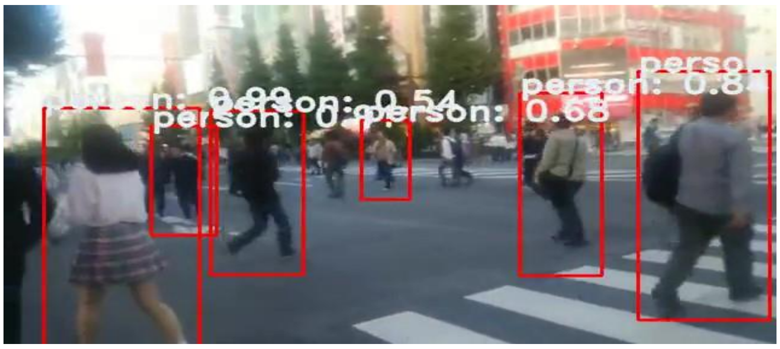
Figure 5.13 shows clearly that the proposed model performs better compared to the original SSD [19], YOLOv3 [16] while detecting small-scale and denser pedestrians in real-time.

To test the validity of the proposed algorithm more intuitively, we captured real-time road video under low light conditions and test results of randomly selected frames 498, 520, and 798 on Jetson Nano board, shown in Figure 5.14. The proposed algorithm was shown to have good adaptability to detect pedestrians under complex environments for real-time video.

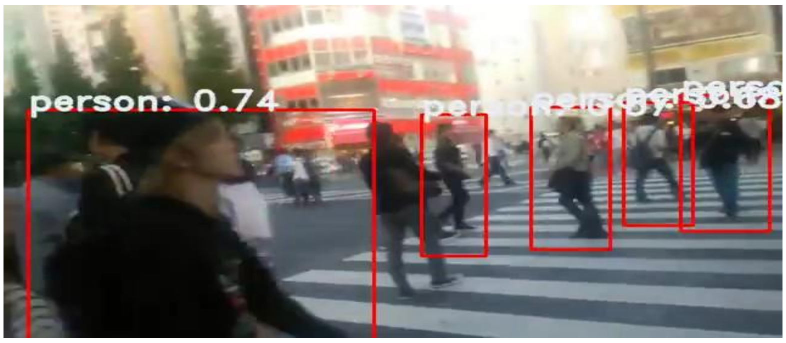
From Figure 5.14, it is clear that the proposed model when tested on Jetson Nano works better while detecting small-scale and denser pedestrians in real-time but fails to detect both occluded and distant smaller pedestrians.



(a)



(b)



(c)

Figure 5.14 Detection effect of the Optimized MobileNet+SSD Network on Jetson Nano.

5.6. Summary

This chapter proposed a new novel framework for small-scale real-time pedestrian detection for assisted driving system. In this work, four new contributions were presented. First contribution, looked at exploiting the contextual information, to achieve considerable improvement in the proposed model without losing any detection accuracy. Second contribution is that the number of network parameters decreased in this model while detection accuracy was still preserved. Third contribution was that by optimally fine tuning the hyper parameters, data augmentation techniques, optimizer, batch normalization etc., which helps in reducing computational power while model execution. Finally, the fourth contribution was real-time implementation of the proposed algorithm on a low-end edge device Jetson Nano. The proposed model's effectiveness was verified using qualitative and quantitative analysis on PASCAL-VOC 2007 test dataset and a comparative analysis was made with existing state-of-the-art pedestrian detection algorithms. Moreover, the proposed model was able to achieve competitive results on Caltech pedestrian dataset.

Therefore, the proposed model is lighter, faster, more efficient and achieves real-time detection performance on a low-end edge device while detecting smaller pedestrians in complex environmental road scenes. Although the proposed model achieves better detection accuracy while detecting small-scale pedestrians, it fails to detect occluded and denser pedestrians accurately. However, the detection speed needs to be improved further on low end edge device.

Chapter 6

EfficientLiteDet: A Real-time Pedestrian and Vehicle Detection Algorithm

This chapter focusses on design and development of a highly efficient, light-weight and fast multi-object detection algorithm to detect tiny and occluded objects and to operate in real-time on an edge device. Since there is a necessity for efficient and accurate pedestrian and vehicle detection algorithms based on vision sensors in real-time for autonomous driving system. Secondly, the proposed algorithm is implemented in real-time on a low-end edge device Jetson Tx2 board. To verify the effectiveness of the proposed model both quantitative and qualitative analysis on multiple autonomous datasets were obtained and made comparative analysis with the existing state-of-the-art object detection algorithms.

6.1 Overview

Recently, in the field of automobile industry there has been a great increase in technological innovation though the number of accidents has grown greatly due to various factors. So, to resolve this challenge, all automobile companies have been moving towards developing advanced unmanned or driver-assistance driving systems and they include many vision sensors and deploy complex algorithms. Since safety is top priority and plays a crucial role in vision-based systems, there is a need to develop accurate and real-time performance of detection algorithms. Therefore, great efforts have been devoted to improve both detection accuracy and speed in order to achieve real-time detection performance. The most representative algorithms for accurate detection of pedestrians and vehicles are Fast and Faster region-based convolutional neural network (Fast/Faster-RCNN) [11-12], SSD [19-21] and YOLO series [14-17]. However, these large scale DCNN based algorithms require high computational and memory resources and need a very large number of fine-tuning parameters.

To realize the real-time object detection, it requires powerful GPU (Graphics Processing Unit) computing power. However, it is difficult to deploy these heavy models on any low-end edge device due to limited computational power and memory constraints. Therefore, there is a need to develop a light-weight detection algorithm to operate on edge devices in real-world applications [136]. Many researchers have proposed lightweight algorithms based on deep learning in many fields such as Object detection [18], [81-83], [135], Pedestrian detection [43], [137] Vehicle detection [71], [73] etc.

Many researchers have already proposed light-weight object detection algorithms to achieve either higher detection accuracy or higher speed under limited hardware constraints. Various lightweight networks such as Tiny-YOLO series [14-17], MobileNet series [81-83] and ShuffleNet series [84-85] are available. Tiny-Yolo models are light-weight models of original YOLO series. Tiny-YOLOv2 model uses 9 convolutional layers out of 19 convolutional layers (DarkNet19) used in original YOLOv2 in order to reduce network complexity. Similarly, Tiny-YOLOv3 model uses 21 convolutional layers out of 53 convolutional layers (DarkNet53) used in original YOLOv3. Along the same lines, Tiny-YOLOv4 is a lightweight model of the original YOLOv4 and uses CSPDarkNet53-Tiny backbone network

Since safety is top priority and plays a crucial role in vision-based systems, there is a need to develop accurate and real-time performance of detection algorithms. Therefore, great efforts has been devoted to improve both detection accuracy and speed in order to achieve real-time detection performance. Directly applying existing models to tackle real-time pedestrian and vehicle detection tasks captured by high-speed moving vehicle scenarios poses two problems. First, the target scale varies drastically because the vehicle speed changes greatly. Second, captured images contain both tiny targets and high-density targets, which brings in occlusion between targets. Therefore, in this chapter, to solve these two issues, a real-time object detection algorithm is proposed which is referred to as EfficientLiteDet (Efficient Light -weight Detector.

Therefore, an efficient lightweight model is designed for detecting pedestrians and vehicles in real-time on a low-end edge device and the same model was verified by implementing it in real-time on Jetson Nano board.

The contributions of this work are summarized as follows:

- One more prediction head is inserted which can accurately detect multi-scale objects, especially tiny targets. The proposed model introduced Transformer prediction head (TPH) which can accurately detect occluded and denser targets.
- In order to focus only on particular targets, we introduced attention mechanism i.e. CBAM in our model.
- To improve the efficiency of the proposed model, mosaic, mix-up and traditional data augmentation techniques were employed during training.
- Experimental results show that when our model was deployed on Nvidia Jetson TX2 board, it runs with 35.1 FPS and achieves real-time detection performance.

6.2 Proposed Methodology

The framework of the proposed model is shown in Figure 6.1. So by modifying Tiny-YOLOv4 [17] and enable it for detecting pedestrians and vehicle targets. This model supports both feature reuse and propagation which in turn reduce both number of parameters and computational complexity. Therefore, it is optimal to select this model which works to achieve real-time performance. The proposed light-weight detection algorithm is composed of Modified CSPResNet50 backbone [136], spatial pyramid pooling network (SPPNet) [86], path aggregation network (PANet) [138] as neck, introduced transformer encoder [139] in both neck and at the backbone end and three-scale transformer prediction head as a head.

6.2.1 Extra Prediction Head for detecting small-targets

Through investigation it is clear that both Caltech [103] and Highway [108] datasets contain several extremely small instances, so in order to detect smaller targets accurately one more prediction head is introduced. Therefore, this three-head prediction head structure can ease the negative influence caused by violent object scale variance. As shown in Figure 6.1, the prediction head (head.1) added is generated from low-level, high-resolution feature map, which is highly sensitive to smaller targets. Although by introducing additional prediction head, both memory cost and computational cost increases, but the performance of the model while detecting smaller targets increases.

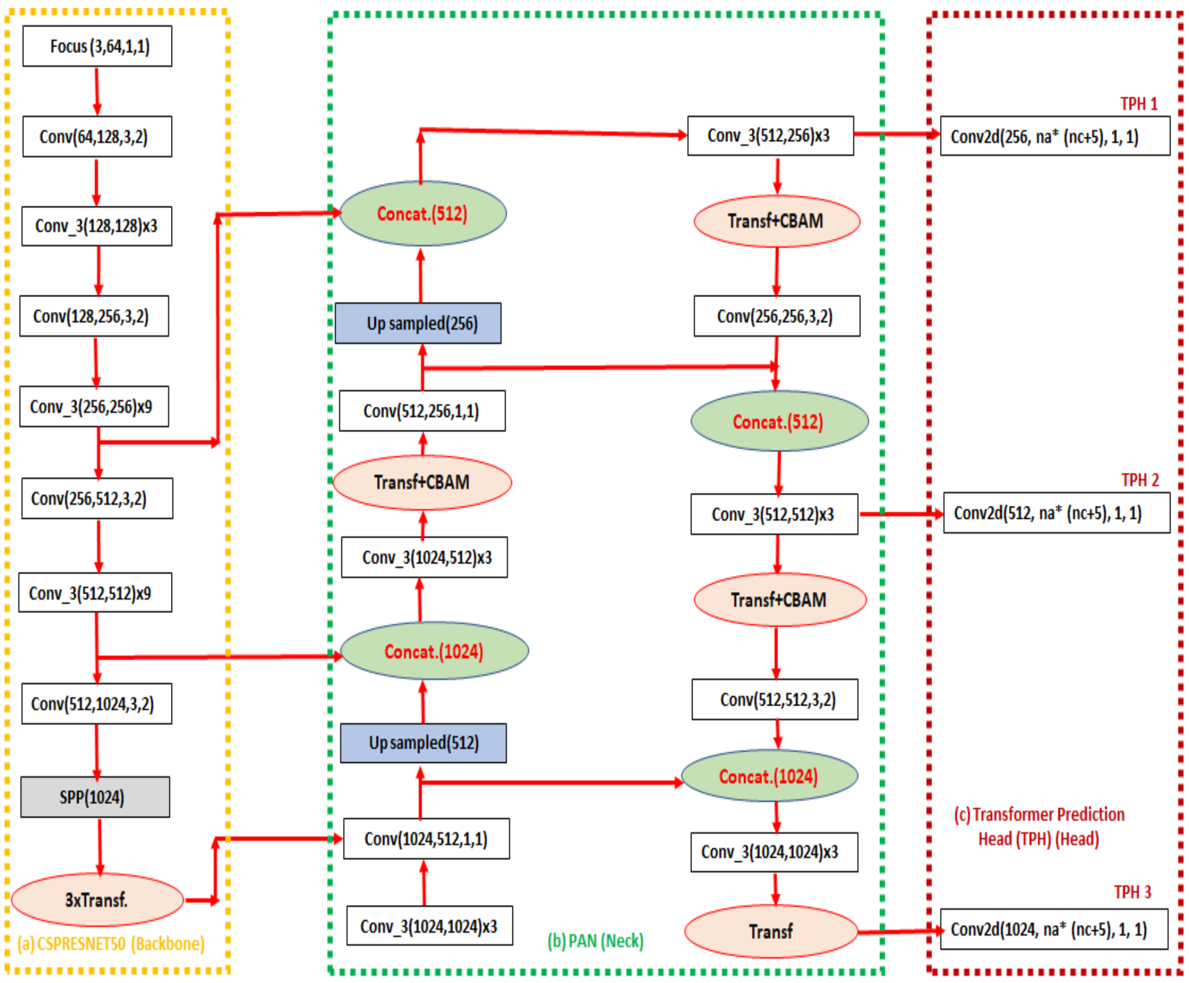


Figure 6.1 EfficientLiteDet Architecture (Proposed Algorithm).

6.2.2 Transformer Encoder Block

Inspired by transformer vision [139] some of the convolutional blocks and CSP bottleneck blocks in original version of YOLOv4 were replaced with transformer encoder blocks. The structure of transformer encoder block is shown in Figure 6.2. Compared to original bottleneck block in CSPDarknet53, we believe that transformer encoder block can capture global information and abundant contextual information. Each transformer encoder has two sub-layers, in which the first sub-layer is multi-head attention layer followed by Multi-layer Perception (MLP) a fully connected layer, and between sub-layers, residual connections were used. Transformer encoder blocks increase the ability to capture different local information. It also explores feature representation potential with self-attention mechanism [119]. On the Caltech dataset, transformer encoder blocks achieve better detection performance on occluded objects with high-density.

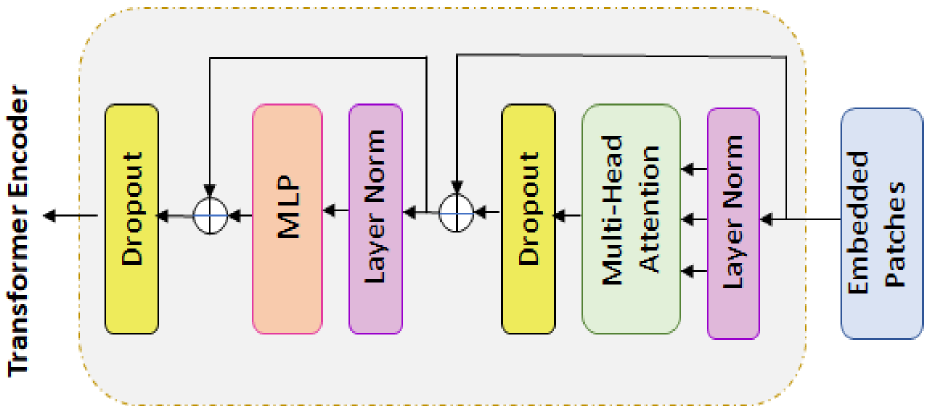


Figure 6.2 Transformer Encoder Architecture.

Based on Tiny-YOLOv4, transformer encoder blocks were applied in the head part to form Transformer Prediction Head (TPH) and at the end of backbone. This is because at the lower end of the network, the feature maps have low resolution. Applying TPH on low-resolution feature maps can decrease expensive computation and memory cost. Moreover, when the resolution of input images are enlarged, some of the TPH blocks are removed at the early layers to make the training process available.

6.2.3 Convolutional Block Attention Module (CBAM)

CBAM [140] is a simple and efficient attention module. Since it is a lightweight module, it can be easily integrated into most CNN architectures, and trained in end-to-end manner. Given a feature map, CBAM sequentially infers the attention map along two separate dimensions of channel and spatial, and then multiplies the attention map with the input feature map to perform adaptive feature refinement. Figure 6.3 shows the architecture of CBAM module. After integrating CBAM into different models on different classification and detection datasets, the performance of the model get vastly improved, which proves the effectiveness of this module. Using CBAM can extract the attention area to help EfficientLiteDet resist confusing information and focus only on useful targets.

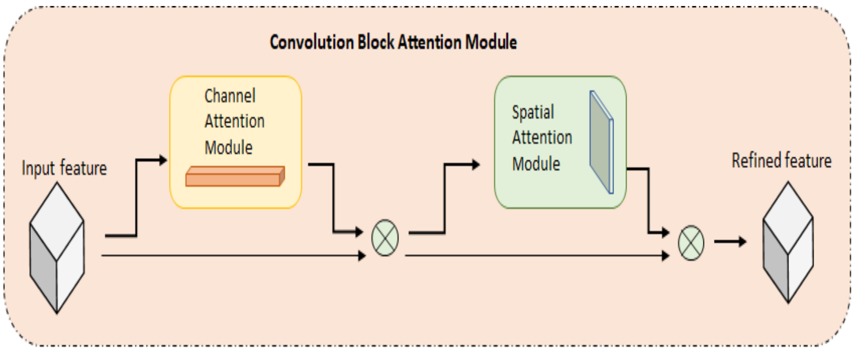


Figure 6.3 CBAM Architecture.

6.2.4 Multi-scale Testing

During the inference phase, initially multi-scale testing of the proposed model was performed. The implementation details of multi-scale-testing were performed in three steps. a) Scaling the testing image to 1.3 times. b) Reducing the image to 1 time, 0.83 times, and 0.67 times. c) Flipping the images horizontally, respectively. Both NMS [141] and Soft-NMS [142] exclude some boxes, while weighted box fusion (WBF) [143] merges all bounding boxes to form the final result. Therefore, it can tackle all inaccurate predictions of the proposed model. To ensemble the final model, WBF method was adopted as it performs much better than NMS. Finally, six different-scaling images were fed to EfficientLiteDet and WBF was used to fuse testing predictions.

6.2.5 Jetson TX2 Evaluation Board

Jetson Tx2 consumes just 7.5 watts and is known to be a power-efficient and fastest supercomputer employed in most AI autonomous machines and advanced robots. Figure 6.4 shows Nvidia Jetson TX2 evaluation board. The specifications of Jetson TX2 board are: 256 CUDA cores, Nvidia PASCAL GPU architecture, two 64-bit Denver CPU cores and quad A57 ARM Cortex CPUs, dual operating modes, inbuilt 8GB LPDDR4 RAM memory, supports a maximum of 32GB external memory, and has a maximum data transfer rate of 59.7 Gbps. It supports only Linux operating system, supports both Ethernet and WiFi connectivity, Bluetooth 4.1, video encoding and decoding 8K @60 FPS and delivers performance over 1.5 Tera FLOPS. So in terms of efficiency and performance in computer vision it is an optimal choice to deploy light-weight algorithms on low-cost edge device. Therefore, in order to test the real-time performance, the proposed algorithm was deployed on Jetson Tx2 evaluation board.

Figure 6.4 Nvidia Jetson TX2 Evaluation board.

6.3 Experimental Discussions

To showcase the robustness of EfficientLiteDet model, both quantitative and qualitative results analysis were obtained and a comparison was made with existing state-of-the-art object detection models. In order to show the effectiveness of EfficientLiteDet algorithm, the proposed model was evaluated on multiple popular pedestrian and vehicle datasets INRIA [102], Caltech [103], Udacity [107], Highway [108] and Pascal Voc-2007 [100].

6.3.1. Datasets

(i) INRIA: INRIA [100] pedestrian dataset contains 614 positive and 1218 negative images of which 1237 bounding-boxes were labelled for person class. Since it includes 1218 negative images without labels, INRIA dataset is very popular and has made laudable contributions to pedestrian detection. For testing 288 images were used to evaluate the model performance. The proposed model followed standard evaluation metrics, average precision (AP) and detection time were evaluated on test dataset and this was compared with several state-of-the-art (SOTA) models.

(ii) Caltech: The Caltech Pedestrian [103] dataset was introduced in 2012, and contains around ten hours of video captured with 600 x 400 resolution at 30 FPS. The dataset was collected from an urban traffic environment and contains about 250,000 frames of which 346,621 were bounding box labelled. It was composed of two subsets: set00 to set05 training images and set06-set10 testing images. As a rule of thumb, we extracted every 30th frame from the subsets and thus extracted 4024 training images and performed testing on 4024 images respectively. This dataset was unique compared to all other datasets, since around 50% of the annotated pedestrian instances were smaller than 50 pixels and this is rare in other datasets. The proposed model followed standard evaluation metrics, and evaluated average precision (AP) on pedestrian heights a) all pixels and b) 20-60 pixels and compared with SOTA models.

(iii) Udacity: The Udacity [107] dataset is a popular vehicle dataset which has videos captured on highway roads. It contains 404,916 and 5614 video frames for training and testing. Since the collected frames contain severe lighting changes, busy traffic, occlusion and sharp road curves, it is a challenging vehicle dataset. So every 30th frame was extracted from the video and created a dataset which possesses 10,797 images for training, 2700 images for validation while testing was performed on 5614 images. The proposed model followed standard evaluation metrics, and measured average precision (AP) and detection speed (FPS) on test dataset which was then compared with SOTA models.

(iv) Highway: The Highway [108] dataset is a captured video from the highway monitoring video of Hangzhou city. The images captured in this dataset cover the far distance of the highway and contain vehicles with wide changes in their scale. This dataset is challenging and robust due to the fact that it captured images from 23 surveillance cameras under circumstances which include, drastic changes in environmental conditions, different lighting conditions and at different times.

The captured video contains 11,129 video frames of 1920 x1080 resolution and 57,290 bounding boxes were labelled. The dataset annotated smaller objects in the vicinity and thus it contains smaller vehicles with wide scale changes. This dataset had 42.17% cars, 7.74% buses and 50.09% trucks and on an average 5.15 annotated instances, which was higher compared to KITTI [50] dataset. Out of 11,129 video frames, 70% (8792) images were used for training and 30% (2337) images used for testing. The proposed model followed standard evaluation metrics, and evaluated mean average precision (mAP) and detection speed (FPS) on test dataset which were then compared with several SOTA models.

(v) PASCAL VOC-2007: “Pattern Analysis, Statistical Modeling, and Computational Learning Visual Object Classes (PASCAL VOC)” 2007 [100] dataset consists of 20 different classes and 9963 labelled images. This dataset has complicated backgrounds, various sizes of human beings, high degree of occlusion, different costumes and various postures of human beings which would improve the trained model performance in order to meet complex real-time traffic scene. Of the 9963 labelled images, training set contains 5011 images while testing set contains 4972 images. The proposed model followed standard evaluation metrics and evaluated mean average precision (mAP) and detection speed (FPS) on test dataset which was then compared with several SOTA models.

The summary of the datasets used for experimentation on the proposed algorithm is shown in Table 6.1.

Table 6.1 Summary of datasets used in experimentation of EfficientLiteDet Model.

Datasets	Training Images	Testing Images
INRIA	614	228
Caltech Pedestrian	4250	4024
Udacity	13497	5614
Highway	8792	2337
Pascal Voc-2007	5011	4972

6.3.2. Experimental Setup

The experiments were carried out on a workstation during the training phase and finally, the testing phase was performed both on the workstation and Jetson Tx2 evaluation board. Figure 6.5 shows the experimental setup and captured real-time pedestrian detection on Jetson Nano evaluation board. The experimental environment during the proposed model training is: Google Colab, single Tesla V100 GPU, CUDA 10.0, CUDNN10.4 GPU acceleration library, Pytorch 1.2 deep learning framework.

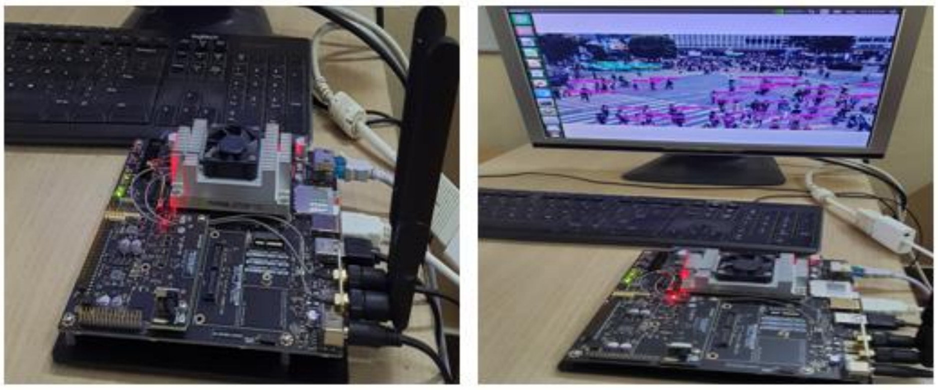


Figure 6.5 Experimental setup and captured results on Jetson Tx2 Evaluation board.

6.3.3. Training and Evaluation of Performance Parameters

The proposed model training settings are consistent from the baseline [7] and implemented on Pytorch framework. The proposed algorithm was trained for 300 epochs on all training datasets and SGD was the optimizer solution used during the training process. We used a learning rate of lr×Batchsize/64 (linear scaling), with an initial learning rate= 0.001 and cosine learning schedule. The weight decay was 0.0004 and SGD momentum was 0.9. The batch size was 16 by default. Other batch sizes which include single GPU training also work well. The input size was evenly drawn from 448 to 832 with 32 strides. FPS and latency were all measured with FP16-precision and batch=1 on a single Tesla V100.

The proposed model adopts the architecture of CSPResNet50 backbone and aggregation of spatial pooling pyramid, path aggregation and Transformer Encoder networks. Some of the training strategies were slightly modified compared to the original Tiny-Yolov4 [17], adding EMA weights updating, cosine learning schedule, and IoU loss. We use BCEwithLogit Loss for training class and object branch and IoU Loss for training reg branch. The general training tricks are orthogonal to improvement of EfficientLiteDet, and thus put them on the baseline. To boost the proposed model performance, both Mosaic and Mix-Up augmentation strategies were included while model training. Mosaic is an efficient augmentation strategy proposed by ultralytics-YOLOv5 [144]. It is then widely used in YOLOv4 [17] and other detectors [18]. Mix-Up was originally designed for image classification task but then modified in BoF [145] for object detection training. So, we adopted mix-up, random-horizontal-flip, color-jitter, multi-scale and mosaic data augmentation techniques in our model. When training neural network models for computer vision problems, data augmentation is a technique often used to improve performance and reduce generalization errors. When using a model to make predictions, image data augmentation on test dataset can also be applied to allow the model to make predictions on multiple versions of images. The prediction of the augmented images can be averaged to obtain better prediction performance. The test images were scaled to three different sizes in multi-scale-testing strategy, and then flipped horizontally, so that a total of six different images were obtained. After testing six different images and fusing the results, the final test result was obtained.

During the testing stage of any dataset on our network: (i) batch size was set to 1, (ii) load the best weight file into the model, (iii) then feed the test images path and (iv) validate the network performance in terms of evaluation metrics: Precision, Recall, Mean Average

Precision (mAP), thresholds ranging from [0.5: 0.95], Average Precision AP50 (IoU@0.5), Weight file (MB), Billion Floating Point Operations (BFLOPS) and detection speed (FPS) or time (milliseconds) on workstation were calculated. Our networks were tested only on benchmark datasets, still we left with KITTI and DETRAC datasets and not considered as they focus more on vehicles and widely used for 3D object detection. Yet our networks would produce satisfactory results on KITTI and DETRAC datasets.

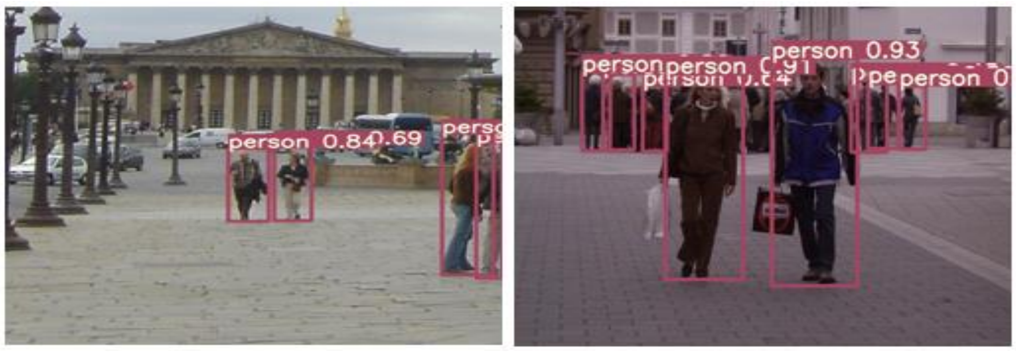
6.4 Results & Discussions

INRIA: The results of EfficientLiteDet model were compared with several SOTA models on INRIA test dataset, which include ACF [67], YOLOv2 [15], YOLOv3 [16], Tiny-YOLOv4 [17] and RSA-YOLO [56]. Table 6.2 shows the comparison of the proposed model with SOTA models evaluated in terms of average precision (AP) and detection speed parameters. From the table below, it is clear that our model achieves 89% AP which is +2.7% higher compared to Tiny-Yolov4 though it runs slower than Tiny-YOLOv4. Figure 6.6 shows sample detection results on INRIA test dataset.

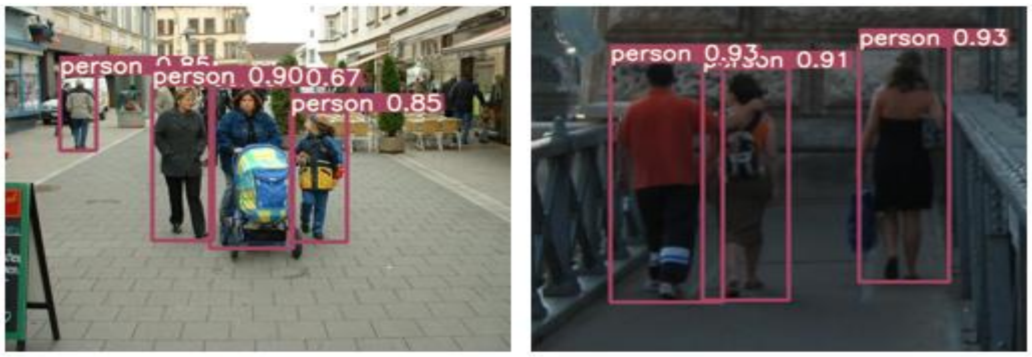
Table 6.2 Comparison experiment on INRIA pedestrian dataset.

Models	%AP (IoU@0.5)	Detection time (ms)
ACF [67]	83.17	65.9
ACF+CNN [67]	84.87	295.9
YoloV2 [15]	87.6	11.9
Yolov3 [16]	83.54	13.5
Improved Yolov3 [43]	90.42	9.6
Y-PD [62]	88.1	13.7
Tiny YOLOv4 [17]	86.3	**3.7**
YOLOv2PD [122]	93.2	**27.5**
Optimized SSD+MobileNet [137]	82.4	12.6
RSA-YOLO [56]	**94**	39
Proposed	89	7

(a) (b)



(c) (d)



(e) (f)

Figure 6.6 Detection results on INRIA Pedestrian dataset (544x544).

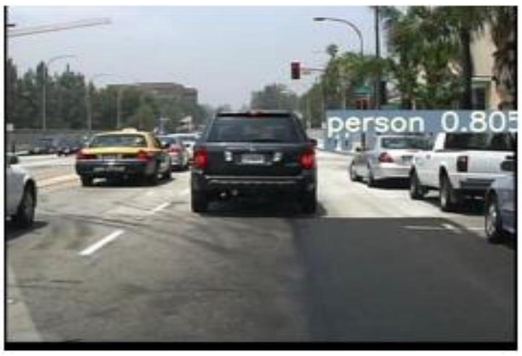
Caltech: Compared qualitative results of EfficientLiteDet model with several SOTA models on Caltech test dataset which includes Checkerboards+ [147], RPN+BF [47], TA-CNN [149], SDS-RCNN [60], SCN [150] and SA-RCNN [49]. Table 6.3 shows the proposed model analysis of precision-recall on Caltech pedestrian datasets for all pedestrian heights (all pixels) and small-scale pedestrian height (20-60 pixels).

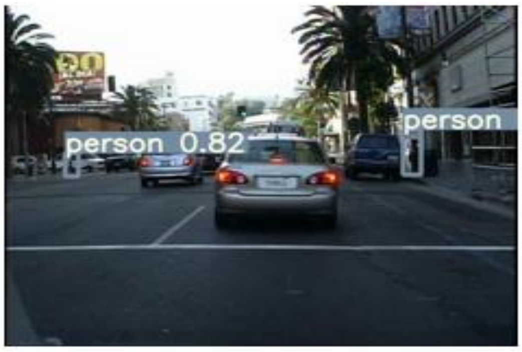
Table 6.3 Comparison experiment on Caltech Pedestrian dataset.

Models	%AP (All Pixels)	%AP (20-60 Pixels)	%AP @ (0.7)
CCF [146]	42.95	24.85	31.1
Checkboards+ [147]	42.22	24	30.1
LDCF [148]	38.54	21.67	27.2

RPN+BF [47]	44.49	24.35	32.4
TA-CNN [149]	37.38	18.64	25.1
SDS-RCNN [60]	47.38	28	35.5
UDN+ [50]	44.95	25.59	33.1
Y-PD [62]	45.26	18.4	32.1
SA-RCNN [49]	46.63	27.11	34.44
YOLOv2PD [122]	45.94	28.7	37.3
SCN [150]	**47.62**	29.46	36
Proposed	46.74	**32.57**	**38.2**

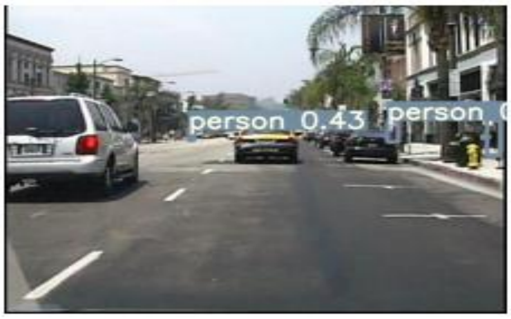
From Table 6.3 it is clear that EfficientLiteDet model achieves 32.57% AP, on pedestrian height (20-60 pixels) i.e. smaller pedestrians, which is comparatively better compared to other SOTA algorithms. But our model achieves 46.74% AP on pedestrian height (all pixels) and underperformed compared to SCN [150] model. Under IoU @0.7, the proposed model achieves 38.2% AP which is higher compared to Y-PD [62], SA-RCNN [49] and SCN [150]. Figure 6.7 shows detection results on Caltech test dataset. From the detection results, it is clear that our model detects smaller pedestrians more accurately and efficiently.

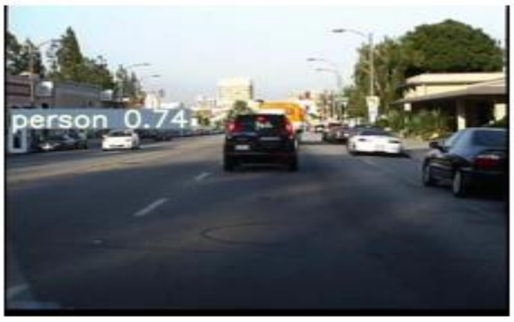




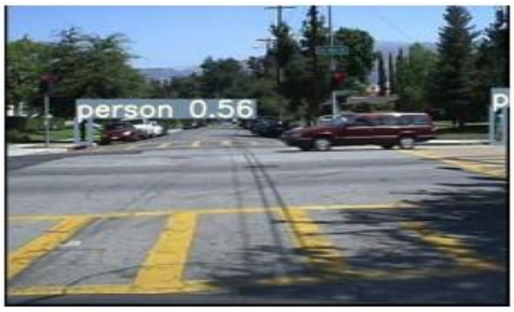
(a)

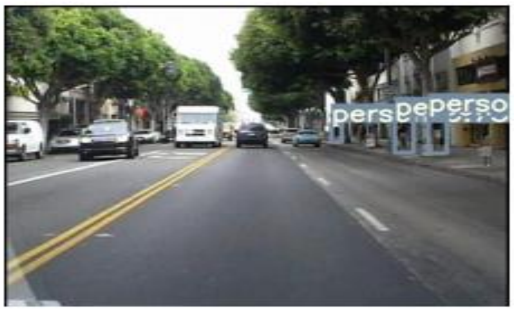
(b)





(c) (d)





(e) (f)

Figure 6.7 Detection results on Caltech Pedestrian dataset (544x544).

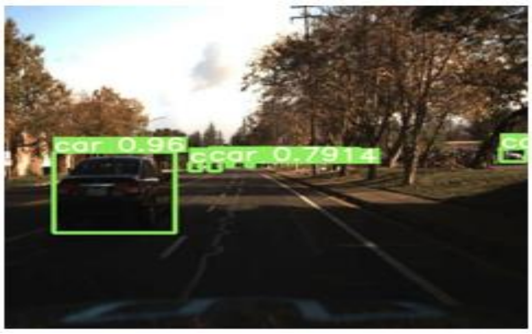
Udacity: The qualitative results of EfficientLiteDet model were compared with several SOTA models on Udacity test dataset, which include Faster-RCNN [11], SSD+MobileNet [81], Tiny-YOLOv2 [15], Tiny-YOLOv3 [16] and Tiny-YOLOv4 [17] algorithms. Table 6.4 shows the comparison of test results of the proposed model on SOTA models and it was evaluated in terms of average precision (0.5:0.95), mean average precision (mAP) and detection speed parameters.

Table 6.4 Comparison experiment on Udacity dataset.

Models	mAP @ 0.5 (%)	%AP@ (0.5:0.95)	Detection Speed (FPS)
Faster RCNN [12]	71.5	32.7	0.8
SSD+MobileNet (512x512) [81]	73.4	39.4	68
Tiny-Yolov2 (544x544) [15]	66.3	35.1	138
Tiny Yolov3 (416x416) [16]	68.5	38.6	202
Optimized MobileNet+SSD (544x544) [137]	74.6	41.3	155
Tiny Yolov4 (544x544) [17]	75.9	42	**270**
Proposed (544x544)	**77.8**	**44.6**	143

From the above results, it is clear that our model achieves 77.8% mAP value which is +9.3%, +3.2% and +1.9% higher compared to Tiny-YOLOv3 [16], Tiny-YOLOv4 [17] and Optimized MobileNet+SSD [137] SOTA models. Figure 6.8 shows sample detection results on Udacity test dataset.

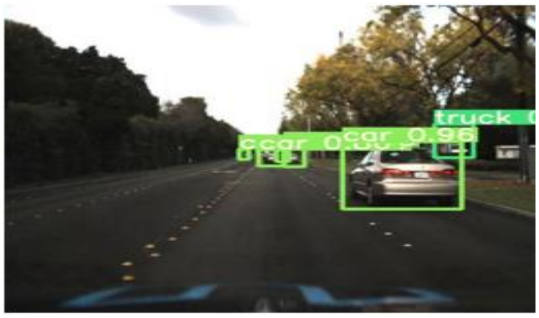
(a)

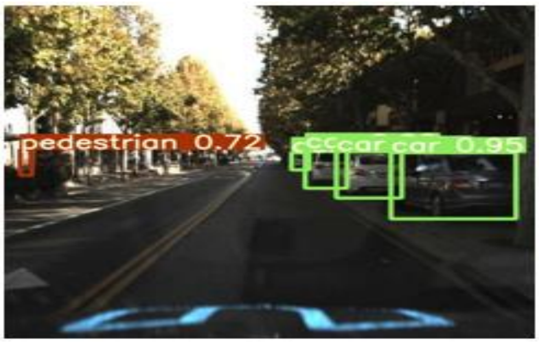


(b)

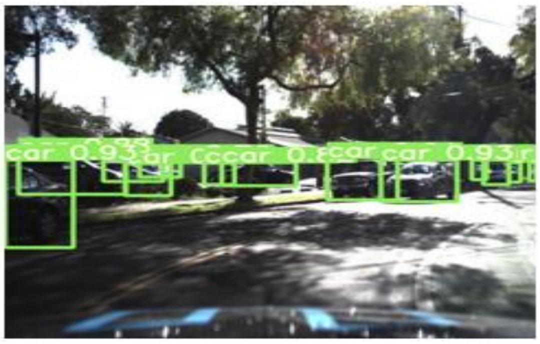
(c)

(d)





(e)



(f)

Figure 6.8 Detection results on Udacity dataset (544x544).

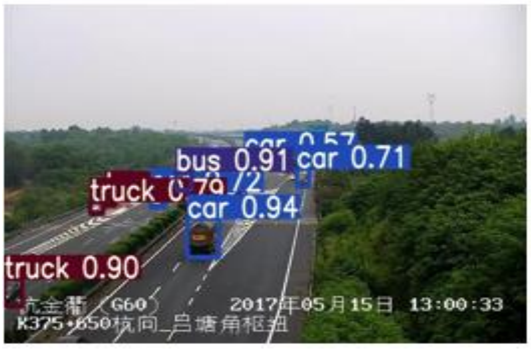
Highway: Compared the qualitative results of EfficientLiteDet model with several SOTA models on Highway vehicle test dataset (considered car, bus, truck classes) which include Faster-RCNN [12], Tiny-YOLOv4 [17] and Scaled Tiny-YOLOv4 [18] algorithms. Table 6.5 shows the comparison of test results of the proposed model on SOTA models and was evaluated in terms of recall, mean average precision (mAP) and detection speed parameters.

Table 6.5 Comparison experiment on Highway Vehicle dataset.

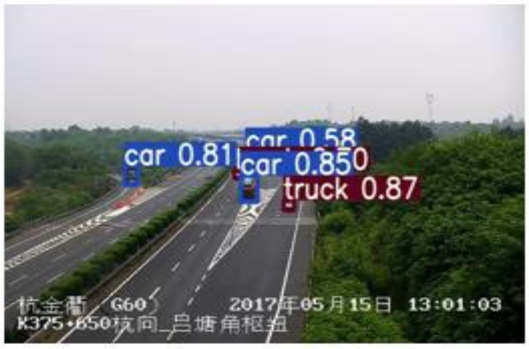
Models	Recall (%)	mAP@ 0.5 (%)	Detection Speed (FPS)
Faster RCNN [12]	77.2	55.6	5.3

Tiny-Yolov2 [15]	**89.4**	62.1	138
Tiny-Yolov3 [16]	81.3	68.4	202
Tiny Yolov4 [17]	87.5	76.2	270
Scaled Tiny-Yolov4 [18]	86.9	78.3	**371**
Proposed (544x544)	88	**80.1**	111

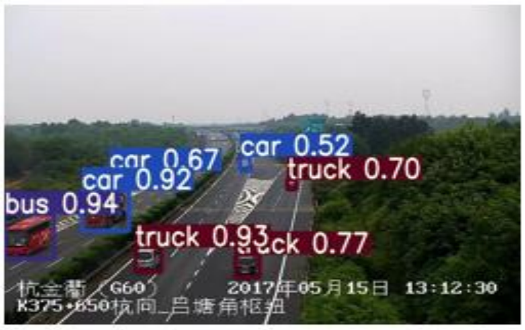
From the above results, it is clear that proposed model achieves 80.1% mAP value which is +11.7%, +3.9% and +1.8% higher compared to Tiny-YOLOv3 [16], Tiny-YOLOv4 [17], and Scaled Tiny-YOLOv4 [18] SOTA models, respectively. Figure 6.9 shows sample detection results on Highway vehicle test dataset.



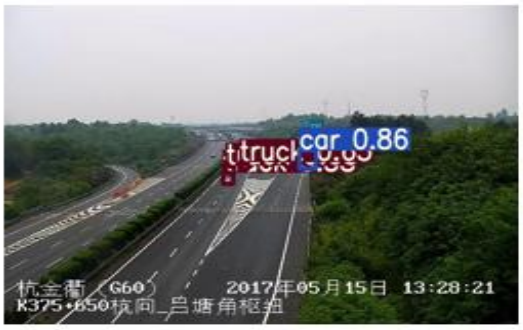
(a)



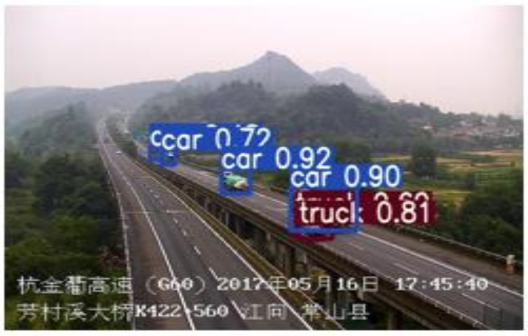
(b)

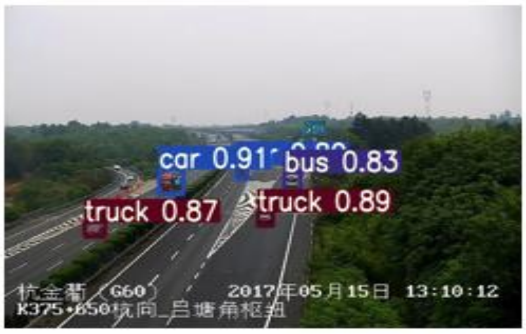


(c)



(d)





(e) (f)

Figure 6.9 Detection results on Highway Vehicle dataset (544x544).

PASCAL VOC-2007: Compared the qualitative results of EfficientLiteDet model with several SOTA models on PASCAL VOC-2007 test dataset (considered person, car, bus, bicycle and motorbike classes) which include Faster-RCNN [12], SSD+MobileNet [81], Tiny-YOLOv3 [16], Tiny-YOLOv4 [17], Pelee [135], Optimized MobileNet+SSD [137], and DLNetV2 [71] algorithms. Table 6.6 shows the comparison of test results of EfficientLiteDet model on several SOTA models, evaluated in terms of mean average precision (mAP), detection speed and model size parameters.

Table 6.6 Comparison experiment on PASCAL VOC-2007 test dataset.

Models	Model size (MB)	mAP@ 0.5 (%)	Detection Speed (FPS)
Faster RCNN [12]	522	70.4	0.5
SSD+MobileNet (300X300) [81]	5.77	71.6	61
SSDLite+MobileNetv2 (300x300) [81]	6.8	67	80
YOLO (416x416) [14]	75.3	57.9	45
Yolov2 (288X288) [15]	67.13	69	47
Yolov2 (544x544) [15]	203	73.4	38
Tiny-Yolov2 (416X416) [15]	42	63.88	210
Yolov3 (416x416) [16]	240	79.3	30
Yolov3 (544x544) [16]	247	81.6	17
Tiny Yolov3 (544x544) [16]	35	68.54	202
YOLOv2PD [122]	64	80.7	36.3
SSAD* (Proposed)	67.2	85.6	17

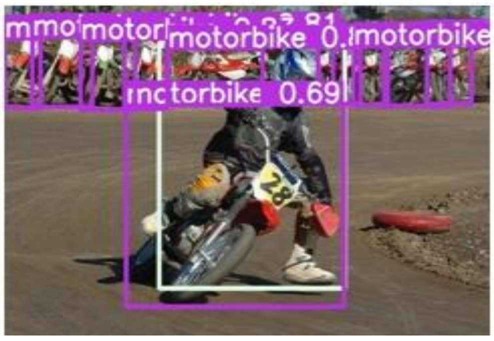
Optimized MobileNet+SSD [137]	23.6	80.04	155
Pelee (304X304) [135]	**5.43**	70.9	120
Improved Tiny-Yolov3 (544x544) [43]	13.9	73.98	207
DLNetV2(544x544) [71]	23.6	75.4	46
Lite-YOLO-SPP (416x416) [73]	14.2	77.44	177
Tiny Yolov4 (544x544) [17]	23.1	84.9	**270**
Proposed (416x416)	14.8	80.5	147
Proposed (544X544)	14.8	**87.3**	143

From the above results, it is clear that our model achieves 87.3% mAP value which is +18.76%, +7.26%, +16.4%, +13.32%, +11.9% and 6.8% higher compared to Tiny-YOLOv3 [16], Optimized MobileNet+SSD [137], Pelee [135], Improved Tiny-YOLOv3 [43], DLNetV2 [71] and Tiny-YOLOv4 [17] SOTA models respectively. The size of our model was 14.8 MB which is far smaller and more light-weight compared to Tiny-YOLOv3 [16], DLNetv2 [71] and Tiny-YOLOv4 [17] models. Figure 6.10 shows sample detection results on VOC-2007 test dataset. From these results, it is obvious that our light-weight EfficientLiteDet algorithm is able to detect smaller and occluded denser targets more accurately and efficiently.

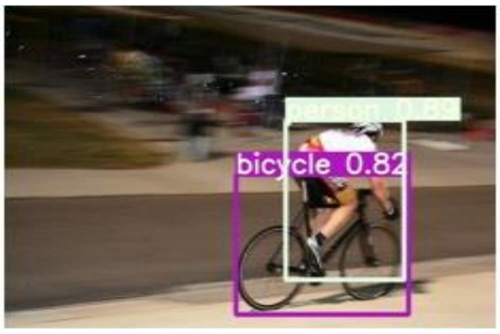
(a)

(b)

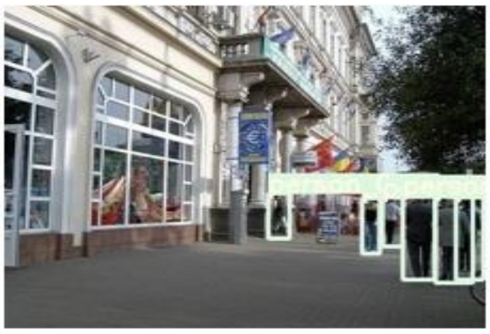
(c)



(d)



(e)



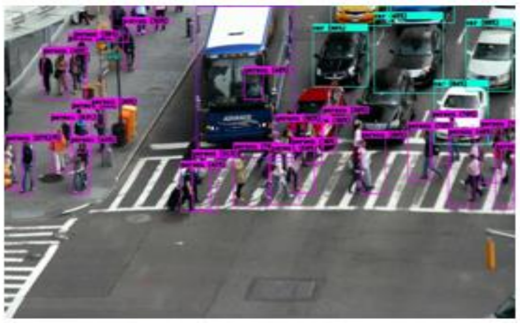
(f)

Figure 6.10 Detection results on PASCAL VOC-2007 test dataset (544x544).

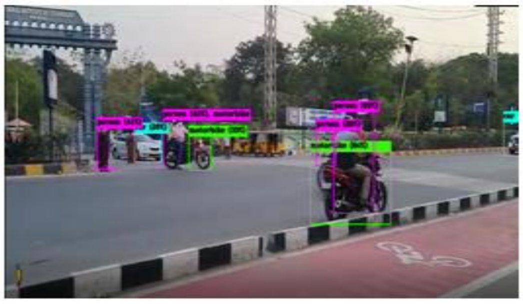
To test the robustness and show the real-time performance of the proposed lightweight model, the whole model was deployed on a low-end edge device, Nvidia Jetson TX2. Compared to Tesla V100 GPU, this board has limited computational power but it consumes very little energy. Figure 6.11 shows the experimental setup of Jetson TX2 board and the detection results on test video in real-time was verified.

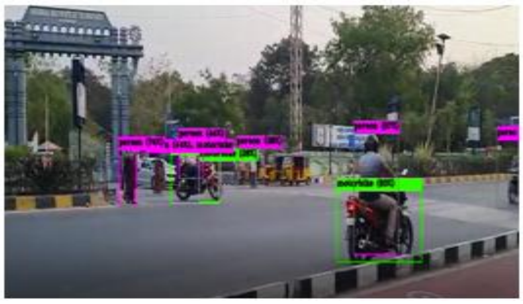
(a)

(b)



(c) (d)





(e) (f)

Figure 6.11 Real-time detection results on Jetson TX2 board on a test video.

During the testing phase of EfficientLiteDet model on Jetson TX2 board, runs with 35.1 FPS and achieve real-time detection performance. Figure 6.12 shows the comparison of the proposed model with several SOTA models during testing on Jetson TX2 board and it was evaluated in terms of speed (FPS), BFLOPS and weight file (MB) parameters. From Figure 6.12, it is clear that EfficientLiteDet model runs in real-time and generates smaller weight file compared to Tiny-YOLO state-of-the-art (SOTA) models.

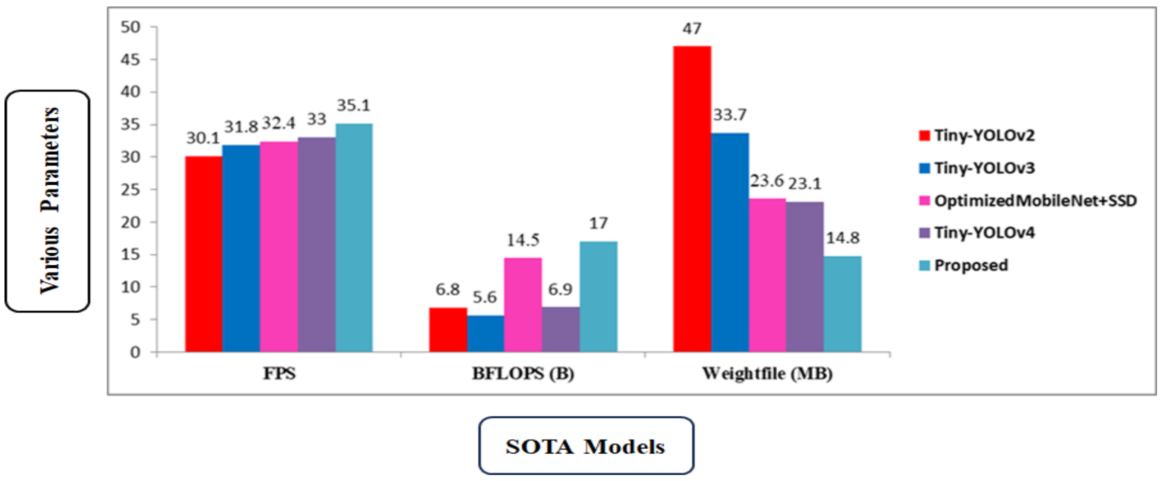


Figure 6.12 Comparison experiment of various SOTA models tested on Jetson TX2.

6.5 Summary

This chapter has proposed a faster, more efficient, accurate and light weight framework for real-time pedestrian detection for autonomous driving system. In the proposed model, best characteristics of ResNet, YOLOv4 and EfficientDet were adopted. In this work, four new contributions were presented. Our first contribution was one more prediction head is inserted which could accurately detect multi-scale objects especially tiny targets. Second contribution was the introduction of transformer encoder blocks in the neck part which helps to detect both occluded and denser targets accurately. The third contribution was CBAM attention module adopted in this model to focus only on particular targets i.e. pedestrians and vehicles. The last contribution was that to improve the efficiency of the proposed model, multi-scale testing was performed during the inference phase. The proposed algorithm was implemented in real-time on a low-end edge device Jetson TX2 board. Therefore, from the experimental results, it is clear that our EfficientLiteDet model is more efficient, faster, lightweight and real-time object detector, and it is more easily adaptable on low-end edge devices.

Chapter 7

Conclusions, Limitations and Future Work

7.1 Conclusions

Deep neural networks (DNNs) achieve tremendous performance across various domains ranging from industrial automation, intelligent transportation system, self-driving cars to embedded vision. However, DNNs are resource intensive, and therefore difficult to deploy on low-end edge devices which have limited computational capabilities and memory constraints. Therefore, to run DNNs efficiently on such low-end edge devices, neural networks that are light-weight, faster, more accurate and more power-efficient should be developed. The goal of this thesis is improving the efficiency of DNNs without losing detection accuracy while detecting multi-scale objects (pedestrians and vehicles) which are smaller, occluded and denser. The same DNNs are deployed on low end edge devices to achieve detection performance in near real-time.

In chapter 3, a robust real-time object detection algorithm was developed. A novel YOLOv2PD network was proposed for the accurate detection of smaller and more densely distributed pedestrians. The structure of YOLOv2PD was designed to improve the network's feature extraction ability by adopting MLFF strategy and, at the higher end, one repeated convolutional layer was removed. To improve detection accuracy, the loss function was improved by applying normalization. From the experimental results, this network achieved 80.7% AP, which is 2.1% higher than that of YOLOv2 Model on Pascal Voc-2007+2012 pedestrian test dataset. The robustness of YOLOv2PD network was validated on a real-time test video and ran with 36.3 FPS compared with SOTA YOLOv2 Model. Quantitative analysis shows that YOLOv2PD network achieved 7.8 average MR on INRIA and 0.381 AP on Caltech pedestrian test datasets.

In chapter 4, an accurate and simple one-shot SSD-based detector, termed SSAD, for effective pedestrian and vehicle detection for autonomous driving system was developed. Specifically, SSAD utilizes a fast and light-weight MAU to discover feature dependencies to extract only useful and relevant regions and suppress irrelevant regions. To improve the detection accuracy of occluded objects, a multi-scale attention unit (MAU) was embedded into SSAD network. Our network improves the detection accuracy of original SSD by a large margin at a small extra computational cost. Based on qualitative and quantitative result analysis, SSAD network achieves superior detection results on four challenging datasets with considerable differences in the image aspect ratio. In particular, it achieves better performance than other SOTA one-stage detectors such as YOLOv2PD, YOLOv3, and MSCM-Net.

In chapter 5, Optimized MobileNet+SSD was developed for detecting smaller pedestrians accurately in real-time and the same network was implemented effectively on low-end edge device Jetson Nano board. Our network lets the components work in coordination in such a manner that their strengths are improved and the number of parameters is decreased compared to SOTA detection architectures. To improve detection accuracy while detecting smaller pedestrians, a feature fusion concatenation module was introduced for injecting contextual information into SSAD network. The performance of optimized MobileNet+SSD was validated on PASCAL VOC-2007 test dataset and compared with state-of-the-art pedestrian detectors. When our model was deployed on low end edge device Jetson Nano, it ran with 34.01 FPS on a real-time test video.

In chapter 6, some cutting-edge techniques i.e. extra prediction head, transformer encoder block both in neck and backbone end, CBAM and some bag of tricks i.e. data augmentation techniques were adopted on Tiny-YOLOv4 baseline model. This led to light-weight pedestrian and vehicle detector referred to as EfficientLiteDet. Our network adopts the best characteristics of ResNet, YOLOv4 and EfficientDet. From the qualitative and quantitative results, it is clear that our network is particularly good at detecting multi-scale targets, smaller and occluded denser targets accurately and effectively in real-time. It achieves real-time inference on edge device Jetson TX2, which ensures that EfficientLiteDet model can be used in real-world scenarios. Since EfficientLiteDet model is very efficient, lightweight and real-time detector, it is more easily adaptable on low-end edge devices. The experimental results on five popular used pedestrian and vehicle datasets, demonstrated the robustness and effectiveness of EfficientLiteDet network.

In summary, this thesis takes important steps towards developing more efficient, ultra-light-weight and faster object detection algorithms for autonomous driving system.

7.2 Limitations & Future Work

In this section, limitations and the potential research direction for future research is presented. Although YOLOv2PD network achieves better detection accuracy while detecting smaller and densely distributed pedestrians in real urban environment scene, still there is a room for improvement in detecting occluded pedestrians and inference speed.

An efficient SSAD object detection algorithm was developed in order to achieve optimal trade-off balance between detection accuracy and speed while detecting small-scale and occluded objects. SSAD achieves excellent results by using feature fusion and attention mechanisms.

Since both YOLOv2PD and SSAD networks are heavier but achieves higher detection accuracy and runs with low inference speed so they are not quite suitable to deploy on edge devices.

So, light-weight algorithms were developed such as Optimized MobileNet+SSD and EfficientLiteDet and which are deployed on low-end edge devices. First an Optimized MobileNet+SSD was developed to detect smaller pedestrians in real-time without losing any detection accuracy where we incorporated a light-weight feature extractor network (MobileNet) as backbone. However, Optimized MobileNet+SSD network can detect only class i.e., pedestrians and fails to detect occluded pedestrians accurately.

Second a highly efficient, lighter and faster multi-object detection network i.e., EfficientLiteDet was developed to detect tiny and occluded objects and operates in real-time. In our network we adopted CSPResNet50 as a feature extractor and a light-weight CBAM attention module was employed to achieve our goals. However, EfficientLiteDet fails to detect heavily occluded pedestrians accurately and also fails when the test images contain heavy rain or dense haze.

The performance of CNN depends on lot of features we use and combine them effectively. But some of the universal features (referred as “Bag of Freebies”-BOF) such as Batch normalization, Weights normalization, Layer normalization, Self-adversial training (SAT), Mosaic data augmentation, Cross-Stage-partial-connections (CSP), DropBlock regularization, Optimal hyperparameters, Random training shapes, Cosine Annealing Scheduler, Weighted Residual Connections (WRC) and Data augmentation during training and testing were applied in most of the SOTA networks, tasks and datasets. Therefore, by combing above features more in a network which would improve the network training without impacting inference speed. By incorporating universal features such as Mish activation, Spatial Pyramidal Pooling (SPP), Path Aggregation Network (PANet), Transformer Encoder (TE), Attention modules (SE, CBAM) and DIoU NMS (referred as "Bag of Specials" -BoS) in our network would improve the detection accuracy of specific targets but at the cost of little increase in inference speed. Therefore, BoF and BoS would boost the detection accuracy and inference speed of the network.

The efficient networks introduced in this dissertation were manually designed i.e., fine tuning of network parameters. Neural architecture search (NAS) allows us to design hardware specific efficient neural networks automatically. All the networks designed in this dissertation adopt pre-trained backbone model for feature extraction. But for specific domain application while detecting only pedestrians and vehicles, an efficient light weight backbone model can be developed with fewer parameters and fewer FLOPS. Therefore, the developed network can achieve more than 100 FPS.

However, the networks developed can be adopted to cross domains for detecting various objects.

I hope that the steps taken in this dissertation would work towards designing efficient DNNs, alongside future research in the area.

Bibliography

[1] Technology Trends, https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2020.

[2] Li Jianguo and Yimin Zhang. "Learning surf cascade for fast and accurate object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3468-3475. 2013.

[3] Papageorgiou, Constantine and Tomaso Poggio. "A trainable system for object detection." International journal of computer vision 38, no. 1, 15-33, 2000.

[4] Lowe, David G. "Object recognition from local scale-invariant features." In Proceedings of the seventh IEEE international conference on computer vision, vol. 2, pp. 1150-1157. Ieee, 1999.

[5] Belongie Serge, Jitendra Malik and Jan Puzicha. "Shape matching and object recognition using shape contexts." IEEE transactions on pattern analysis and machine intelligence 24, no. 4, 509-522, 2002.

[6] Felzenszwalb Pedro F, Ross B. Girshick and David McAllester. "Cascade object detection with deformable part models." In 2010 IEEE Computer society conference on computer vision and pattern recognition, pp. 2241-2248. Ieee, 2010.

[7] Dalal Navneet and Bill Triggs. "Histograms of oriented gradients for human detection." In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 1, pp. 886-893. Ieee, 2005.

[8] Heikkilä Marko, Matti Pietikäinen and Cordelia Schmid. "Description of interest regions with local binary patterns." Pattern recognition 42, no. 3, 425-436, 2009.

[9] Murthy Chinthakindi Balaram, Mohammad Farukh Hashmi, Neeraj Dhanraj Bokde and Zong Woo Geem. "Investigations of object detection in images/videos using various deep learning techniques and embedded platforms-A comprehensive review." Applied sciences 10, no. 9, 3280, 2020.

[10] Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580-587. 2014.

[11] Girshick Ross. "Fast r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.

[12] Ren Shaoqing, Kaiming He, Ross Girshick and Jian Sun. "Faster RCNN: Towards real-time object detection with region proposal networks." Advances in neural information processing systems 28, 2015.

[13] He Kaiming, Georgia Gkioxari, Piotr Dollar and Ross Girshick. "Mask r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 2961-2969. 2017.

[14] Redmon Joseph, Santosh Divvala, Ross Girshick and Ali Farhadi. "You only look once: Unified, real-time object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788. 2016.

[15] Redmon Joseph and Ali Farhadi. "YOLO9000: better, faster, stronger." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263-7271. 2017.

[16] Redmon Joseph and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv: 1804.02767, 2018.

[17] Bochkovskiy Alexey, Chien-Yao Wang and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection." arXiv preprint arXiv: 2004.10934, 2020.

[18] Wang Chien-Yao, Alexey Bochkovskiy and Hong-Yuan Mark Liao. "Scaled-yolov4: Scaling cross stage partial network." In Proceedings of the IEEE/cvf conference on computer vision and pattern recognition, pp. 13029-13038. 2021.

[19] Liu Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "Ssd: Single shot multibox detector." In European conference on computer vision, pp. 21-37. Springer, Cham, 2016.

[20] Li Zuoxin and Fuqiang Zhou. "FSSD: feature fusion single shot multibox detector." arXiv preprint arXiv: 1712.00960, 2017.

[21] Fu Cheng-Yang, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. "Dssd: Deconvolutional single shot detector." arXiv preprint arXiv: 1701.06659,2017.

[22] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." In Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001, vol. 1, pp. I-I. Ieee, 2001.

[23] Viola Paul and Michael J. Jones. "Robust real-time face detection." International journal of computer vision 57, no. 2, 137-154, 2004.

[24] Lowe David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60, no. 2, 91-110, 2004.

[25] Felzenszwalb Pedro F, Ross B. Girshick, David McAllester, and Deva Ramanan. "Object detection with discriminatively trained part-based models." IEEE transactions on pattern analysis and machine intelligence 32, no. 9, 1627-1645, 2010.

[26] Girshick, R.B, Felzenszwalb P.F, Mcallester D.A. “Object detection with grammar models”. In Advances in Neural Information Processing Systems, Curran Associates Inc., San Francisco, CA, USA, pp. 442–450, 2011.

[27] Girshick R.B. “From Rigid Templates to Grammars: Object Detection with Structured Models”. Ph.D. Thesis, The University of Chicago, Chicago, IL, USA, 2012.

[28] Uijlings J.R., Van De Sande K.E, Gevers T, Smeulders A.W. “Selective search for object recognition”. Int. J. Comput. Vis. 104, 154–171, 2013.

[29] Krizhevsky A, Sutskever I, Hinton G.E. “Imagenet classification with deep convolutional neural networks”. In Advances in Neural Information Processing Systems, Curran Associates, Inc., San Diego, CA, USA, pp. 1097–1105, 2012.

[30] Abe, Shigeo. “Support vector machines for pattern classification”. Vol. 2. London: Springer, 2005.

[31] Zeiler M.D, Fergus R. “Visualizing and understanding convolutional networks”. In Proceedings of the European Conference on Computer Vision, Nancy, France, 14–18 September 2014; Springer: Berlin, Germany, pp. 818–833, 2014.

[32] Wu B, Iandola F, Jin P.H, Keutzer K. “SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving”. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26, pp. 129–137, 2017.

[33] Iandola F.N, Han, S, Moskewicz M.W, Ashraf K, Dally W.J, Keutzer K. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size”. arXiv:1602.07360, 2016.

[34] Law H, Deng J, “ Cornernet: Detecting objects as paired keypoints”. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 pp. 734–750, 2018.

[35] Duan K, Bai S, Xie L, Qi H, Huang Q, Tian Q, “Centernet: Object detection with keypoint triplets”, arXiv:1904.08189, 2019.

[36] Dollar P, Tu Z, Perona P, Belongie S “Integral Channel Features”, BMVC Press: London, UK, 2009.

[37] Maji S, Berg A.C, Malik J. “Classification using intersection kernel support vector machines is efficient”. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28, pp. 1–8, 2008.

[38] Zhu Q, Yeh M.C, Cheng K.T, Avidan S. “Fast human detection using a cascade of histograms of oriented gradients”. In Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22, Volume 2, pp. 1491–1498, 2006.

[39] P. Dollar, R. Appel, S. Belongie and P. Perona, "Fast Feature Pyramids for Object Detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532-1545, doi: 10.1109/TPAMI.2014.2300479, 2014.

[40] Mohan A, Papageorgiou C, Poggio T. “Example-based object detection in images by components”. IEEE Trans. Pattern Anal. Mach. Intell. 23, 349–361, 2001.

[41] Sadeghi M.A, Forsyth. D. “30hz object detection with dpm v5”. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin, Germany, pp. 65–79, 2014.

[42] Hosang J, Omran M, Benenson R, Schiele B. “Taking a deeper look at pedestrians”. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12, pp. 4073–4082, 2015.

[43] Yi Z, Yongliang S, Jun Z. “An improved tiny-yolov3 pedestrian detection algorithm”. Optik 2019, 183, 17–23.

[44] R. Benenson, M. Mathias, R. Timofte and L. Van Gool, “Pedestrian detection at 100 frames per second,” in IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, pp. 2903–2910, 2012.

[45] S. Paisitkriangkrai, C. Shen and A. Van Den Hengel, “Strengthening the effectiveness of pedestrian detection with spatially pooled features,” in European Conference on Computer Vision, Springer, pp. 546–561, 2014.

[46] C. Wojek, S. Walk and B. Schiele, “Multi-cue onboard pedestrian detection,” in IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 794–801, 2009.

[47] L. Zhang, L. Lin, X. Liang and K. He, “Is faster R-CNN doing well for pedestrian detection?” in Proceedings of the European Conference on Computer Vision, Springer, Cham, pp. 443–457, 2016.

[48] Pal, Mahesh. "Random forest classifier for remote sensing classification." International journal of remote sensing 26, no. 1 (2005): 217-222.

[49] J. Li, X. Liang, S. Shen, T. Xu, J. Feng et al., “Scale-aware fast R-CNN for pedestrian detection,” IEEE Transactions on Multimedia, vol. 20, no. 4, pp. 985–996, 2018.

[50] W. Ouyang, H. Zhou, H. Li, Q. Li, J. Yan et al., “Jointly learning deep features, deformable parts, occlusion and classification for pedestrian detection,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 8, pp. 1874–1887, 2018.

[51] Y. Pang, J. Xie, M. H. Khan, R. M. Anwer, F. S. Khan et al., “Mask-guided attention network for occluded pedestrian detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), pp. 4966–4974, 2019.

[52] S. Zhang, J. Yang and B. Schiele, “Occluded pedestrian detection through guided attention in CNNs,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, pp. 6995–7003, 2018.

[53] T. Song, L. Sun, D. Xie, H. Sun and S. Pu, “Small-scale pedestrian detection based on somatic topology localization and temporal feature aggregation,” arXiv preprint arXiv: 1807.01438, 2018.

[54] Y. Zhang, Y. Bai, M. Ding, S. Xu and B. Ghanem, "KGSNet: Key-Point-Guided Super-Resolution Network for Pedestrian Detection in the Wild," in IEEE Transactions on Neural Networks and Learning Systems. vol. 29, pp. 1-15, 2020.

[55] C. Lin, J. Lu, G. Wang and J. Zhou, "Graininess-Aware Deep Feature Learning for Robust Pedestrian Detection," in IEEE Transactions on Image Processing, vol. 29, pp. 3820-3834, 2020.

[56] W. Y. Hsu and W. Y. Lin, "Ratio-and-Scale-Aware YOLO for Pedestrian Detection," in IEEE Transactions on Image Processing, vol. 30, pp. 934-947, 2021.

[57] B. Han, Y. Wang, Z. Yang and X. Gao, “Small-scale pedestrian detection based on deep neural network,” in IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 7, pp. 3046–3055, July 2020.

[58] Yang Fan, Houjin Chen, Jupeng Li, Feng Li, Lei Wang, and Xiaomiao Yan. "Single shot multibox detector with kalman filter for online pedestrian detection in video." IEEE Access 7, 15478-15488, 2019.

[59] Cheng Yongren, Changhong Chen and Zongliang Gan. "Enhanced single shot multibox detector for pedestrian detection." In Proceedings of the 3rd International Conference on Computer Science and Application Engineering, pp. 1-7. 2019.

[60] G. Brazil, X. Yin and X. Liu, "Illuminating pedestrians via simultaneous detection & segmentation," in Proceedings of the IEEE International Conference on Computer Vision, pp. 4950–4959, 2017.

[61] Z. Cai, M. Saberian and N. Vasconcelos, "Learning complexity-aware cascades for deep pedestrian detection," in IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, pp. 3361–3369, 2015.

[62] Z.Liu, Z.Chen, Z. Li and W. Hu, "An efficient pedestrian detection method based on YOLOv2." Mathematical Problems in Engineering, vol. 1, 2018.

[63] X. Du, M. El-Khamy, V. I. Morariu, J. Lee and L. Davis, "Fused deep neural networks for efficient pedestrian detection," arXiv preprint arXiv: 1805.08688, 2018.

[64] Cao Jingwei, Chuanxue Song, Silun Peng, Shixin Song, Xu Zhang, Yulong Shao, and Feng Xiao. "Pedestrian detection algorithm for intelligent vehicles in complex scenarios." *Sensors* 20, no. 13, 3646, 2020.

[65] Ma Jun, Honglin Wan, Junxia Wang, Hao Xia, and Chengjie Bai. "An improved one-stage pedestrian detection method based on multi-scale attention feature extraction." Journal of Real-Time Image Processing 18, no. 6, 1965-1978, 2021.

[66] P. Viola, M.J. Jones, D. Snow, "Detecting pedestrians using patterns of motion and appearance", Int. J. Comput. Vision 63 (2), 153–161, 2005.

[67] Mateus A, Ribeiro D, Miraldo P & Nascimento J. C. "Efficient and robust pedestrian detection using deep learning for human-aware navigation". Robotics and Autonomous Systems, 113, 23- 37, 2019.

[68] Wu, Zhongyuan, Jun Sang, Qian Zhang, Hong Xiang, Bin Cai, and Xiaofeng Xia. "Multi-scale vehicle detection for foreground-background class imbalance with improved YOLOv2." *Sensors* 19, no. 15 (2019): 3336.

[69] Hasan, Irtiza, Shengcai Liao, Jinpeng Li, Saad Ullah Akram, and Ling Shao. "Pedestrian Detection: Domain Generalization, CNNs, Transformers and Beyond." arXiv preprint arXiv: 2201.03176, 2022.

[70] X. Wang, S. Wang, J. Cao and Y. Wang, "Data-Driven Based Tiny-YOLOv3 Method for Front Vehicle Detection Inducing SPP-Net," in IEEE Access, vol. 8, pp. 110227-110236, 2020, doi: 10.1109/ACCESS.2020.3001279.

[71] Chen L, Ding Q, Zou Q, Chen Z, & Li L. "DenseLightNet: a light-weight vehicle detection network for autonomous driving". IEEE Transactions on Industrial Electronics, 67(12), 10600-10609, 2020.

[72] Liu Wei, Shengcai Liao and Weidong Hu, “Towards accurate tiny vehicle detection in complex scenes”, Neurocomputing 347, 24-33, 2019.

[73] Rani, Esther “LittleYOLO-SPP: A delicate real-time vehicle detection algorithm”, Optik 225, 165818, 2021.

[74] Hu Xiaowei, Xuemiao Xu, Yongjie Xiao, Hao Chen, Shengfeng He, Jing Qin, and Pheng-Ann Heng. "SINet: A scale-insensitive convolutional neural network for fast vehicle detection." IEEE transactions on intelligent transportation systems 20, no. 3, 1010-1019, 2018.

[75] Nguyen, Hoanh, “Improving faster R-CNN framework for fast vehicle detection”, Mathematical Problems in Engineering 2019.

[76] Dai, Xuerui. "HybridNet: A fast vehicle detection system for autonomous driving." Signal Processing: Image Communication 70, 79-88.

[77] S. Zhang, R. Benenson, M. Omran, J. Hosang and B. Schiele, “Towards reaching human performance in pedestrian detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 973–986, 2018.

[78] P. Dollar, S. Belongie and P. Perona, “The fastest pedestrian detector in the west,” in Proceedings of the British Machine Vision Conference, BMVA Press, pp. 1–11, 2010.

[79] K. He, X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition,” IEEE in Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, pp. 770–778, 2016.

[80] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, “Densely connected convolutional networks,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 2261–2269, 2017.

[81] Howard A.G, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. “MobileNets: efficient Convolutional Neural Networks for Mobile Vision Applications”. arXiv 2017,arXiv:1704.04861.

[82] Sandler M, Howard A, Zhu M, Zhmoginov A & Chen, L. C. “Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520, 2018.

[83] Howard A, Sandler M, Chen B, et al. “Searching for MobileNetV3”, 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South) pp. 1314-1324, doi: 10.1109/ICCV.2019.00140, 2019.

[84] Zhou X, Lin M, et al. “ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices”. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Salt Lake City, UT, USA, 2018:6848-6856, 2018.

[85] Ma Ningning, Xiangyu Zhang, Hai-Tao Zheng and Jian Sun. "Shufflenet v2: Practical guidelines for efficient cnn architecture design." In Proceedings of the European conference on computer vision (ECCV), pp. 116-131. 2018.

[86] K. He, X. Zhang, S. Ren and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1904–1916, 2015.

[87] Simonyan, Karen and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv: 1409.1556, 2014.

[88] Szegedy Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

[89] J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

[90] Kanungo Tapas, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. "An efficient k-means clustering algorithm: Analysis and implementation." IEEE transactions on pattern analysis and machine intelligence 24, no. 7, 881-892, 2002.

[91] Ma J, Wan H, Wang J, Xia H, Bai, C, “An improved scheme of deep dilated feature extraction on pedestrian detection”. SIViP, 2020.

[92] J. Nie, “Design of visual feature detection system for intelligent driving of electric vehicle,” in Proceedings of the International Conference on Robots & Intelligent System (ICRIS), ACM, Amsterdam, Netherlands, February 2018.

[93] E. Sun, A. Nieto, and Z. Li, “GPS and Google Earth based 3D assisted driving system for trucks in surface mines, International Journal of Mining Science and Technology, vol. 20, pp. 138–142, 2016.

[94] Y.S. Chen, S.C. Chiu and S.S. Hsiau, “Safe technology with a novel rear collision avoidance system of vehicles,” International Journal of Automotive Technology, vol. 20, no. 4, pp. 693–699, 2019.

[95] F. K. Zhang, Y. Feng, and L. I. Ce, “Fast vehicle detection method based on improved YOLOv3,” Computer Engineering and Applications, vol. 3, pp. 133–139, 2019.

[96] Mukhtar A, Xia L, Tang TB, “Vehicle detection techniques for collision avoidance systems: a review. IEEE Trans Intel Transport Syst 16.5:2318–2338, 2015.

[97] X. Le, L. Xiang and Z. Hua, “Vehicle Detection on Road Frontage Based on SSD,” Software Guide, vol. 5, pp. 27–36, 2019.

[98] Lin T. Y, Dollar P, Girshick R, He K, Hariharan B & Belongie S “Feature pyramid networks for object detection”. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117-2125, 2017.

[99] Itti L, Koch C, Niebur E, “A model of saliency-based visual attention for rapid scene analysis”. IEEE TPAMI 20 (11), 1254–1259, 1999.

[100] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, Jan. 2015.

[101] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft coco: Common objects in context. In European Conference on Computer Vision, pages 740–755. Springer, 2014.

[102] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05), San Diego, CA, USA, pp. 886–893, 2005.

[103] P. Dollar, C. Wojek, B. Schiele and P. Perona, "Pedestrian detection: a benchmark," in IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 304–311, 2009.

[104] Zhang, S., Benenson, R., & Schiele, B.: CityPersonss: a diverse dataset for pedestrian detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3213–3221, 2017.

[105] Ess A, Leibe B, Gool L.V, "Depth and appearance for mobile scene analysis". In: IEEE International Conference on Computer Vision (ICCV), 2007.

[106] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in CVPR, 2012.

[107] Udacity. 2018. https://github.com/udacity/self-driving-car/. Accessed: 20-09-2021.

[108] Song H, Liang H, Li H, Dai Z & Yun X. "Vision-based vehicle detection and counting system using deep learning in highway scenes". European Transport Research Review, 11(1), 1-16, 2019.

[109] L. Wen, D. Du, Z. Cai, Z. Lei, M.C. Chang, H. Qi, J. Lim, M.-H. Yang, S. Lyu, in: UA-DETRAC: a new benchmark and protocol for multi-object detection and tracking, 2015

[110] Itti, L, Koch C, Niebur E, 1998, "A model of saliency-based visual attention for rapid scene analysis". IEEE TPAMI 20 (11), 1254–1259.

[111] Mnih, V, Heess N, Graves A, et al. "Recurrent models of visual attention", In: NIPS, pp. 2204–2212, 2014.

[112] Yoo D, Park S, Lee J.-Y, Paek A.S, So Kweon, I. "Attentionnet: Aggregating weak directions for accurate object detection", In: CVPR, pp. 2659–2667, 2015.

[113] Fu J, Zheng H, Mei T, "Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition", In: CVPR, pp. 3, 2015.

[114] Hu H, Gu J, Zhang Z, Dai J, Wei Y, "Relation Networks for Object Detection", In: CVPR, 2018.

[115] Cheng J, Dong L, Lapata M, "Long short-term memory-networks for machine reading", In: EMNLP, 2016.

[116] Zhang Y, Yi P, Zhou D, Yang X, Zhang Q, Wei P, "CSANet: channel and spatial mixed attention CNN for pedestrian detection". IEEE Access 8, 76243–76252, 2020.

[117] Jianan Li, Yunchao Wei, Xiaodan Liang, Jian Dong, Tingfa Xu, Jiashi Feng and Shuicheng Yan. "Attentive contexts for object detection". IEEE TMM, 19(5):944– 954, 2017.

[118] Lin Z, Feng M, Santos, C.N.d, Yu M, Xiang B, Zhou B, Bengio Y, "A structured self-attentive sentence embedding", In: ICLR, 2017.

[119] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A.N, Kaiser Ł,, Polosukhin I. "Attention is all you need", In: NIPS, pp. 5998–6008, 2017.

[120] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L, Lerer A, "Automatic differentiation in PyTorch", In: NIPS-W, 2017.

[121] Zhang S, Wen L, Bian X, Lei Z, Li S.Z, "Single-shot refinement neural network for object detection", In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4203–4212, 2018.

[122] Chintakindi Balaram Murthy, Mohammad Farukh Hashmi, Ghulam Muhammad, and Salman A. AlQahtani, "YOLOv2PD: An Efficient Pedestrian Detection Algorithm Using Improved YOLOv2 Model" Computers, Materials and Continua, 2021.

[123] W. Nam, P. Dollar and J. H. Han, "Local decorrelation for improved pedestrian detection," in Proc. Adv. Neural Inf. Process. Syst, pp. 424–432, 2014.

[124] C. Zhu and Y. Peng, "A boosted multi-task model for pedestrian detection with occlusion handling," IEEE Trans. Image Process., vol. 24, no. 12, pp. 5619–5629, Dec. 2015.

[125] Cao, Y. Pang and X. Li, "Learning multilayer channel features for pedestrian detection," IEEE Trans. Image Process., vol. 26, no. 7,pp. 3210–3220, Jul. 2017.

[126] J. Cao, Y. Pang, and X. Li, "Pedestrian detection inspired by appearance constancy and shape symmetry," IEEE Trans. Image Process., vol. 25, no. 12, pp. 5538–5551, Dec. 2016.

[127] Li Z, Chen Z, Wu Q.J, Liu C.: "Real-time pedestrian detection with deep supervision in the wild". SIViP 13(4), 761–769, 2019.

[128] Wang X, Xiao T, Jiang Y, Shao S, Sun J & Shen C. "Repulsion loss: Detecting pedestrians in a crowd". In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7774-7783, 2018.

[129] Zhang S, Yang X, Liu Y, Xu, C, "Asymmetric multi-stage CNNs for small-scale pedestrian detection". Neurocomputing 12–26, 2020.

[130] Liu W, Liao S, Hu W, Liang X, Chen X, “Learning efficient single-stage pedestrian detectors by asymptotic localization fitting”. In: 2018 European Conference on Computer Vision (ECCV), pp. 618–634, 2018.

[131] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 1800-1807, doi: 10.1109/CVPR.2017.195, 2017.

[132] Cao G, Xie X, Yang W, Liao Q, Shi G & Wu J, “Feature-fused SSD: Fast detection for small objects,” In Ninth International Conference on Graphic and Image Processing (ICGIP), Vol. 10615, p. 106151E. International Society for Optics and Photonics, 2017.

[133] Z. Zhao, P. Zheng, S. Xu and X. Wu, "Object Detection With Deep Learning: A Review," in IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 11, pp. 3212-3232, doi: 10.1109/TNNLS.2018.2876865, 2019.

[134] Abadi. M, Barham. P, Chen.J, Chen. Z, Davis. A, Dean. J & Kudlur. M, Tensorflow: “A system for large-scale machine learning,” In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI}, pp. 265-283, 2016.

[135] Wang R. J, Li X & Ling C. X, “Pelee: A real-time object detection system on mobile devices”. arXiv preprint arXiv:1804.06882, 2018.

[136] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. “Cspnet: A new backbone that can enhance learning capability of cnn”. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pages 390–391, 2020.

[137] Murthy, C. B., Hashmi, M. F., & Keskar, A. G, “Optimized MobileNet+ SSD: a real-time pedestrian detection on a low-end edge device”. International Journal of Multimedia Information Retrieval, 1-14, 2021.

[138] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. “Path aggregation network for instance segmentation”. In Proceedings of the IEEE Conference on Computer Vision.

[139] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929, 2020.

[140] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. “Cbam: Convolutional block attention module”. In Proceedings of the European conference on computer vision (ECCV), pages 3–19, 2018.

[141] Alexander Neubeck and Luc Van Gool. Efficient nonmaximum suppression. In 18th International Conference on Pattern Recognition (ICPR 06), volume 3, pages 850–855. IEEE, 2006.

[142] Derui Wang, Chaoran Li, Sheng Wen, Qing-Long Han, Surya Nepal, Xiangyu Zhang, and Yang Xiang. Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples. IEEE Transactions on Cybernetics, 2021.

[143] Roman Solovyev, Weimin Wang, and Tatiana Gabruseva. “Weighted boxes fusion: Ensembling boxes from different object detection models”. Image and Vision Computing, 107:104117, 2021.

[144] Glenn Jocher et al. yolov5. https://github.com/ ultralytics/yolov5, 2021.

[145] Zhi Zhang, Tong He, Hang Zhang, Zhongyuan Zhang, Junyuan Xie, and Mu Li. “Bag of freebies for training object detection neural networks”. arXiv preprint arXiv:1902.04103, 2019.

[146] B. Yang, J. Yan, Z. Lei, and S. Z. Li, “Convolutional channel features,” in Proc. IEEE Int. Conf. Comput. Vis, pp. 82–90, 2015.

[147] S. Zhang, R. Benenson, and B. Schiele, “Filtered channel features for pedestrian detection,” in Proc. CVPR, vol. 1, no. 2, pp. 1751–1760, 2015.

[148] W. Nam, P. Dollar, and J. H. Han, “Local decorrelation for improved pedestrian detection,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 424–432.

[149] Y. Tian, P. Luo, X. Wang, and X. Tang, “Pedestrian detection aided by deep learning semantic tasks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recogniton, pp. 5079–5087, 2015.

[150] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in CVPR, 2012.

[151] Tripathi, Subarna, Gokce Dane, Byeongkeun Kang, Vasudev Bhaskaran, and Truong guyen. "Lcdet: Low-complexity fully-convolutional neural networks for object detection in embedded systems." In Proceedings of the IEEE Conference on Computer Vision and Pattern recognition Workshops, pp. 94-103. 2017.

[152] H. Mao, S. Yao, T. Tang, B. Li, J. Yao and Y. Wang, "Towards Real-Time Object Detection on Embedded Systems," in IEEE Transactions on Emerging Topics in Computing, vol. 6, no. 3, pp. 417-431, 1 July-Sept. 2018.