Pedestrian Detection based on Hand-crafted and Deep Learning Methods

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

DOCTOR OF PHILOSOPHY

Submitted by

Sweta Panigrahi

(Roll No. 717046)

Under the guidance of

Dr.U.S.N.Raju

Assistant Professor

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL

WARANGAL - 506 004, (T.S.) INDIA

December - 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WRANGAL
WARANGAL - 506 004, (T.S.) INDIA

THESIS APPROVAL FOR Ph.D.

This is to certify that the thesis entitled, Pedestrian Detection based on Hand-crafted and Deep Learning Methods, submitted by Ms. Sweta Panigrahi [Roll No.717046] is approved for the degree of DOCTOR OF PHILOSOPHY at National Institute of Technology Warangal.

Examiner

Research Supervisor
Dr.U.S.N.Raju
Dept. of Computer Science and Engg.
NIT Warangal

Chairman
Prof. S. G. Sanjeevi
Dept. of Computer Science and Engg.
NIT Warangal

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WRANGAL
WARANGAL - 506 004, (T.S.) INDIA

CERTIFICATE

This is to certify that the thesis entitled, Pedestrian Detection based on Hand-crafted and Deep Learning Methods, submitted in partial fulfillment of requirement for the award of degree of DOCTOR OF PHILOSOPHY to National Institute of Technology Warangal, is a bonafide research work done by Ms. Sweta Panigrahi [Roll No.717046] under my supervision. The contents of the thesis have not been submitted elsewhere for the award of any degree.

Research Supervisor
Dr. U.S.N.Raju
Assistant Professor
Dept. of Computer Science and Engg.
NIT Warangal, India

Place: NIT Warangal
Date: 10th December, 2021

DECLARATION

This is to certify that the work presented in the thesis entitled *“Pedestrian Detection based on Hand-crafted and Deep Learning Methods"* is a bonafide work done by me under the supervision of Dr.U.S.N. Raju 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.

Sweta Panigrahi
(Roll No. 717046)
Date: 10th December, 2021.

ACKNOWLEDGMENTS

It is with great pleasure that I acknowledge my sincere thanks and deep sense of gratitude to my supervisor Dr.U.S.N.Raju, Department of Computer Science and Engineering for his valuable guidance throughout the course. His technical perception, profound knowledge, sincere effort in guiding a student and devotion to work have greatly charmed and constantly motivated me to work towards the goal. He always gave me ample time for discussions, reviewing my work and suggesting requisite corrections.

I extend my gratitude to all my Doctoral Scrutiny Committee members Prof. S. G. Sanjeevi, Dr. Ch. Sudhakar, Dr. R. Padmavathy and Dr. D Sreenivasarao for their insightful comments and suggestions during my research work. I am immensely thankful to Prof. P. Radha Krishna, Head of Dept. of CSE, for providing adequate facilities in the department to carry out the research work. I wish to express my sincere thanks to Prof. N.V. Ramana Rao, Director, NIT Warangal for his official support and encouragement.

I would also like to thank my co-research scholar and friends for their valuable suggestions and for extending selfless cooperation. My gratitude to my family for their unconditional love, support and prayers for my success in achieving the goal. Lastly, I am deeply indebted to the almighty for giving the opportunity to pursue this line.

Sweta Panigrahi

Dedicated to

My Family and Teachers

ABSTRACT

Pedestrian detection is one of the most challenging research areas in computer vision, as it involves classifying the image as well as localizing the pedestrian. Owing to its applications especially in automated surveillance and robotics, it is exceedingly sought-after. The problem of pedestrian detection, although approached by many computer vision researchers is far from solved. The scale, pose, occlusion, illumination, and many such factors affect the performance of the methods. Traditional methods use hand-crafted features to characterize pedestrians. The hand-crafted features can be used to extract shape, color and texture features; which is then classified by using Support Vector Machine (SVM). As in recent years, deep learning models such as Convolutional Neural Networks (CNNs) have become an eminent state-of-the-art in detection challenges, which unlike the manually designed feature extraction mechanism, results in more accuracy. This work gradually moves from proposing new hand-crafted feature algorithms to designing sophisticated complex CNN structures.

Most of the algorithms for pedestrian detection use the Histogram of Oriented Gradients (HOG) as the basic shape feature and combine other features with the HOG to form the feature set, which is usually applied with a classifier such as Support Vector Machine (SVM). Hence, the HOG feature is the most efficient and fundamental feature for pedestrian detection. However, the HOG feature produces feature vectors of different lengths for different image resolutions; thus, the feature vectors are incomparable for the SVM. To handle this a Scale-Invariant Histogram of Oriented Gradients (SI-HOG) for pedestrian detection is proposed. The proposed method forms a scale-space pyramid wherein the histogram bin is calculated. Thus, the gradient information from all the scales is encapsulated in a single fixed-length feature vector. The proposed method is also combined with color and texture features.

To continue the progress made in this field, a deep learning based approach is proposed. A modification of the pre-trained ResNet18 named Multi-layer Feature Fused-ResNet (MF2-ResNet) is proposed. MF2-ResNet is used for 1) feature extraction; which is then classified by using SVM; 2) End-to-End feature extraction and classification by the CNN network. To work on the region proposal aspect of the pedestrian detection, the next deep learning model considered in this thesis is the two-stage detection network Faster R-CNN. Modifications of the most commonly used deep CNN model ResNet18 is proposed. The modified CNN structure named Dilate-Condense Resnet (DCResNet) and MF2-ResNet forms the base of the Faster R-

CNN model utilized to predict the locations of pedestrians in the image. The proposed method has been improved in terms of the feature map extraction of the image. As the two-stage detection network requires more computation and train-test time, lastly, the single-stage You Only Look Once (YOLO) detection network is worked with. The single-stage detection networks YOLOv2 and YOLOv3 has attained a satisfactory performance in object detection without compromising the computation speed and is among the state-of-the-art CNN based method. YOLO framework can be leveraged to use in pedestrian detection as well. In this work, improved YOLOv2 networks, called DSM-IDM-YOLO, InceptionDepth-wiseYOLOv2 and FireYOLOv2 is proposed. The proposed models use a modified DarkNet19 and DarkNet53, engineered for a robust feature formation. A LightWeight FireYOLOv2 is also contributed to this work to obtain a trade-off between detection accuracy and computation speed. A modified YOLOv3 is proposed, which is named MultiScale-MultiLevel-SqueezeNetYOLOv3 (MS-ML-SNYOLOv3). An improved SqueezeNet base network forms the basis of the proposed YOLOv3.

The hand-crafted and CNN proposed methods are tested on three established benchmark pedestrian datasets: INRIA, NICTA, and Daimler. The Faster R-CNN and YOLO proposed methods are evaluated on INRIA Pedestrian, PASCAL VOC 2012 and Caltech Pedestrian datasets. For evaluation, various hand-crafted shape, texture and color features as well as comparison with state-of-the-art detection methods i.e., Faster-RCNN, YOLOv2, YOLOv3 and Single Shot Multibox Detector (SSD) is performed. For comparison, Detection Error Trade-off Curve, Precision Recall Curve, Log Average Miss Rate (LAMR) and Average Precision (AP) performance metrics are used. Friedman Test and F-distribution Test is used for statistical analysis of the proposed methods.

Contents

ACKNOWLEDGEMENTS………………………………………………………………… i

ABSTRACT…………………………………………………………………………… iii

List of Figures………………………………………………………………………….. viii

List of Tables…………………………………………………………………………... xii

Abbreviations………………………………………………………………………….. xiv

Chapter 1: Introduction…………………………………………………………………. 1

1.1 Object Detection…………………………………………………………... 1

1.1.1 Objectness Detection………………………………………………… 1

1.1.2 Category based Object Detection……………………………………. 1

1.2 Pedestrian Detection……………………………………………………… 2

1.3 Different Methods for PD……………………………………………….. 3

1.3.1 Hand-crafted Features + SVM…………………………………….. 3

1.3.2 Convolutional Neural Network Features + SVM……………………. 3

1.3.3 End-to-End Convolutional Neural Network…………………………. 3

1.3.4 Two-stage Network with base CNN………………………………… 4

1.3.5 Single-stage Network with base CNNs……………………………… 4

1.4 Different Hand-crafted Features for PD…………………………………… 4

1.5 Different CNN Features for PD…………………………………………… 5

1.6 Different Region Proposals for PD……………………………………….... 5

1.7 Introduction to State-of-the-art Technology for PD………………………. 5

1.8 Motivation, Aim and Problem Statement of the Thesis…………………… 5

1.9 Objectives of this Work…………………………………………………… 6

1.10 Different Benchmark Image Datasets in this Work……………………… 8

1.11 Different Performance Measure for Comparing PD Methods…………… 12

1.12 Organization of the Thesis……………………………………………….. 15

Chapter 2: Related Work……………………………………………………………... 16

2.1 Feature Extraction………………………………………………………… 16

2.1.1 Shape Features for PD……………………………………………… 16

2.1.2 Color Features for PD……………………………………………… 20

2.1.3 Texture Features for PD…………………………………………….. 21

2.1.4 CNN based Features for PD………………………………………… 22

2.2 Classifier………………………………………………………………….... 23

2.3 Region Proposal Networks………………………………………………… 23

2.4 Literature on PD…………………………………………………………… 26

Chapter 3: A Scale Invariant Histogram of Oriented Gradients Feature Extraction for Pedestrian Detection in Multiresolution Image Dataset…………………………………….. 31

3.1 Introduction…………………………………………………………………….... 31

3.1.1 Histogram of Oriented Gradients Feature Pyramid………………………….. 31

3.2 Methodology……………………………………………………………………. 31

3.2.1 Scale Invariant Histogram of Oriented Gradients... 32

3.2.2 Derivation of SI-HOG Feature Vector………………………………………. 33

3.3 Experimental Results and Discussion……………………………………………... 34

3.3.1 INRIA Pedestrian Dataset…………………………………………………… 34

3.3.2 NICTA Pedestrian Dataset………………………………………………….... 36

3.3.3 Daimler Pedestrian Dataset………………………………………………….. 38

3.3.4 Result of Fusion Strategies…………………………………………………... 39

3.3.5 Statistical Analysis………………………………………………………….... 39

3.4 Observations... 41

Chapter 4: A Multi-layer Feature Fused-Resnet Model for Pedestrian Detection……….. 42

4.1 Introduction…………………………………………………………………….... 42

4.1.1 Base Convolutional Neural Network Features………………………………. 42

4.2 Methodology……………………………………………………………………. 43

4.2.1 Multi-layer Feature Fused-Resnet…………………………………………… 43

4.2.2 MF2-ResNet Trained Model Algorithm…………………………………….... 44

4.2.3 Multi-layer Feature Fused-Resnet Features + SVM…………………………. 45

4.2.4 End-to-End Multi-layer Feature Fused-Resnet………………………………. 46

4.3 Experimental Results and Discussion……………………………………………... 46

4.3.1 INRIA Pedestrian Dataset…………………………………………………… 47

4.3.2 NICTA Pedestrian Dataset………………………………………………….... 47

4.3.3 Daimler Pedestrian Dataset………………………………………………….. 50

4.3.4 Statistical Analysis………………………………………………………….... 52

4.4 Observations…………………………………………………………………….. 53

Chapter 5: Faster R-CNN based on Dilate-Condense ResNet and Multi-layer Feature Fused-Resnet Model for Pedestrian Detection……………………………………………. 54

5.1 Introduction…………………………………………………………………….... 54

5.1.1 ResNet Feature Extraction Layer with Faster R-CNN………………………. 54

5.2 Methodology……………………………………………………………………. 55

5.2.1 Dilate-Condense ResNet based Faster R-CNN……………………………… 55

5.2.2 Multi-layer Feature Fused-Resnet based Faster R-CNN…………………….. 56

5.3 Experimental Results and Discussion……………………………………………... 62

5.3.1 INRIA Pedestrian Dataset………………………………………………… 62
5.3.2 PASCAL VOC 2012 Dataset…………………………………………….. 64
5.3.3 Statistical Analysis……………………………………………………... 64
5.4 Observations………………………………………………………………….. 67
Chapter 6: YOLO based on DarkNet: Depth-wise Separable, Inception Depth-wise & Fire and SqueezeNet: Multiscale-Multilevel for Pedestrian Detection…………………….. 68
6.1 Introduction…………………………………………………………………... 68
6.1.1 Depth-wise Separable Convolution Module…………………………. 68
6.1.2 Inception Module……………………………………………………….. 69
6.1.3 Squeeze and Expand Module………………………………………... 69
6.2 Methodology…………………………………………………………………. 70
6.2.1 Depth-wiseSeparableModule-InceptionDepthwiseModule-YOLO…………. 70
6.2.2 InceptionDepth-wiseYOLOv2…………………………………………….. 73
6.2.3 FireYOLOv2……………………………………………………………. 76
6.2.4 Multiscale-Multilevel-SqueezeNet…………………………………….... 78
6.3 Experimental Results and Discussion……………………………………... 83
6.3.1 INRIA Pedestrian Dataset……………………………………………… 84
6.3.2 PASCAL VOC 2012 Dataset…………………………………………... 88
6.3.3 Caltech Pedestrian Dataset…………………………………………... 91
6.3.4 Statistical Analysis……………………………………………………... 96
6.4 Observations…………………………………………………………………. 99
Chapter 7: Conclusion and Future Scope……………………………………………. 100
List of Publications………………………………………………………………….. 102
References……………………………………………………………………………… 103

List of Figures

1.1 a) Objectness detection b) Category based object detection……………………….…… 1

1.2 Overview of Pedestrian Detection…………………….…………………………………. 2

1.3 Block diagram for Hand-crafted features with Classifier…………………………….. 3

1.4 Block diagram for Deep CNN features with Classifier………………………..…... 3

1.5 Block Diagram for End-to-End CNN…………………….………………………............. 3

1.6 Block Diagram for two-stage Network with base CNN……………………………..... 4

1.7 Block Diagram for single-stage Network with base CNN………………………………. 4

1.8 Structure Diagram of Pedestrian Detection Methods and Datasets in this Work……..... 7

1.9 INRIA Region Images: First and second row Positive Pedestrian samples third and fourth row Negative Pedestrian-free Samples………………………………………… 9

1.10 INRIA Full Images: First row shows positive train images. Second row shows positive test image…………………….……………………….………………........................... 9

1.11 NICTA Region Images: First row Positive Pedestrian samples second row Negative Pedestrian-free Samples…………………….…………………………………………… 9

1.12 Daimler Region Images: First row Positive Pedestrian samples second row Negative Pedestrian-free Samples…………………….…………………………………………… 10

1.13 Sample images from PASCAL VOC 2012 Dataset…………………………………... 11

1.14 Sample images from Caltech Pedestrian Dataset……………………………………... 12

1.15 Confusion Matrix…………………….…………………….…………………………… 13

1.16 ROC Curve Example…………………….…………………….………………............. 13

1.17 DET Curve Example…………………….…………………….………………………… 13

1.18 P-R Curve Example…………………….………………………………………………….. 14

2.1 Representation of cells and block for HOG feature calculation………………………… 17

2.2 8×8 cell of an image…………………….…………………….………………………… 18

2.3 Magnitude of 8×8 cell…………………….…………………….……………………….. 18

2.4 Direction of 8×8 cell...…………………….…………………….……………………….. 18

2.5 Process of updating the 9-bin histogram for the cell…………………….……………… 18

2.6 Final 9-bin histogram for the cell…………………….……………………………….. 18

2.7 Formation of ExHOG feature vector…………………….……………………………… 20

2.8 8-Neighborhood of pixel around $I_{c.}$ …………………….…………………………….. 21

2.9 LBP calculation (a) Original Image (b) Resultant binary numbers for (a). (c) Decimal weights for the corresponding locations………………………………………………… 21

3.1 Block diagram of the proposed SI-HOG……………………………………………….. 33
3.2 The ROC Curve for Single Resolution INRIA Pedestrian Dataset..……………………… 35
3.3 The ROC Curve for Multiresolution INRIA Pedestrian Dataset…..…………………… 35
3.4 The DET Curve for Single Resolution INRIA Pedestrian Dataset..…………………….. 35
3.5 The DET Curve for Multiresolution INRIA Pedestrian Dataset..……………………….. 36
3.6 The DET Curve for Single Resolution NICTA Pedestrian Dataset.…….………………. 37
3.7 The DET Curve for Multiresolution NICTA Pedestrian Dataset………..……………. 37
3.8 The DET Curve for Single Resolution Daimler Pedestrian Dataset…………................ 38
3.9 The DET Curve for Multiresolution Daimler Pedestrian Dataset.…………….............. 38
4.1 The framework of Pedestrian Detection using 1) CNN features with SVM Classifier and 2) End-to-End CNN………………………………………………………………. 42
4.2 Block Diagram for Multi-layer Feature Fused-ResNet (MF2-ResNet)…………………. 44
4.3 Block Diagram for Multi-layer Feature Fused-Resnet Features with SVM…................ 45
4.4 Block Diagram for End-to-End Multi-layer Feature Fused-Resnet Network…............... 46
4.5 DET Curve for INRIA Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM………………………………………….…………………………… 48
4.6 DET Curve for INRIA Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network…………………………………………………………………….. 48
4.7 DET Curve for NICTA Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM……………………………………………………………………. 49
4.8 DET Curve for NICTA Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network…………………………………………………………………….. 50
4.9 DET Curve for Daimler Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM……………………………………………………………………. 51
4.10 DET Curve for NICTA Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network…………………………………………………………………….. 51
5.1 The Block Diagram of the Proposed Faster RCNN with the modified DCResNet as the base…………………….…………………….………………………………………... 57
5.2 The structure of Dilute ad Condense 3×3 Network for I, input for the Feature Map Block I……………..……………………….………………………………………… 58
5.3 The structure of Dilute ad Condense 3×3 Network for II, input for the Feature Map Block II…………………………………………………………….………………. 58
5.4 The structure of Dilute ad Condense 3×3 Network for III, input for the Feature Map Block III……………………………………...………………………………….. 58

5.5 The Block Diagram of the Proposed Faster RCNN with the modified MF2ResNet as the base………………………………………………………………….......................... 60
5.6 DET Curve for INRIA Pedestrian Dataset……………………………….............. 62
5.7 P-R Curve for INRIA Pedestrian Dataset……………………………………….... 63
5.8 Sample INRIA Pedestrian output images with score and bounding boxes for the proposed Faster RCNN(DCResNet)……………………………………………….. 63
5.9 Sample INRIA Pedestrian output images with score and bounding boxes for the proposed Faster RCNN(MF2ResNet)……………………………………………… 64
5.10 DET Curve for PASCAL VOC 2012 Dataset…………………………………… 65
5.11 P-R Curve for PASCAL VOC 2012 Dataset…………………………………… 65
5.12 Sample PASCAL VOC 2012 output images with score and bounding boxes for the proposed Faster RCNN(DCResNet)……………………………………………….. 66
5.13 Sample PASCAL VOC 2012 output images with score and bounding boxes for the proposed Faster RCNN(MF2ResNet)……………………………………………… 66
6.1 The overall feature map formation in Depth-wise Separable Convolution……............... 69
6.2 General Idea behind a traditional Inception Module…………………………………….. 69
6.3 Fire Module………………………………………………………………………… 70
6.4 Block Diagram of the Proposed DSM-IDM-YOLO………………………………….. 72
6.5 Block Diagram of Depth-wise Separable Module I of DSM-IDM-YOLO……............... 72
6.6 Block Diagram of Depth-wise Separable Module II of DSM-IDM-YOLO…………….. 72
6.7 Block Diagram of Inception Depth-wise Module of DSM-IDM-YOLO……………….. 73
6.8 Idea behind the Proposed Inception Depth-wise Convolution Module…………………. 73
6.9 Block Diagram of InceptionDepth-wiseYOLOv2………………………………………. 76
6.10 Block Diagram of FireYOLOv2………………………………………………………. 79
6.11 Block Diagram of the Proposed MS-ML-SNYOLOv3………..…...................... 81
6.12 DET Curve of INRIA Pedestrian Dataset for YOLOv2 Methods…..……………………. 85
6.13 P-R Curve of INRIA Pedestrian Dataset for YOLOv2 Methods…..……………………. 85
6.14 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO……………………………….... 86
6.15 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2…………………………………………………………………………….. 86
6.16 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2…………….. 86

6.17 Scale Analysis of INRIA Pedestrian Dataset for YOLOv2 Methods…………………… 87
6.18 DET Curve of INRIA Pedestrian Dataset for YOLOv3 Method...……………………… 87
6.19 P-R Curve of INRIA Pedestrian Dataset for YOLOv3 Method…………………………. 88
6.20 DET Curve of PASCAL VOC 2012 Dataset for YOLOv2 Methods……………………. 89
6.21 P-R Curve of PASCAL VOC 2012 Dataset for YOLOv2 Methods…...………………… 90
6.22 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO……………………............... 90
6.23 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2……………………………………………………………………….. 91
6.24 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2……............. 91
6.25 Scale Analysis of PASCAL VOC 2012 Dataset for YOLOv2 Methods…..…….............. 92
6.26 DET Curve of Caltech Pedestrian Dataset for YOLOv2 Methods……………………… 93
6.27 P-R Curve of Caltech Pedestrian Dataset for YOLOv2 Methods…..…………………… 93
6.28 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO……………………............... 94
6.29 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2……………………………………………………………………….. 94
6.30 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2……............. 94
6.31 Scale Analysis of Caltech Pedestrian Dataset for YOLOv2 Methods…...……............ 95
6.32 DET Curve of Caltech Pedestrian Dataset for YOLOv3 Method………..……………… 95
6.33 P-R Curve of Caltech Pedestrian Dataset for YOLOv3 Method………………………… 96

List of Tables

1.1 Characteristics of the INRIA, NICTA, and Daimler Pedestrian Datasets……….............. 11

2.1 YOLOv2 with base DarkNet19 Feature Extraction Layers……………………………… 25

2.2 YOLOv3 with base DarkNet53 Feature Extraction Layers……………………………… 26

3.1 Miss Rate values for Single and Multiresolution INRIA Pedestrian Dataset……………. 36

3.2 Miss Rate values for Single and Multiresolution NICTA Pedestrian Dataset …………... 37

3.3 Miss Rate values for Single and Multiresolution Daimler Pedestrian Dataset ………….. 39

3.4 Rank table of Friedman Test for the proposed methods in Single and Multiresolution Dataset…………………………………………………….. 40

4.1 Miss Rate for INRIA Pedestrian Dataset for the Proposed Methods……………………. 49

4.2 Miss Rate for NICTA Pedestrian Dataset for the Proposed Methods……....................... 50

4.3 Miss Rate for Daimler Pedestrian Dataset for the Proposed Methods……......………… 52

4.4 Rank Table of Friedman Test for the Proposed Methods in Multi-layer Feature Fused-Resnet Features + SVM………………………………………..…........................... 52

4.5 Rank Table of Friedman Test for the Proposed Methods in End-to-End Multi-layer Feature Fused-Resnet Network…………………………...........…......................... 53

5.1 FasterRCNN base ResNet18 layers……………………………………..….................. 55

5.2 The Proposed FASTER RCNN(DCRESNET) network’s layer and characteristics…….. 59

5.3 The Proposed FASTER RCNN(MF2RESNET) network’s layer and characteristics…… 61

5.4 LAMR and AP values for INRIA Pedestrian Dataset…………………………………… 63

5.5 LAMR and AP values for PASCAL VOC 2012 Dataset……………………………….. 65

5.6 Faster RCNN (DCResNet) F-DISTRIBUTION Test Analysis………………………….. 67

5.7 Faster RCNN (MF2ResNet) F-DISTRIBUTION Test Analysis………………………… 67

6.1 DSM-IDM-YOLO Feature Layers architecture and characteristics…………………….. 74

6.2 InceptionDepth-wiseYOLOv2 Feature Layers architecture and characteristics………… 77

6.3 FireYOLOv2 Feature Layers architecture and characteristics…………….….................. 80

6.4 Layers and Feature Map characteristics of proposed MS-ML-SNYOLOv3…………….. 82

6.5 LAMR and AP values of INRIA Pedestrian Dataset for YOLOv2 Methods……………. 85

6.6 LAMR and AP values of INRIA Pedestrian Dataset for YOLOv3 Method...................... 88

6.7 LAMR and AP values of PASCAL VOC 2012 Dataset for YOLOv2 Methods…...……. 90

6.8 LAMR and AP values of Caltech Pedestrian Dataset for YOLOv2 Methods………….... 93

6.9 LAMR and AP values of Caltech Pedestrian Dataset for YOLOv3 Method……………. 96

6.10 Rank Table of Friedman Test for the Proposed DSM-IDM-YOLO……………………..…. 97

6.11 Rank Table of Friedman Test for the Proposed InceptionDepth-wiseYOLOv2………… 98

6.12 Rank Table of Friedman Test for the Proposed FireYOLOv2 and LightWeightFireYOLOv2……………………………………………......…................. 98

6.13 Rank Table of Friedman Test for the Proposed MS-ML-SNYOLOv3…......................... 99

Abbreviations

CNN	Convolutional Neural Network
YOLO	You Only Look Once
TP	True Positive
FP	False Positive
FN	False Negative
TN	True Negative
ROC	Receiver Operator Characteristics
FPPW	False Positive Per Window
DET	Detection Error Trade-Off
FPPI	False Positive Per Image
LAMR	Log Average Miss Rate
P-R	Precision-Recall
AP	Average Precision
HOG	Histogram of Oriented Gradients
ExHOG	Extended Histogram of Oriented Gradients
LBP	Local Binary Pattern
SVM	Support Vector Machine
RPN	Region Proposal Network
rbf	radial basis function
SSD	Single Shot Multibox Detector
SI-HOG	Scale Invariant Histogram of Oriented Gradients
MF2-ResNet	Multi-layer Feature Fused-ResNet
FC	Fully Connected
DCResNet	Dilate-Condense ResNet
DSM-IDM-YOLO	Depth-wiseSeparableModule-InceptionDepthwiseModule-YOLO
ReLu	Rectified Linear Unit
MS-ML-SN	Multiscale-Multilevel-SqueezeNet

Chapter 1

Introduction

Pedestrian detection is a well-established challenging problem in the computer vision field. The task is to detect the presence of different pedestrians in the image along with their location in a bounding box format. It plays an essential role in computer vision applications. Even though extensive research has been performed on pedestrian detection, significant improvements were made in recent works, which suggests that the research has not reached a saturation point.

1.1 Object Detection

The aim of object detection is to identify the presence of various individual objects in a given image. It has various applications among pedestrian detection, vehicle detection, manufacturing industry, security, and face detection. It can be divided into two sub categories: Objectness detection and Category based object detection.

1.1.1 Objectness Detection

In objectness detection, all the objects in the image are detected. No class label is specified in this case. An example is shown in Figure 1.1(a). In this figure, every object such as tree, building structure are detected.

1.1.2 Category based Object Detection

In category based object detection, the objects are detected in the image as per the specified class label. An example is shown in Figure 1.1(b). In this figure, the class specified is vehicle and hence the moto bike and car objects are detected.

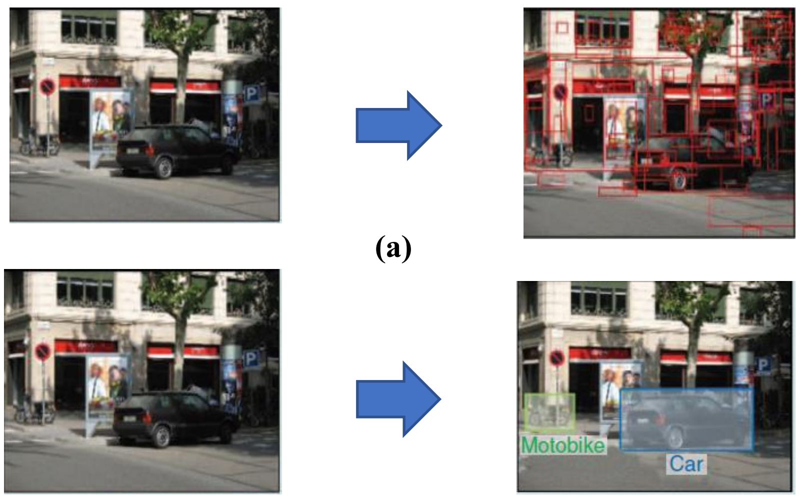


Figure 1.1 a) Objectness detection **b)** Category based object detection

1.2 Pedestrian Detection

Pedestrian detection (PD) comes under category based object detection and is one of the recognized area of object detection. Various applications, such as human–computer interaction for video games, robotics, video surveillance, and smart vehicles, have motivated research on human and pedestrian detection. Nonetheless, pedestrian detection is a challenging problem owing to the large intra-class variability arising from clothing, color, appearance, and pose. In addition, external factors such as illumination, background clutter, and partial occlusions further complicate the problem [1, 2].

Most pedestrian detection algorithms involve similar stages of computation. First, the pixel-level content of the image undergoes complex transformations to represent higher-level features, which are computed via feature-extraction methods. Second, region extraction is carried out based on the similarities of the features in a region. Third, from these region proposals, the features are fed to a classifier, which determines whether the region represents a pedestrian.

In Figure 1.2, the overview of this process is shown. The pedestrian detection can be solved either by per-window or per-image evaluation. In per-window evaluation, the regions are pre-processed and later the feature extraction and classification are performed. In per-image evaluation, the features extraction is followed by region proposals to predict the locations and confidence scores of the pedestrians.

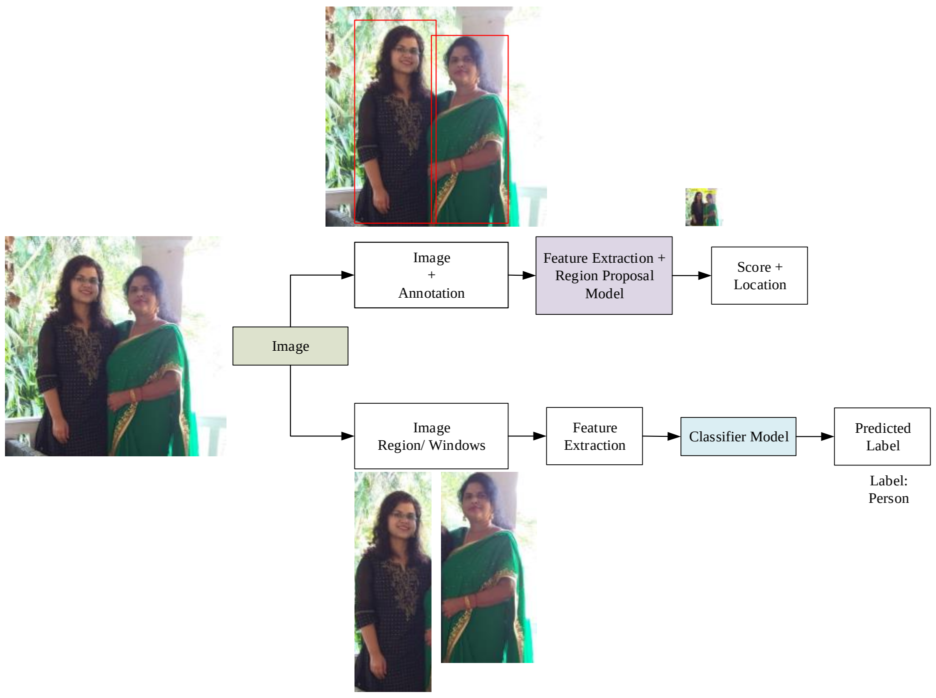


Figure 1.2 Overview of Pedestrian Detection

1.3 Different Methods for PD

The pedestrian detection can be approached broadly by five methods as described below.

1.3.1 Hand-crafted Features + Classifier

In this method, hand-crafted features such as shape, texture, and color are extracted from the image. Then, the feature extracted are feed into the classifier to predict the label i.e., pedestrian or non-pedestrian. The process is shown in Figure 1.3.

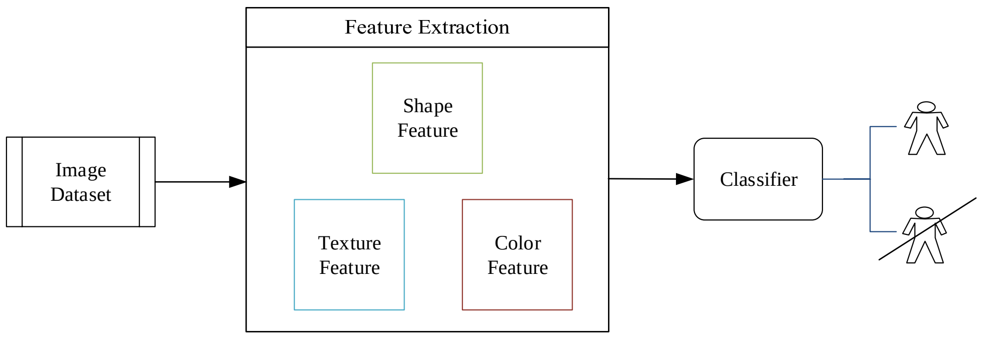


Figure 1.3 Block diagram for Hand-crafted features with Classifier

1.3.2 Convolutional Neural Network Features + Classifier

In this method, the features are extracted from a trained Convolutional Neural Network (CNN) [3] model. A pre-trained CNN is used and trained via transfer learning [4]. The features are extracted from a higher layer of the trained CNN. These features are then classified i.e., pedestrian or non-pedestrian label. The process is shown in Figure 1.4.

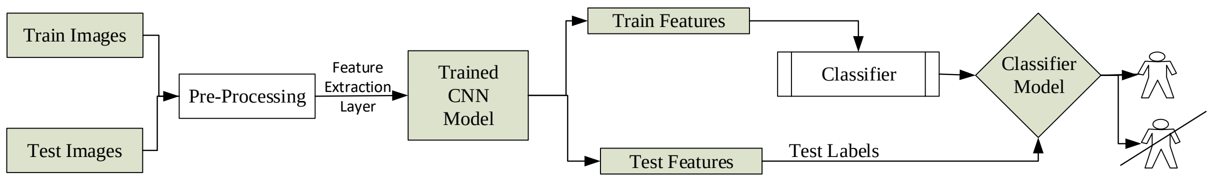


Figure 1.4 Block diagram for CNN features with Classifier

1.3.3 End-to-End Convolutional Neural Network

In this method, the features extraction as well as classification is performed by the End-to-End CNN. A pre-trained CNN is used and trained via transfer learning. The predicted classification labels are either pedestrian or non-pedestrian label. The process is shown in Figure 1.5.

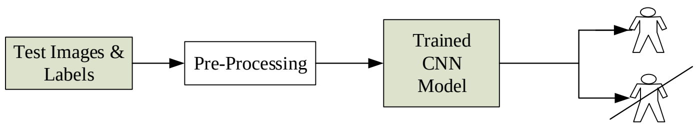


Figure 1.5 Block Diagram for End-to-End CNN

1.3.4 Two-stage Network with base CNN

In this method, along with the classification label, the bounding box of the pedestrian is also predicted. This process takes place in two stages i.e., region proposals and classification. A base CNN serves as a backbone used for feature extraction. The feature map is processed for proposing regions as well as for classification. The process is shown in Figure 1.6.

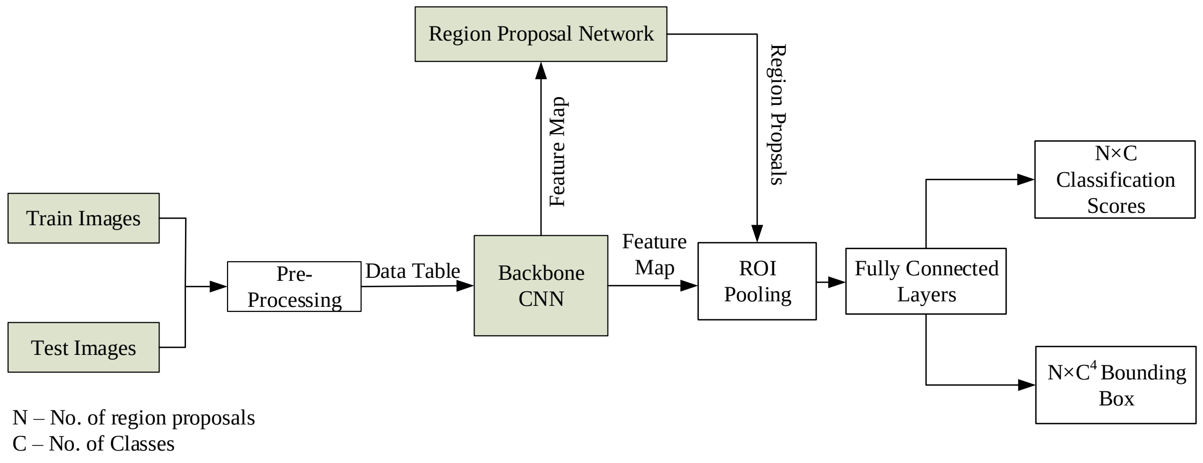


Figure 1.6 Block Diagram for two-stage network with base CNN

1.3.5 Single-stage Network with base CNN

In this method also the classification label along with the location is yielded. This network performs the region proposal and classification in one pipeline. The network uses the base CNN to extract features. This is further processed to generate the final detection result by utilizing the classification scores and the bounding box. The process is shown in Figure 1.7.

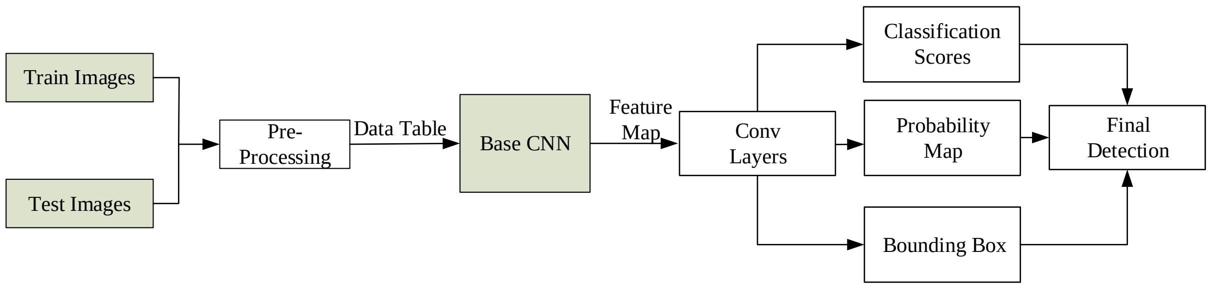


Figure 1.7 Block Diagram for single-stage network with base CNN

1.4 Different Hand-crafted Features for PD

Feature extraction methods are used to represent the image content. The use of traditional image processing methods is known as hand-crafted methods and the features extracted by these methods yield hand-crafted features. A particular hand-crafted feature algorithm can only describe a single feature of the image, which can be shape, texture or color. The algorithms for shape, texture and color feature extraction are given in detail in Chapter-2.

1.5 Different CNN Features for PD

Feature extraction methods which are performed by deep learning methods produce a better representation of the image feature when compared to that of hand-crafted mechanism. Specifically, the CNNs are used for this purpose. With the backpropagation error correction process the CNN models provide substantially improved features. The CNN features can be extracted from any layer of the network, but the higher-level layers are preferred for this as they contain detailed features of the image. Various pre-trained CNN models are available which are given in detail in Chapter-2.

1.6 Different Region Proposals for PD

An image can contain one or more than one object of interest. Here, the region proposal method comes into play. Region proposal algorithms are used to segment the regions or areas from the image yielding regions of the object of interest. The region proposal methods are based on both traditional image processing and deep learning methods. In the traditional approach, regions are grouped under neighboring area feature whereas the deep learning approach uses the CNN feature maps for generating region proposals. These methods are given in detail in Chapter-2.

1.7 Introduction to State-of-the-art Technology for PD

To identify or localize the objects or pedestrian in the images, recent state-of-the-art technologies are getting developed and used successfully. The state-of-the-art technology in detection domain constitutes only deep learning methods. The deep learning methods involve CNN as the base with additional modules to complete the detection task. The pre-trained CNN models serve the base for feature extraction and the region proposal step also uses either the base CNN feature map or probability map from the CNN. These methods are discussed in detail in Chapter-2. The state-of-the-art technologies in detection takes less computation overhead and are successfully applied in real time scenarios.

1.8 Motivation, Aim and Problem Statement of the Thesis

Motivation

The following three issues motivated for this research work.

- Extraction of comprehensive feature representation of image for pedestrian detection purpose using traditional hand-crafted features.
- Role of CNN features for pedestrian detection which outperforms the traditional hand-crafted methods.

- Deep learning region proposal methods to improve localization and detection of pedestrians in the images.

Aim

This thesis aims to provide a better feature extraction mechanism for pedestrian detection. Both the hand-crafted and deep CNN methods are explored to perfect the feature content of the image.

Problem statement

Pedestrian detection is a challenging and vital problem in computer vision applications. The variation in factors such as pose, illumination, background, truncation and occlusion make the recognition and localization of pedestrians in an image a gruelling task. A detailed and elegant feature extraction mechanism can tackle the problem to a great extent. In this pursuit, hand-crafted and deep learning based feature extraction methods are proposed in this work.

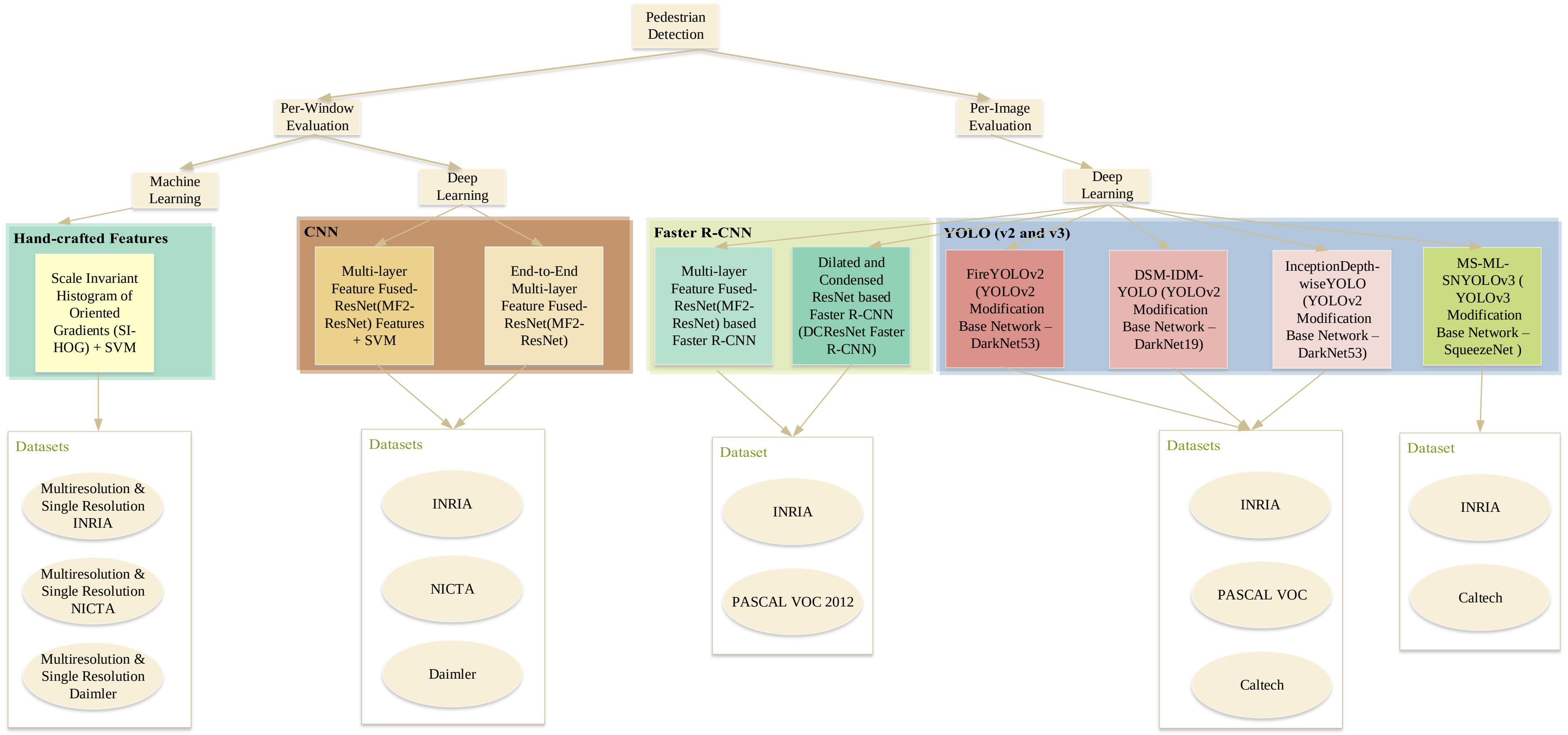
1.9 Objectives of this Work

To address the pedestrian detection problem, improved feature extraction techniques from both the hand-crafted and the deep learning methods are presented in this work. The following objectives are set in this thesis.

- To propose a new robust hand-crafted feature extraction mechanism.
- To improve the convolutional neural network based feature extraction methods.
- To propose Faster RCNN architectures with a modified base convolutional neural network.
- To employ and improve You Only Look Once (YOLOv2 and v3) paradigm.

To achieve the given objectives the contributions of the thesis are listed below and it’s represented in a structure diagram in Figure 1.8.

- A Scale Invariant Histogram of Oriented Gradients Feature Extraction for Pedestrian Detection in Multiresolution Image Dataset.
- A Multi-layer Feature Fused-Resnet Model for Pedestrian Detection.
- Faster RCNN based on Dilate-Condense ResNet and Multi-layer Feature Fused-Resnet Model for Pedestrian Detection.
- YOLOv2 based on DarkNet: Depth-wise Seperable, Inception Depth-wise & Fire modules for Pedestrian Detection.
- MultiScale-MultiLevel-SqueezeNet based YOLOv3 for Pedestrian Detection.



Pedestrian Detection
Per-Window Evaluation
Per-Image Evaluation
Machine Learning
Deep Learning
Deep Learning
Hand-crafted Features
Scale Invariant Histogram of Oriented Gradients (SI-HOG) + SVM
CNN
Multi-layer Feature Fused-ResNet(MF2-ResNet) Features + SVM
End-to-End Multi-layer Feature Fused-ResNet(MF2-ResNet)
Faster R-CNN
Multi-layer Feature Fused-ResNet(MF2-ResNet) based Faster R-CNN
Dilated and Condensed ResNet based Faster R-CNN (DCResNet Faster R-CNN)
YOLO (v2 and v3)
FireYOLOv2 (YOLOv2 Modification Base Network – DarkNet53)
DSM-IDM-YOLO (YOLOv2 Modification Base Network – DarkNet19)
InceptionDepth-wiseYOLO (YOLOv2 Modification Base Network – DarkNet53)
MS-ML-SNYOLOv3 (YOLOv3 Modification Base Network – SqueezeNet)
Datasets
Multiresolution & Single Resolution INRIA
Multiresolution & Single Resolution NICTA
Multiresolution & Single Resolution Daimler
Datasets
INRIA
NICTA
Daimler
Dataset
INRIA
PASCAL VOC 2012
Datasets
INRIA
PASCAL VOC
Caltech
Dataset
INRIA
Caltech

Figure 1.8 Structure Diagram of Pedestrian Detection Methods and Datasets in this Work

1.10 Different Benchmark Image Datasets in this Work

In this work, five pedestrian benchmark datasets: INRIA Pedestrian, NICTA Pedestrian, Daimler Pedestrian, PASCAL VOC 2012 and Caltech Pedestrian are used and are described in this section.

❖ *INRIA Pedestrian Dataset*

In the INRIA Pedestrian dataset [1], people are standing at different orientations.

(a) *Training set:* It consists of 614 positive images that contain 1,239 pedestrians. A total of 2,478 pedestrians are obtained when both the left and right reflection of an individual pedestrian is considered. There are 1,218 negative (pedestrian-free) images. 10 patches are sampled from each pedestrian-free image randomly. This gives 12,180 negative images.

(b) *Test set:* It consists of 288 positive images that contain 566 pedestrians. A total of 1,132 pedestrians are obtained when both the left and right reflection of an individual pedestrian is considered. There are 453 negative (pedestrian-free) images. 10 patches are sampled from each pedestrian-free image randomly. This gives 4,530 negative images.

(c) *Image Regions:* The positive images are cropped to 128×64 windows. The same size is maintained while extracting negative windows. The window size is selected based on the aspect ratio of a pedestrian.

Figure 1.9 and Figure 1.10 shows the region images and full images from INRIA Pedestrian dataset.

❖ *NICTA Pedestrian Dataset*

The NICTA Pedestrian dataset [5] contains pedestrian positive and negative pedestrian set consisting of 6 resolutions i.e., 8×20, 16×20, 16×40, 32×40, 32×80 and 64×80. The different characteristics of this image database are:

a) *Pedestrian positives:* This set contains pedestrians in the form of a cropped window from existing marked-up input images. The source images for the positive data are scaled into the 6 resolutions as mentioned above.

b) *Pedestrian negatives:* A large negative set is extracted from a set of 5,207 high-resolution pedestrian free images in diverse environments. The negative sets are also grouped into 6 resolutions which are alike in themselves.

c) *Image regions:* Images are taken from 64×80 group. 6,000 and 3,000 positive images along with their mirror-image form positive train and test set respectively. In addition to those 18000 and 9000 negative images forms the negative train and test set respectively. This gives us 30000 images for training and 15000 images for testing.

Some sample images from the dataset are shown in Figure 1.11.

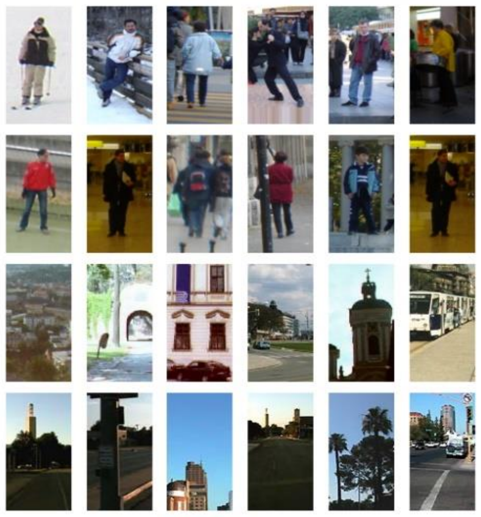


Figure 1.9 INRIA Region Images: First and second row Positive Pedestrian samples third and fourth row Negative Pedestrian-free Samples

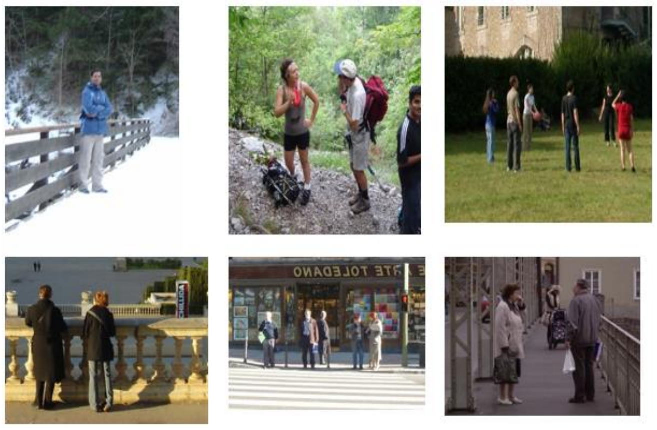


Figure 1.10 INRIA Full Images: First row shows positive train images. Second row shows positive test image

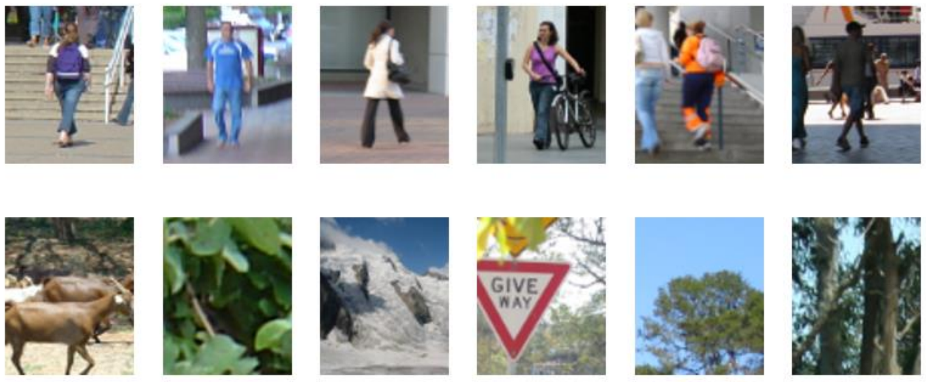


Figure 1.11 NICTA Region Images: First row Positive Pedestrian samples second row Negative Pedestrian-free Samples

❖ *Daimler Pedestrian Dataset*

The Pedestrian images in this Daimler dataset [6] were yielded by manual labeling and extraction of pedestrians in the video frames. The videos were recorded at various day settings with no constraint on clothing or pose. The pedestrians are un-occluded and upright. The pedestrian-free images were obtained from videos that did not contain pedestrians. This dataset is a grayscale image set.

(a) Base dataset: The Daimler Pedestrian dataset contains images with a size of 18 × 36 extracted from videos. The base train data set consists of three sets of 4000 positive or pedestrians and 5000 negative or pedestrian-free images. It also consists two test sets of 4800 positive or pedestrians and 5000 negative or pedestrian-free images.

(b) Image regions: 4800 and 2400 positive images along with their mirror-image form positive train and test set respectively. In addition to those 24000 and 12000 negative images forms the negative train and test set respectively. This gives us 28800 images for training and 14400 images for testing.

Some sample images are shown in Figure 1.12.

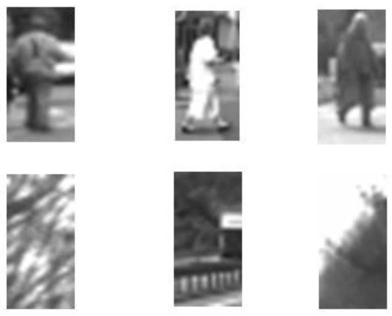


Figure 1.12 Daimler Region Images: First row Positive Pedestrian samples second row Negative Pedestrian-free Samples

❖ *Multiresolution Dataset*

- The INRIA Pedestrian dataset images were available in normalized windows of size 64 × 128. The same size was maintained while extracting negative windows. Bicubic interpolation was used to form the multiresolution dataset.
- The NICTA Pedestrian dataset contained both positive and negative pedestrian images having six resolutions: 8 × 20, 16 × 20, 16 × 40, 32 × 40, 32 × 80, and 64 × 80. Thus, we did not need to create multiresolution images, as they were already provided.
- The Daimler Pedestrian dataset contained images with a size of 18 × 36. Similar to the case of INRIA, we obtained multiresolution images via resizing.

The characteristics of the multiresolution dataset is shown in Table 1.1.

Table 1.1 Characteristics of the INRIA, NICTA, and Daimler Pedestrian Datasets

Dataset	Resolution	Image Dimension	# Positive Train images	# Negative Train images	# Positive Train images	# Negative Train images
	Single	64×128	2416	1218*10	1132	453*10
INRIA		128×256	805	406*10	377	151*10
(Color)	Multi	64×128	805	406*10	377	151*10
		32×64	806	406*10	378	151*10
	Single	64×80	12000	18000	6000	9000
		64×80	2000	3000	1000	1500
NICTA		32×80	2000	3000	1000	1500
(Color)	Multi	32×40	2000	3000	1000	1500
		16×40	2000	3000	1000	1500
		16×20	2000	3000	1000	1500
		8×16	2000	3000	1000	1500
	Single	18×36	4800	24000	2400	12000
Daimler		72×144	1600	8000	800	4000
(Gray)	Multi	36×72	1600	8000	800	4000
		18×36	1600	8000	800	4000

❖ *PASCAL VOC 2012 Dataset*

a) The PASCAL VOC 2012 dataset [7] presents standardized images and has 20 classes. Out of which one is the 'Person' class. This dataset has been widely used as a benchmark for object detection tasks.

b) The train set is taken from the trainval set containing 4087 images. The images are inclusive of occluded and truncated person images.

c) The test set contains 5138 images. The images are of various sizes.

d) The annotation files are in XML format. The annotation file includes the label and bounding box co-ordinates.

Some sample images are shown in Figure 1.13.

Figure 1.13 Sample images from PASCAL VOC 2012 Dataset

❖ ***Caltech Pedestrian Dataset***

a) The Caltech Pedestrian dataset [8] comprises of nearly 10 hours 30Hz video in 640×480 resolution. The video is captured from a vehicle driving in an urban environment through regular traffic.

b) The train data set consists of six training sets (set00-set05), each with 6-13 one-minute-long seq files, included with all the annotation information.

c) The test data set consists of five sets (set06-set10). The training images and the test images are extracted by taking every 30th frame in the seq files.

d) Thus, there are 4250 training images and 4024 test images. These images are inclusive of various levels of occlusion and height.

e) The 'Reasonable' set is considered in this work which includes images of pedestrians with height > 50 pixels and the occluded area between 1% to 35%.

Some sample images are shown in Figure 1.14.

Figure 1.14 Sample images from Caltech Pedestrian Dataset

1.11 Different Performance Measure for Comparing PD Methods

In this work, different detection metrics are used to evaluate the performance of the proposed methods. The detection metrics are categorized into per-window and per-image evaluation scheme.

The detection metrics in per-window evaluation are:

a) Confusion Matrix

For a two-class classification that is pedestrian/non-pedestrian in this case, a predicted label Positive/Pedestrian or Negative/Non-pedestrian is generated. Considering the actual label of the test dataset, a 2×2 confusion matrix [9] is obtained. From this confusion matrix, four values are derived as shown in Figure 1.15.

• *True Positive (TP):* correct classification of a pedestrian,

• *False Positive (FP)*: misclassification of a non-pedestrian,

- *False Negative (FN*): misclassification of a pedestrian and
- *True Negative (TN):* correct classification of a pedestrian.

b) Receiver Operator Characteristics Curve (ROC)

The ROC curve [10] is formed by plotting false positive per window (FPPW) against the detection rate (true positive rate). An example ROC Curve is shown in Figure 1.16.

c) Detection Error Trade-Off Curve (DET)

The DET curve [11] is formed by plotting false positive per window against miss rate. It plots error rates on a log-log scale. An example DET Curve is shown in Figure 1.17.

d) Miss Rate

The miss rate [1] is taken from the DET curve at a corresponding FPPW value.

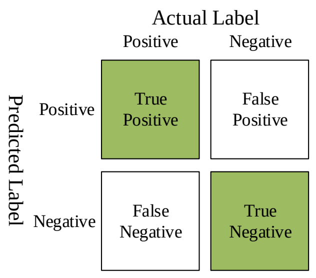


Figure 1.15 Confusion Matrix

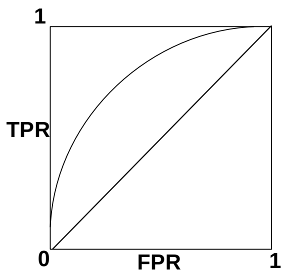


Figure 1.16 ROC Curve Example

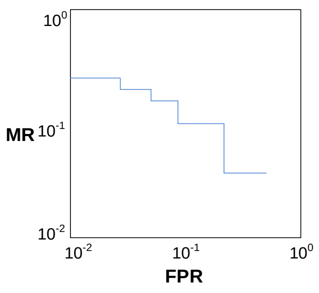


Figure 1.17 DET Curve Example

The detection metrics in per-image evaluation are:

a) Detection Error Trade-off Curve

The DET curve is formed by plotting false positive per-image (FPPI) against miss rate. It plots error rates on a log-log scale.

b) Log Average Miss Rate (LAMR)

The LAMR [12] value is calculated by averaging miss rate at nine FPPI rates evenly spaced in log-space in the range 10^{-2} to10^{-0}.

c) Precision-Recall Curve

A point on the precision-recall curve [13] is determined by considering all objects above a given model score threshold as a positive prediction, then calculating the resulting precision and recall for that threshold. An example P-R Curve is shown in Figure 1.18.

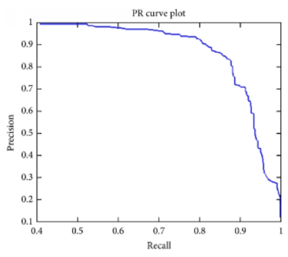


Figure 1.18 P-R Curve Example

d) Average Precision (AP)

It is defined as the mean precision at the set of 11 equally spaced recall values, $Recall_i$ = [0, 0.1, 0.2, …, 1.0] as given in Equation 1.1.

$$AP = \frac{1}{11}\sum_{\mathrm{Re}call_i} \mathrm{Pr}ecision(\mathrm{Re}call_i) \tag{1.1}$$

The Statistical Analysis conducted are: (Per-window & Per-image)

a) ***Friedman Test [14]***

- This test with a significance level of 95% and α = 0.05 is conducted to the miss rates yielded by each detection method.
- The independent variables, i.e., the methods are denoted by k. The Friedman test examines the rank of each of these variables. The null hypothesis states that all the algorithms are equivalent.
- The mean rank produced by the Friedman test for each of the pedestrian detection methods is evaluated. The chi-square is calculated as per Equation 1.2.
- The critical value of chi-square at a degree of freedom (=k-1) is found. If, the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected.

$$\aleph_F^2 = \frac{12 \times N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] and \quad R_j = \frac{1}{N}\sum_i r_i^j \tag{1.2}$$

Where, k = no. of algorithms, N = no. of datasets and r_i^j is the rank of the jth of algorithms on the ith of N datasets.

b) ***F-distribution Analysis [15]***

- The F-distribution has k-1 and (k-1)(N-1) degrees of freedom; where k and N are the numbers of algorithm and datasets, respectively.
- On application of F-distribution, the calculated F-distribution is found out as per Equation 1.3.

- The critical value of F-distribution with the degree of freedom (k-1,(k-1)(N-1)) and α = 0.05 is calculated. If the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

$$F_F = \frac{(N-1)\aleph_F^2}{N(k-1)-\aleph_F^2} \tag{1.3}$$

Where, k = no. of algorithms, N = no. of datasets and $\aleph_F^2$ is the chi-square value.

1.12 Organization of the Thesis

The rest of the chapters of this thesis are organized as follows: Chapter-2 describes the hand-crafted feature extraction algorithms, the CNN architecture along with pre-trained nets and the region proposal mechanisms involved in the pedestrian detection process. This chapter also discusses the traditional and deep learning literatures in pedestrian detection.

Chapter-3 proposes a scale-invariant shape feature extraction mechanism to explore the gradient information from all the scales of a scale-space pyramid structure of the image. It yields fixed length feature vector which serves as an advantage when processed by a classifier.

Chapter-4 advances the detection task by utilizing the proposed CNN models to improve the image's feature formation mechanism. The CNN form a more accurate feature map owing to the backpropagation error correction involved.

Chapter-5 resolved the region proposal part of the detection pipeline. It uses two-stage detection network Faster R-CNN with the proposed CNN acting as the base to predict the bounding box co-ordinates of the pedestrians. However, due to computation overhead in Chapter-6, the single-stage detection network YOLO is used wherein the base CNN is improved. The conclusion and future scope are discussed in Chapter-7.

Chapter 2

Related Work

In this chapter, some of the significant pedestrian detection methods are discussed. Existing hand-crafted features and deep learning features are given in detail. In the hand-crafted features, shape, texture and color feature extraction algorithms are explained. In the deep learning features, the CNN architectures and its usage is explained. For region proposal, existing deep learning models are given. Lastly, some of the important pedestrian detection literatures are discussed.

2.1 Feature Extraction

2.1.1 Shape Features for PD

Visual characteristics of a region or object's shape include particulars about its boundary. Shape feature descriptors encompass edge magnitude and direction, giving it a quantitative value. Histogram of Oriented Gradients (HOG) is widely used to extract shape features [1]. An introduction to HOG and its variants, HOG-18 [2] and Extended-HOG (Ex-HOG) [2] is specified in the proceeding section.

Histogram of Oriented Gradients

HOG [1] is a dense feature-extraction method. It gives information for shape and is most popularly used for pedestrian detection [16, 17]. The process of obtaining the HOG features is described in Algorithm 2.1.

Algorithm 2.1 Histogram of Oriented Gradients Feature extraction process with 9-bins

1. Take an image as input.
2. Consider a cell size from this image, as shown in Figure 2.1.
3. Compute the vertical (g_y) and horizontal (g_x) gradients over this cell by using [–1,0,1] and $[-1,0,1]^T$ filters, respectively.
4. Obtain the magnitude (M) and direction (θ) for each pixel by using Equations (2.1) and (2.2).
5. Choose a block size with four cells from the original image, as shown in Figure 2.1.
6. Obtain the 1×9 histogram bin (H) for each cell of the block by considering M and θ using Equations (2.3) and (2.4).
7. Concatenate the four 1×9 histogram bins of the whole block to form a 1×36 histogram bin.
8. Normalize the histogram bin obtained in step 7.

9. With a stride of 1, perform steps 3 to 8 for all the blocks in the image.
10. Concatenate all these histogram bins to obtain the feature vector for the whole image.

$$M(i,j) = \sqrt{{g_x}^2 + {g_y}^2} \tag{2.1}$$

$$\theta(i,j) = \tan^{-1}\left(\frac{g_y}{g_x}\right) \tag{2.2}$$

$$H\left(\left\lceil \frac{\theta(i,j)}{20} \right\rceil \operatorname{mod} 9 \times 20\right) = H\left(\left\lceil \frac{\theta(i,j)}{20} \right\rceil \operatorname{mod} 9 \times 20\right) + \left(M(i,j) \times f_1\right) \tag{2.3}$$

$$H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \operatorname{mod} 9 \times 20\right) = H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \operatorname{mod} 9 \times 20\right) + \left(M(i,j) \times f_2\right) \tag{2.4}$$

Here, $f_1 = (\theta(i,j) \operatorname{mod} 20)/20$, and $f_2 = 1 - f_1$.

The process of calculating the 9-bin histogram for a cell is explained here. As the first step, the cell size is considered as 8×8, as shown in Figure 2.2. The results for the magnitude (M) and direction (θ) of each pixel are shown in Figure 2.3 and Figure 2.4, respectively. We compare the respective magnitude (M) and direction (θ) to obtain a 1×9 vector (H). The vector H is indexed as 0°, 20°, 40°, ..., 160°. While selecting an index from the vector, the value of the direction cell is considered. After selection of an index, its value is selected from the corresponding magnitude cell. Consider the value in the blue circle of the direction cell: as the value 21.01 lies between 20 and 40 in H, the bin 20 and 40 in H are filled with 153.59 and 8.17, respectively; i.e., the corresponding magnitude value 161.76 is split between the two. Now, let us take the value in the red circle of the direction cell, which is 109.09. As it lies between indices 100 and 120, the corresponding magnitude value 55.03 is split as 30.02 in bin 100 and 25.01 in bin 120, as shown in Figure 2.5. Similarly, the whole vector H can be filled, and we obtain the final result, as shown in Figure 2.6.

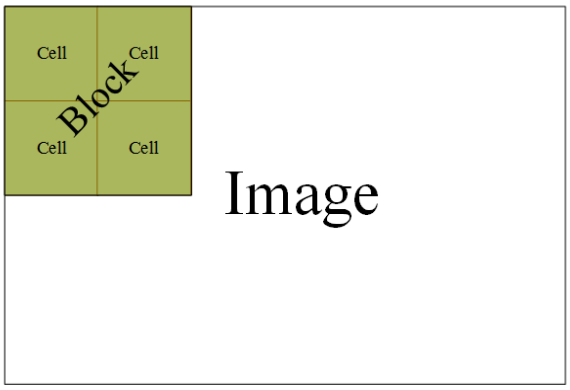
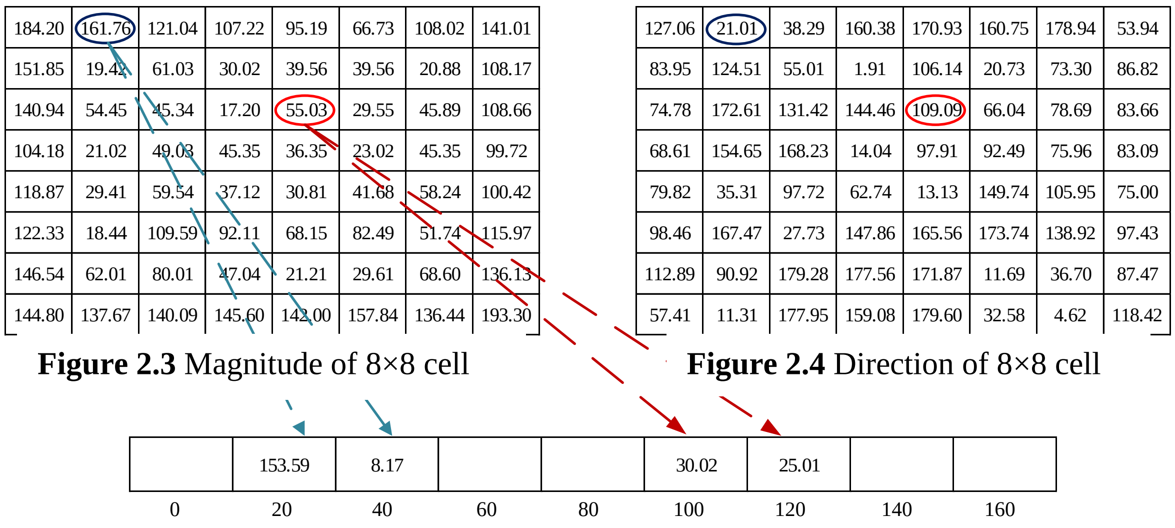


Figure 2.1 Representation of cells and block for HOG feature calculation

114	147	56	72	92	87	114	89
111	151	95	101	94	63	108	83
98	136	91	102	81	50	108	95
74	97	65	87	76	51	99	95
60	117	43	58	76	51	97	107
53	121	57	70	106	87	115	121
78	135	140	136	142	133	136	92
110	122	137	117	85	116	170	127

Figure 2.2 8×8 cell of an image



184.20	161.76	121.04	107.22	95.19	66.73	108.02	141.01
151.85	19.42	61.03	30.02	39.56	39.56	20.88	108.17
140.94	54.45	45.34	17.20	55.03	29.55	45.89	108.66
104.18	21.02	49.03	45.35	36.35	23.02	45.35	99.72
118.87	29.41	59.54	37.12	30.81	41.68	58.24	100.42
122.33	18.44	109.59	92.11	68.15	82.49	51.74	115.97
146.54	62.01	80.01	47.04	21.21	29.61	68.60	136.13
144.80	137.67	140.09	145.60	142.00	157.84	136.44	193.30

Figure 2.3 Magnitude of 8×8 cell

127.06	21.01	38.29	160.38	170.93	160.75	178.94	53.94
83.95	124.51	55.01	1.91	106.14	20.73	73.30	86.82
74.78	172.61	131.42	144.46	109.09	66.04	78.69	83.66
68.61	154.65	168.23	14.04	97.91	92.49	75.96	83.09
79.82	35.31	97.72	62.74	13.13	149.74	105.95	75.00
98.46	167.47	27.73	147.86	165.56	173.74	138.92	97.43
112.89	90.92	179.28	177.56	171.87	11.69	36.70	87.47
57.41	11.31	177.95	159.08	179.60	32.58	4.62	118.42

Figure 2.4 Direction of 8×8 cell

0	20	40	60	80	100	120	140	160
	153.59	8.17			30.02	25.01		

Figure 2.5 Process of updating the 9-bin histogram for the cell

0	20	40	60	80	100	120	140	160
920.10	527.63	418.51	464.25	972.27	666.28	483.47	247.20	602.89

Figure 2.6 Final 9-bin histogram for the cell

Histogram of Oriented Gradients (18 bins)

The difference between HOG using a 9-bin histogram (described in previous section) and that using an 18-bin histogram is that for a cell, we obtain an 18-bin histogram (0°, 20°, 40°, ..., 340°) as the direction is calculated in four-quadrant inverse tangent format. The detailed process is given in Algorithm 2.2.

Algorithm 2.2 Histogram of Oriented Gradients Feature extraction process with 18-bins

1. Take an image as input.
2. Consider a cell size from this image, as shown in Figure 2.1.

3. Compute the vertical (g_y) and horizontal (g_x) gradients over this cell by using [–1,0,1] and $[-1, 0, 1]^T$ filters, respectively.
4. Obtain the magnitude (M) and direction (θ) for each pixel by using Equations (2.1) and (2.5).
5. Obtain the histogram bin (H) by considering M and θ for all the cells using Equations (2.6) and (2.7).
6. Calculate the clipped L2-norm of the 2×2 cell with a stride of 1 and a clipping value of 0.08 to obtain a clipped histogram bin (H_c).
7. Apply the L2-norm to H_c to renormalize it and form H_{cn}.
8. Obtain the histogram bin for each cell ($H_{sum,N}$) by using Eq. (8).
9. Concatenate all the $H_{sum,N}$ values to form the final feature vector.

$$\theta(i,j) = a\tan 2\left(\frac{g_y}{g_x}\right) \tag{2.5}$$

$$H\left(\left\lceil \frac{\theta(i,j)}{20} \right\rceil \bmod 18 \times 20\right) = H\left(\left\lceil \frac{\theta(i,j)}{20} \right\rceil \bmod 18 \times 20\right) + \left(M(i,j) \times f_1\right) \tag{2.6}$$

$$H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 18 \times 20\right) = H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 18 \times 20\right) + \left(M(i,j) \times f_2\right) \tag{2.7}$$

Here, $f_1 = (\theta(i,j) \bmod 20)/20$, and $f_2 = 1 - f_1$.

$$H_{sum,\mathrm{N}}(i) = H_{cn,N}(i) + H_{cn.N}\left(i + \frac{numberofHbins}{2}\right) \tag{2.8}$$

Here, $1 \le i \le (numberofHbins / 2)$, and N represents the number of cells in a block.

Extended Histogram of Oriented Gradients

As $H_{sum,n}$ (Equation (2.8)) is calculated, $H_{diff,N}$ is similarly calculated using equation (2.9). The ExHOG is formed simply by concatenating $H_{sum,N}$ and $H_{diff,N}$ using Equation (2.10). This is explained in Figure 2.7, where the number of cells in a block is taken to be four.

$$H_{diff,N}(i) = \left| H_{cn,N}(i) - H_{cn,N}\left(i + \frac{numberofHbins}{2}\right) \right| \tag{2.9}$$

Here, $1 \le i \le (numberofHbins / 2)$, and N represents the number of cells in a block.

$$ExHOG_N = H_{sum,N} \mid\mid H_{diff,N} \tag{2.10}$$

Here, N represents the number of cells in a block.

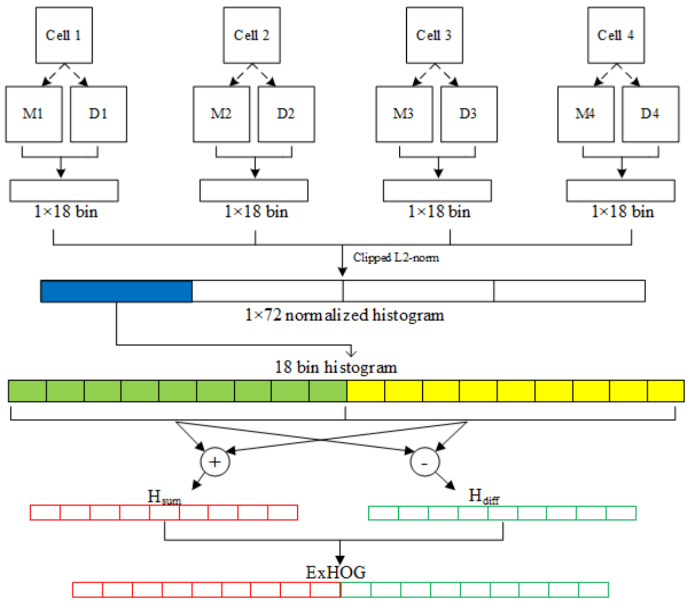


Figure 2.7 Formation of ExHOG feature vector

2.1.2 Color Features for PD

Color features are the basic characteristic of the content of images. Color features are represented in the form of color models. The color model gives measurable value to colors, in the form of a tuple, having three or four values; showing the ratio in which color components are used. In the proceeding section, color feature extraction algorithms are presented using HSI (Hue, Saturation, Intensity) color model.

Color correlogram

This feature was proposed by J. Huang et al. [18]. Using this feature, we characterized not only color distribution of pixel values but also the spatial correlation of pixel pairs, depending on the distance *k*. The D8 distance is used for distance measurement. The D8 distance between two pixels $I_1(x_1,y_1)$ and $I_2(x_2,y_2)$ is given by Equation (2.11).

$$\left|I_1(x_1,y_1)-I_2(x_2,y_2)\right|_{D_R} = MAX\left\{|x_1-x_2|,|y_1-y_2|\right\} \tag{2.11}$$

The color correlogram is represented by a square matrix S^k of dimensions N×N. The value of each cell of matrix $S^K(i,j)$ carries information about the probability of cooccurrence between a pixel having the value *i* and another pixel having the value *j* at a fixed D8 distance *k*. The color correlogram of an image *I* is given by Equation (2.12).

$$S^k(i,j)=\frac{1}{N_i\times 8k}\sum_{m=1}^{M}\sum_{n=1}^{N}1-\left[\frac{1}{2}\left(\frac{|I(m,n)-i|}{L_{\max}}\right)+\frac{1}{2}\left(\frac{|I(m+\Delta x,n+\Delta y)-j|}{L_{\max}}\right)\right] \tag{2.12}$$

$$\forall \Delta x,\Delta y\in\left\{(0,1,2,......k)\,and\,(\max(\Delta x,\Delta y)=k)\right\}\quad\text{and}\quad \forall i,j\in\left\{0,1,2,......L_{\max}\right\}$$

Here, N_i represents the color histogram [19] of i, which is given by Equation (2.13).

$$N_i = \sum_{m=1}^{M}\sum_{n=1}^{N} 1 - \left\lceil \frac{|I(m,n)-i|}{L_{\max}} \right\rceil \tag{2.13}$$

Color Autocorrelogram

As all the possible color pair combinations are considered in calculating the color correlogram, a very long feature vector is obtained. To reduce the feature-vector length, we often use a color autocorrelogram [18, 20]. The color autocorrelogram considers the spatial correlation between only identical colors. The color autocorrelogram α^k creates a one-dimensional matrix of size 1×N, where the value of each cell $\alpha^k(l)$ carries information about the probability of finding two pixels both having color l and having a D_8 distance of k, which can be represented using diagonal elements of the color correlogram matrix S^k. The color autocorrelogram of an image I is given by Equation (2.14).

$$\alpha^k(l) = S(i,j); \forall i,j,l \in \{0,1,2,\ldots,L_{\max}\} \quad and \quad i = j = l \tag{2.14}$$

2.1.3 Texture Features for PD

The texture of a region or image characterizes the presence of repetitive patterns. Th e three approaches to describe the texture of a region is statistical, structural and spectral. The texture algorithms discussed here comes under the statistical approach. Different local patterns, such as LBP [21] are used to extract texture information from an image. Figure 2.8 shows the 8-neighborhood of the central pixel I_c. Figure 2.9 shows the steps to calculate the value for central pixel. The calculation of the LBP feature vector is done by the Equation (2.15).

$$H(L)|_{LBP} = \sum_{m=1}^{M}\sum_{n=1}^{N} 1 - \left\lceil \frac{|LBP(m,n)-L|}{L_{\max}} \right\rceil ; L \in \left[0, 2^P - 1\right] \tag{2.15}$$

I_1	I_2	I_3
I_8	I_c	I_4
I_7	I_6	I_5

Figure 2.8 8-Neighborhood of pixel around I_c.

84	60	32
76	45	40
22	11	95

1	1	0
1	🟩	0
0	0	1

128	64	32
1		16
2	4	8

Figure 2.9 LBP calculation **(a)** Original Image **(b)** Resultant binary numbers for (a). **(c)** Decimal weights for the corresponding locations

2.1.4 CNN based Features for PD

CNN is one of the sub-categories of deep learning. It is used in computer vision fields like classification and detection. CNN uses convolution layers to extract hierarchical feature representation [22]. The output of each layer is known as a feature map, obtained by convolving weight matrix over receptive fields. The weight matrix or the filter is vital in identifying features. Apart from the convolution layers, a pooling layer is used, conventionally after every convolutional layer. It downsamples the feature map by applying techniques such as max-pool, min-pool and L2-norm to yield more robust feature maps [23, 24]. Another layer is added to introduce non-linearity in the system [23, 24]. This can be done by the ReLu layer which simply changes all negative activations to zero. This gives substantial speedup as opposed to tanh and sigmoid non-linear functions. In the end, fully connected layers are placed which maps the input from the preceding layer to an N-dimensional vector. N denotes the number of classes in the dataset. This vector contains the probability of the classes present.

Transfer Learning

CNN's have a huge number of weights, which are learned in the process of training. In the absence of an enormous amount of data, we cannot have an effective CNN model. This is where transfer learning comes into the picture. Transfer learning involves taking the weights learned by other pre-trained CNN and molding them into a network that is suitable for a target dataset [4]. This process is called fine-tuning the pre-trained model. There are various CNNs which are trained on millions of images from the ImageNet dataset [25]. ImageNet contains a huge number of images that covers over a thousand classes. The initial layers of these pre-trained CNNs detect low-level features like edges and blobs. These basic features are common to most of the naturally occurring datasets. Therefore, these low-level features are freezed and only the fully connected layers are trained according to the dataset requirement. The proceeding section briefly introduces the pre-trained CNNs which are used in this work.

Pre-trained CNN: AlexNet, ResNet, Inceptionv3, Xception, SqueezeNet, MobileNetv2 and Darknet

AlexNet [26]: AlexNet secured the first position in ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2012. It is mainly used for image classification problems. ImageNet dataset was used to train AlexNet. The total number of parameters and neurons of AlexNet is 60 million and 650,000 respectively. Five to six days were required to train AlexNet on the ImageNet dataset using two GTX 580 3GB GPUs.

ResNet [27]: Residual Network being a 152-layered network architecture gives a remarkable performance in classification, localization and detection problems. ResNet won ILSVRC 2015 with 3.6% error rate. An 8 GB GPU machine was used for 14 to 21 days to train ResNet. The degradation problem is resolved with the help of a deep residual learning framework. There are different variations of ResNet available having the number of layers 18, 34, 50, 101 and 152.

Inceptionv3 [28]: This is a 42-layer deep network trained on the ImageNet dataset. Several techniques are employed in this architecture to improve the performance. Some of them are factorized convolutions, smaller receptive field, grid size reduction and asymmetric convolutions. For training, a distributed system was employed using a NVIDIA Kepler GPU for 100 epochs with batch size of 32.

Xception [29]: This is a 71-layer deep network also trained on the ImageNet dataset. It can classify up to 1000 object categories. Like ResNet, this also follows a directed acyclic graph structure. The feature extraction module is formed by 36 convolution layers. It was trained on 60 NVIDIA K80 GPUs. Xception was trained on ImageNet dataset.

SqueezeNet [30]: SqueezeNet is an 18-layer deep network which has reduced the model size considerably without affecting the accuracy. SqueezeNet uses fire modules which are squeeze and expand networks. The model also comes with simple bypass and complex bypass variation. Due to the smaller size of the CNN, it was able to be trained on FPGAs.

MobileNetv2 [31]: It is a light weight model with 53 layers. It was depthwise convolution and pointwise convolution. It showed good performance in classification, detection and segmentation task. It was trained on 16 GPU asynchronous workers.

Darknet [32, 33]: This network is 19 layers deep. Darknet-19 is a series network. This network is pre-trained on the 1000 categories of the ImageNet dataset. Darknet-19 was introduced as a base network for YOLOv2 [32]. DarkNet19 has 19 convolutional layers. It uses 1×1 convolution layers to reduce the parameters in the model. It strikes a good balance between the accuracy and complexity of the model. The DarkNet19 model can be divided into six blocks considering the scale of the feature maps. Later DarkNet53 model was introduced in YOLOv3 [33]. It has 53 convolution layers of 1×1, 3×3, and 1×1 blocks. Using a Pascal Titan X system, it showed a speed of 30 frames per second.

2.2 Classifier

The SVM [34] is one of the most widely used mechanisms for solving pattern-classification problems. It is a supervised learning method that maximizes the margin of a linear decision boundary (hyperplane), thus achieving maximum separation between the two object classes.

For pedestrian classification, linear and nonlinear SVM classifiers have been used in combination with various feature sets [6, 35, 36]. Nonlinear SVM classification, e.g., using *rbf* or polynomial kernels for underlying mapping of the samples into a higher-dimensional space, results in further performance gain. However, its memory requirements and computational costs are substantially higher than those of linear SVM classification. Usually, linear SVM classifiers are preferred over nonlinear SVM classifiers to gain speed and minimize the overfitting problem [2].

2.3 Region Proposal Networks

The region proposal networks are divided into two-stage and single-stage networks. The two-stage networks are R-CNN, Fast R-CNN and Faster R-CNN. The single-stage networks are YOLOv2 and YOLOv3. Their usage in detection mechanism is discussed in this section.

- **R-CNN**

R-CNN was the first two-stage region proposal network [37]. The selective search algorithm is used in this framework to extract region proposals [38]. The selective search algorithm generates around 2000 region proposals for an image. Firstly, a feature map gets generated after running a region proposal in the object detector system. Then, two fully connected networks are used on the feature map to obtain a linear vector. This linear vector is further applied with two SVM network individually for classification and regression. For every image, the region proposal runs through the base convolutional neural network. This increases the processing time. It lacked in optimal time and space use, but provided a good starting point.

- **Fast R-CNN**

Fast R-CNN addressed the drawbacks of R-CNN [39]. Fast R-CNN reduced the computation as it processed the test image just one time. It shared the computation between the region proposal system and the detector system. It also adds a Region of Interest pooling layer which yields a fixed-length feature vector for the regions. It also performed regression and classification simultaneously. This improved the time and memory requirements when compared to R-CNN. However, there was still an obstacle, which is the selective search for region proposal.

- **Faster R-CNN**

Faster R-CNN was published in 2015 and it is the most widely used region proposal method [40]. The detection paradigm is made up of 1) a region proposal algorithm, employed to output the probable regions of the desired objects followed by 2) feature extraction by the pre-trained CNNs, 3) classification layer to predict the object’s class and 4) regression stage to fine-tune the bounding boxes generated by the region proposal method. The previous R-CNN versions used the selective search algorithm. But Faster R-CNN speeds up this process by including a convolutional network for region proposal termed as the Region Proposal Network (RPN). Faster R-CNN uses a pre-trained CNN as a backbone network which enables it to share weights with the CNN used in the detection process.

- **YOLOv2**

It is proposed as an improvement on YOLOv1 [32,41]. Some of the several advances in YOLOv2 are batch normalization, anchor boxes with k-means clustering, and a base CNN network DarkNet19 to extract features. The feature extraction layers for YOLOv2 with base DarkNet19 is shown in Table 2.1.

Table 2.1 YOLOv2 with base DarkNet19 Feature Extraction Layers

Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
Input				256×256×3
Conv1	3×3/1	32	Input	256×256×32
Maxpool_1	2×2/2		Conv1	128×128×32
Conv2	3×3/1	64	Maxpool_1	128×128×64
Maxpool_2	2×2/2		Conv2	64×64×64
Conv3	1×1/1	128	Maxpool_2	64×64×128
Conv4	3×3/1	64	Conv3	64×64×64
Conv5	1×1/1	128	Conv4	64×64×128
Maxpool_3	2×2/2		Conv5	32×32×128
Conv6	1×1/1	256	Maxpool_3	32×32×256
Conv7	3×3/1	128	Conv6	32×32×128
Conv8	1×1/1	256	Conv7	32×32×256
Maxpool_4	2×2/2		Conv8	16×16×256
Conv6	1×1/1	512	Maxpool_4	16×16×512
Conv7	3×3/1	256	Conv6	16×16×256
Conv8	1×1/1	512	Conv7	16×16×512
Conv9	3×3/1	256	Conv8	16×16×256
Conv10	1×1/1	512	Conv9	16×16×512
Maxpool_5	2×2/2		Conv10	8×8×512
Conv11	1×1/1	1024	Maxpool_5	8×8×1024
Conv12	3×3/1	512	Conv11	8×8×512
Conv13	1×1/1	1024	Conv12	8×8×1024
Conv14	1×1/1	512	Conv13	8×8×512
Conv15	3×3/1	1024	Conv14	8×8×1024

- **YOLOv3**

YOLOv3 shows further improvement on YOLOv2. It uses regression to compute the score for each prediction. YOLOv3 uses an improved CNN as the base network, DarkNet53. This network has 53 layers which are made up of 1×1, 3×3, and 1×1 blocks added with bypass [33]. YOLOv3 generates prediction from different scales of the network. In this work, two scales are considered and each scale has 3 anchor boxes. So, the output tensor of YOLOv3 is N×N×[3*(4+1+1)] considering 4 bounding box location offset, 1 predicted object, and 1 class prediction. The DarkNet53 model can be divided into five blocks considering the scale of the feature maps. The feature extraction layers for YOLOv3 with base DarkNet53 is shown in Table 2.2.

Table 2.2 YOLOv3 with base DarkNet53 Feature Extraction Layers

	Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
	Input				256×256×3
	conv1	3×3/1	32	Input	256×256×32
	conv2	3×3/2	64	conv1	128×128×64
1×	conv3	1×1/1	32	conv2	128×128×32
	conv4	3×3/1	64		128×128×64
	Residual_1				128×128×64
	conv5	3×3/2	128	Residual_1	64×64×128
2×	conv6	1×1/1	64	conv5	64×64×64
	conv7	3×3/1	128		64×64×128
	Residual_2				64×64×128
	conv10	3×3/2	256	Residual_2	32×32×256
8×	conv11	1×1/1	128	conv10	32×32×128
	conv12	3×3/1	256		32×32×256
	Residual_3				32×32×256
	conv27	3×3/2	512	Residual_3	16×16×512
8×	conv28	1×1/1	256	conv27	16×16×256
	conv29	3×3/1	512		16×16×512
	Residual_4				16×16×512
	conv44	3×3/2	1024	Residual_4	8×8×1024
4×	conv45	1×1/1	512	conv44	8×8×512
	conv46	3×3/1	1024		8×8×1024
	Residual_5				8×8×1024

2.4 Literature on PD

In computer vision applications such as object detection, medical imaging and robotics there is a major role of deep learning. Substantial amount of advancement has been made in the field of computer vision. M.M. Badza et al. [42] developed a new CNN architecture for classifying brain tumor types into three classes. The k-fold cross-validation approach was used to train the model. J. Ker et al. [43] reviewed the usage of deep learning in medical image processing which was focussed on CNNs. The review describes the advantages of machine learning algorithms compared to the hand-crafted features. Applications such as classification, localization, detection and segmentation are covered in this review. S.S. Yadav et al. [44] explores machine learning and deep learning methods for classification of pneumonia from chest X-rays. It also studies the effects of data augmentation, transfer learning and network complexity as per the application requirement. Y. Liu et al. [45] have proposed a deep ensemble network comprising of a CNN and an Attention-Guided Network to detect Glaucoma using stereo images. S.P. Singh et al. [46] have used CT scans to train a 3D CNN model into detecting four subclasses of brain hemorrhage. The network proposed in the paper can be used to detect abnormalities in different organs. J. Ker et al. [47] have propose another 3D CNN model for two-class and four-class classification of brain hemorrhage using CT scans with additional image thresholding to improve the accuracy of the network. S.P. Singh et al. [48] discusses the application of 3D CNNs in medical imaging. Areas such as segmentation, detection, localization and classification are discussed along with the challenges associated with 3D CNN in medical imaging. J. Ker et al. [49] used pre-trained CNN to detect brain tumor by successfully applying transfer learning. The authors have also collected brain histology images dataset. A. Dhillon et al. [50] has reviewed CNN architectures in object detection domain including human detection. Various pre-trained CNNs are described along with a summary of datasets, applications and the accuracy obtained. R. Li et al. [51] has proposed a situation aware framework using Electroencephalography (EEG) and machine learning.

In this thesis, a particular application i.e., pedestrian detection is focussed upon. Pedestrian detection is the task of identifying and marking the location of the presence of a pedestrian in an image [52]. It has several applications like smart surveillance [53], robotics [54], automatic driving [55] and human behavior analysis [56]. To improve pedestrian detection performance, researchers have endeavored into a varying amount of work [57,58]. Even so, a huge variation persists in poses, occlusions, viewpoints and illumination. This increases the difficulty in devising a perfect method. Thus, in recent years, this area has attracted quite a lot of attention [58, 59]. Pedestrian Detection involves a pipeline of feature extraction and classification. Features should have a well-formed and robust representation of the objects i.e., pedestrians in this case. Even though extensive research has been performed on pedestrian detection, significant improvements were made in recent works, which suggests that the research has not reached a saturation point. There is considerable literature related to pedestrian detection methods. Several methods for feature extraction have been proposed, including Edge Templates [60], the Haar Wavelet [61], Histogram of Oriented Gradients (HOG) [1], the covariance descriptor [62], Shape Models [63] and SIFT descriptors [64]. In a major breakthrough, N. Dalal et al. [1] proposed the HOG for extracting shape features. It is a dense representation of gradient information for a region. It is invariant to slight changes in translation and rotation. Local normalization helps in illumination changes. They also introduced a new annotated pedestrian dataset called INRIA with a varying background and pose. A. Satpathy et al. [2] modified the HOG using an 18-bin histogram, yielding the extended histogram of oriented gradients (ExHOG). ExHOG solves an issue of HOG wherein gradients of opposite directions in the same cell are assigned to the same histogram bin. Both linear and nonlinear kernel support vector machines (SVM) were used, and ExHOG with the nonlinear kernel performed better for the INRIA and Daimler datasets. S. Nigam et al. [65] proposed a multiresolution approach for detecting pedestrians using LBPs. This approach has two limitations: 1) it can be applied to only grayscale images and 2) images must be resized to a fixed size scale. Consequently, the multiresolution property of the dataset is lost. G. Overett et al. [5] introduced a multiresolution dataset (NICTA) of >25551 pedestrians, which gives a total of 50000 pedestrians, including left and right reflection. The negative set was sampled from 5207 high-resolution people-free images. However, the authors focused on single-resolution image sets. J. Yan et al. [66] proposed a multiresolution approach for traffic scenes. It employs a deformable part model [67] to map low-resolution and high-resolution pedestrians onto a common space. The detector then learns from these mapped features of different resolutions. However, correctly identifying true or false positives requires vehicle–pedestrian localization, assuming that in traffic scenes, pedestrians are around vehicles. Thus, more complexity is introduced to the system. P. Dollar et al. [12] evaluated the state-of-the-art pedestrian detection methods. Detection was performed at three scales: far, medium, and near. The images were captured using a camera mounted on a vehicle. There was visible degradation for the far and near scales. P. Hurney et al. [68] combined HOG with a texture feature, i.e., the local binary pattern (LBP), for grayscale pedestrian images. Feature vectors of LBP variations with 16 & 8 neighborhoods and radii of 2 & 1, respectively, were obtained. The feature vectors were given to a radial basis function (rbf)-kernel SVM. The results indicated that the HOG with an LBP having neighborhood 8 and radius 1 outperformed others. M. Bilal et al. [69] used integer-only features from color information and orientation histograms. Classification is done by implementing a soft cascade for fast evaluation of kernel classifier. The authors could identify true negatives at the early stages from the kernel function's energies. R. Lahmyed et al. [70] proposed to use both visible and thermal image of a scene in pedestrian detection system. They have used a modification of OTSU method to segment the thermal images in order to get the locations of probable pedestrians. The locations get mapped to visible images, thereafter features are calculated. B.T. Bastian et al. [71] proposed to merge data specific dictionary learned histogram of sparse codes and aggregate channel features for pedestrian detection. K. Kumar et al. [72] proposed to combine Histogram of Significant Gradients, a variation of HOG with Non-Redundant Uniform Local Binary Pattern to yield a feature descriptor. The authors then used a linear SVM classifier for feature training. X. Zhanga et al. [73] proposed a pedestrian detection method which combines HOG features with the color image's edge features on depth images. They have used shearlet transform to yield edge features from the color images. The combined feature is used to train the SVM classifier. However, the existing hand-crafted feature methods are unable to bridge the gap between the current performance and the preferred performance.

A significant improvement occurred when Convolutional Neural Network (CNN) was introduced [3,74,75]. CNN is a deep learning architecture and it can learn complex feature representations. Also, the backpropagation training algorithm allows for a better feature formation mechanism then manually designed features [3]. Since the advent of AlexNet, a pretrained CNN model on the ImageNet dataset [26], many CNN models have been proposed for detection purposes [27,76,77]. The researchers have improved the feature formation process in the CNNs, such as skip connection in ResNet50 and inception module in GoogleNet [78]. In recent years, there is a tremendous amount of research in pedestrian detection using deep learning. D. Tome et al. have used a deep convolutional neural network to improve accuracy for pedestrian detection. They have used pre-trained AlexNet and GoogleNet with optimization [79]. W. Ouyang et al. has proposed the use of the deformable parts model and deep learning to handle occlusion. For the visibility relationship between the parts, a deep model is proposed [80]. Q. Hu et al. have used the features produced by the deep learning CNN models and used them to train decision models. They have also added a manually designed optic flow feature to the Deep CNN features [81]. L. Chunze et al. used deep features to detect visible parts of an occluded pedestrian as well as small sized ones. A multi-scale network is used where the feature maps are selected based on the target pedestrian's size and the corresponding receptive field of the feature map [82]. L. Jianan et al. have used a Fast R-CNN network for detecting pedestrians using subnetworks to extract features at various scales [83]. L. Chunze et al. proposed a deep learning structure to detect pedestrians at various scales. They have used an attention map with the feature maps to detect pedestrians at respective scales [84]. A method based on convolutional sparse coding is used by P. Sermanet et al. [77] to pre-train the CNN for pedestrian detection. The combination of CNN and hand-crafted features are used to build a complexity-aware cascaded detector by J. Hosang [85]. Pedestrian detection and other semantic tasks are optimized jointly by Y. Tian et al. [86].

A deep learning object detection can be divided into two-stage and single-stage methods. In a two-stage process, the first stage develops region proposals, and the second stage classifies them. The two-stage network R-CNN [37] worked by applying CNN features on the generated object proposals. Faster R-CNN [40] added a region proposal network which increased the accuracy. In a one-stage method, the extraction of region proposal is eliminated, and the detection results are created from the images directly. Single Shot Multibox Detector (SSD)

[87], a single-stage detection method, has a significantly high speed. The YOLO model [41] is the first single-stage deep learning method. It models the detection task as an end-to-end regression problem, thus gaining more speed. The basic YOLO was improved with the addition of anchor boxes to predict the bounding boxes. The improved version named YOLOv2 [32] also introduced a new pre-trained CNN model, DarkNet-19. Z. Yi et al. proposed a modified tiny-yolov3 [33] is proposed to improve the feature formation process by adding three convolutional layers to improve the detection accuracy [88]. W. Lan et al. adds a passthrough layer in YOLOv2 to link the high- and low-resolution feature maps [89]. Z. Liu et al. has also added passthrough layers and fusion layers to YOLOv2, combining the feature across all levels [90]. W.Y. Hsu et al. proposes a ratio-aware YOLO model which uses multiresolution fusion to lower the miss rate in small and varying aspect ratio pedestrians [91]. X. Tang et al. has given a scale-aware YOLOv3 modification using two sub-networks to improve small-scale pedestrian detection in real-time [92].

Chapter 3

A Scale Invariant Histogram of Oriented Gradients Feature Extraction for Pedestrian Detection in Multiresolution Image Dataset

In this chapter, a hand-crafted feature extraction mechanism is proposed for pedestrian detection. The proposed method adapts feature extraction methods for a scale independent system. This method achieves the following two objectives. It can process multiresolution or single resolution images. It extracts shape features by introducing a scale-space in HOG. Weighted gradient information of the scale-space ensures that the HOG feature vector is independent of resolution. 2) Previously, there has been no fusion of shape features, texture features and color features with the SVM classifier. Therefore, we combine the proposed method, i.e., Scale-Invariant Histogram of Oriented Gradients (SI-HOG), which gives the scale-independent shape feature, with texture and color features. As shape features are the basis for pedestrian detection, they are concatenated with texture and color features.

3.1 Introduction

3.1.1 Histogram of Oriented Gradients Feature Pyramid

To perform pedestrian detection, first, feature extraction is performed. Because we are focusing on pedestrians, the most important feature is shape. For extracting shape information, the best-performing feature is HOG. A dataset containing images of different resolutions (dimensions) produces HOG feature vectors of different lengths, this is because HOG features are dependent on the size of the image. This becomes an obstacle for the classifier. If we wish to apply HOG, all the images in the dataset must be resized to the same dimension. Hence, the multiresolution property is lost. We address this problem by utilizing a scale-space pyramid to extract shape features, which are not dependent on the dimension of the image.

3.2 Methodology

In the proposed method, a scale-space pyramid is utilized to extract the shape features, which are not dependent on the scale of the image. The SI-HOG method is described in Subsection 3.2.1 and the mathematical derivation of the SI-HOG feature vector is given in Subsection 3.2.2.

3.2.1 Scale Invariant Histogram of Oriented Gradients

The proposed SI-HOG method for pedestrian detection is described in two parts i.e., Feature Extraction and Classification.

Part I (Feature Extraction)

1. Consider three scales to construct a pyramid structure by placing the image at the bottom (Scale$_1$) and then subsampling it with a factor of two to create another image (Scale$_2$). The process is repeated on Scale 2 to obtain the third image (Scale$_3$). For Scale 1, take the largest dimension R_n, the maximal resolution from the set, having a resolution, let's say, p×q. The resolution for Scale 2 is p/2×q/2, and that for Scale 3 is p/4×q/4.
2. For each image, a scale-space (S_j, $1 \leq j \leq 3$) pyramid is constructed with the three resolutions, as described in Step 2.
3. For Scale 1, consider the cell size as m×m. For Scale 2, the cell size is m/2×m/2, and for Scale 3, it is m/4×m/4. Thus, the same number of cells is maintained for all the scales.
4. Obtain a 9-bin histogram (H) using Equations (3.1) and (3.2) for every cell in a scale S_j, with $1 \leq j \leq 3$.
5. Concatenate the histogram bins obtained in Step 4 for each scale S_j to form $H_{scale,j}$, where $1 \leq j \leq 3$.
6. Take the average of $H_{scale,1}$, $H_{scale,2}$, and $H_{scale,3}$ according to Equation (3.3).
7. Considering 2×2 overlapping cells with a stride of 1, apply block normalization on H_{avg} to obtain the final feature vector for the image.

Part II (Classification)

1. The feature vectors of both positive and negative images (with labels 1 and –1, respectively) are used to train the SVM.
2. The Test Set feature vector including the positive and negative images is given to the trained SVM Model to obtain a label (either 1 or –1).
3. The actual and predicted labels of the test set are compared to obtain a confusion matrix

The proposed method is represented in Figure 3.1.

$$H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 9 \times 20\right) = H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 9 \times 20\right) + (M(i,j) \times f_1) \qquad (3.1)$$

$$H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 9 \times 20\right) = H\left(\left\lfloor \frac{\theta(i,j)}{20} \right\rfloor \bmod 9 \times 20\right) + (M(i,j)\times f_2) \tag{3.2}$$

$$H_{avg} = \sum_{j=1}^{3} H_{scale,j} / 3 \tag{3.3}$$

Where, the size of $H_{scale,j} = 9 \times \frac{p*q}{m*m}$

3.2.2 Derivation of SI-HOG Feature Vector

1. The concatenated Histogram ($H_{scale,\ j}$) of all the scales (S_j) is formed as:

- *Scale 1*: Resolution of image is p×q and the cell size is m×m. So, we get (p*q)/(m*m) number of cells. For each cell, a 9-bin histogram is formed. So, for scale 1, 9×((p*q)/(m*m)) (=$H_{scale,1}$) is yielded by concatenating each 9-bin histogram vertically.

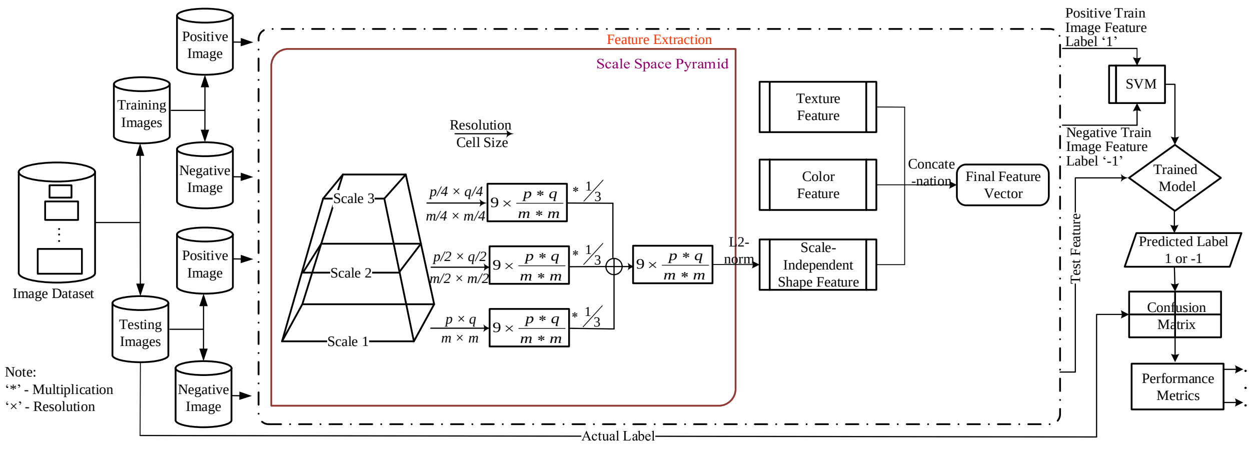


Figure 3.1 Block Diagram of the Proposed SI-HOG

- *Scale 2*: Resolution of image is p/2×q/2 and the cell size is m/2×m/2. So, we get (p/2*q/2)/(m/2*m/2) number of cells which results into (p*q)/(m*m) cells. Similarly, for each cell, a 9-bin histogram is formed. Finally, for scale 2 also, 9×((p*q)/(m*m)) (=$H_{scale,2}$) is yielded by concatenating each 9-bin histogram vertically.
- *Scale 3*: Resolution of image is p/4×q/4 and the cell size is m/4×m/4. So, we get (p/4*q/4)/(m/4*m/4) number of cells which results into (p*q)/(m*m) cells. Similarly, for each cell, a 9-bin histogram is formed and 9×((p*q)/(m*m)) (=$H_{scale,3}$) is yielded by concatenating each 9-bin histogram vertically.

2. As all the Histogram of the scales are given equal weightage, $H_{scale,1}$, $H_{scale,2}$ and $H_{scale,3}$ each are multiplied by a factor of ⅓ and then summed to get H_{avg}. Size of H_{avg} will be equal to that of $H_{scale,1}$, $H_{scale,2}$ and $H_{scale,3}$ i.e., 9×((p*q)/(m*m)).

3. As per the construction of a block comprising of four cells, H_{avg} is block normalized following L2-norm to yield the final feature vector. The length of the final feature vector is given in Equation (3.4).

$$LengthOfSI - HOGFeatureVector = NumberOfBlocks * LengthOfEachBlock \tag{3.4}$$

   Where, *number of blocks* = $((p/m) - 1)*((q/m) - 1)$ and

   length of each block = *number of cells per block*number of bins* = 4*9 = 36
4. The feature vector yielded from Positive Training Set is assigned label '1' and that of Negative Training Set is assigned label '-1'.
5. The feature vector along with the label vector from the training set is used to train SVM, which yields a model.
6. The Positive Testing Set and the Negative Testing Set along with their actual labels are given as input to SVM model.
7. The output is a set of predicted labels which is further used to construct confusion matrix and the performance metrics.

3.3 Experimental Results and Discussion

The proposed method is compared with twelve existing algorithms on three standard pedestrian datasets: INRIA, NICTA and Daimler. The details of these pedestrian datasets are given in Chapter-1. Both the single and multiresolution version of the datasets are considered in this work. For the three datasets, per-window evaluation was employed. The value of *m*, which determines the cell size for the scale-space, was 16, 8 and 12 for INRIA, NICTA and Daimler, respectively. These values were experimentally determined to perform well for the datasets. For INRIA, a linear SVM classifier with a regularization parameter (*C*) of 0.01 was used, whereas for NICTA and Daimler, a nonlinear SVM classifier with a *rbf* kernel was used. The classification was performed using the LIBSVM machine-learning library. Subsection 3.3.1, 3.3.2 and 3.3.3 discusses the performance for the INRIA, NICTA and Daimler pedestrian datasets respectively. Subsection 3.3.4 shows the fusion strategies result. Subsection 3.3.5 shows the statistical analysis of the results.

3.3.1 INRIA Pedestrian Dataset

The proposed method is applied on the INRIA Pedestrian dataset and the confusion matrix is obtained from the classification result. Thus, the ROC Curve and DET Curve are derived from the confusion matrix result. The ROC Curve for single and multiresolution INRIA is shown in Figure 3.2 and Figure 3.3 respectively. The DET Curve for the single and multiresolution INRIA is shown in Figure 3.4 and Figure 3.5 respectively. The miss rate is obtained at 10^{-3} FPPW and given in Table 3.1. In the single resolution case, the proposed method achieved the lowest miss rate of 7.47%. The second lowest miss rate was achieved by HOG-9. Thus, the proposed method was the best-performing method when compared with other shape features as well as texture and color features. In the multiresolution case, the proposed method was the only shape feature-extraction method. With the addition of texture and color information, miss rates of 6.98% and 40.24% were achieved for INRIA. For INRIA, combining texture and color features with the proposed method further improved the miss rate. With the addition of texture and color information, miss rates of 3.28% and 6.98% is yielded for single and multi -resolution respectively.

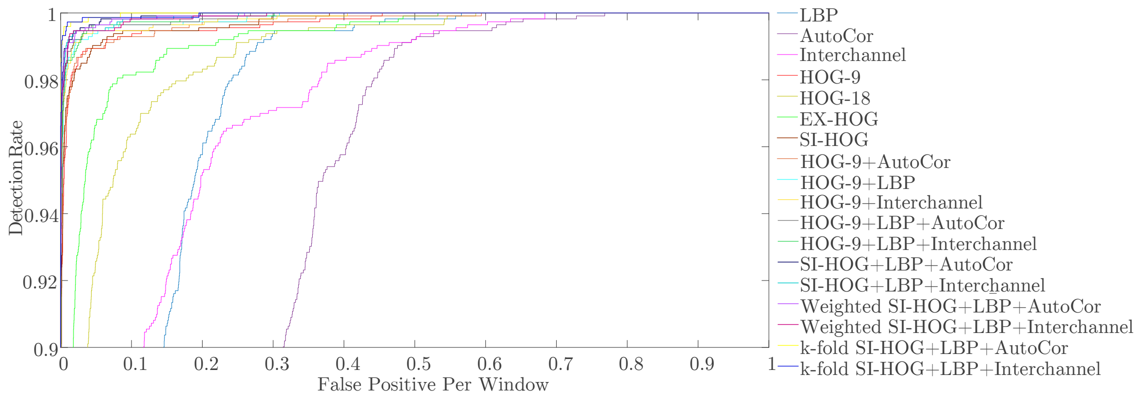


Figure 3.2 The ROC Curve for Single Resolution INRIA Pedestrian Dataset

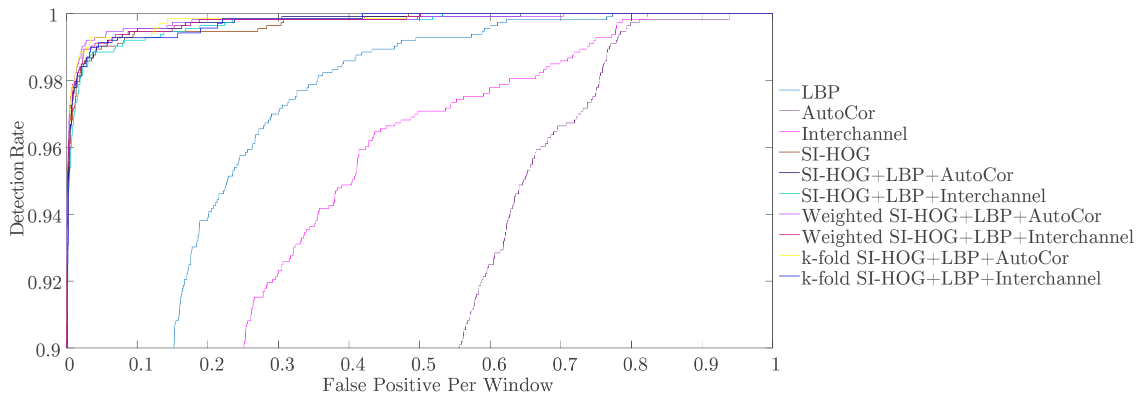


Figure 3.3 The ROC Curve for Multiresolution INRIA Pedestrian Dataset

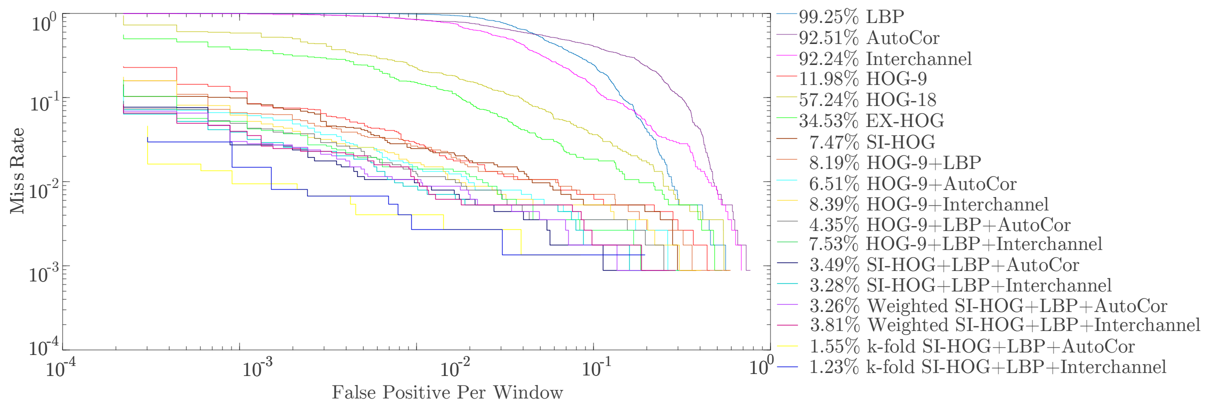


Figure 3.4 The DET Curve for Single Resolution INRIA Pedestrian Dataset

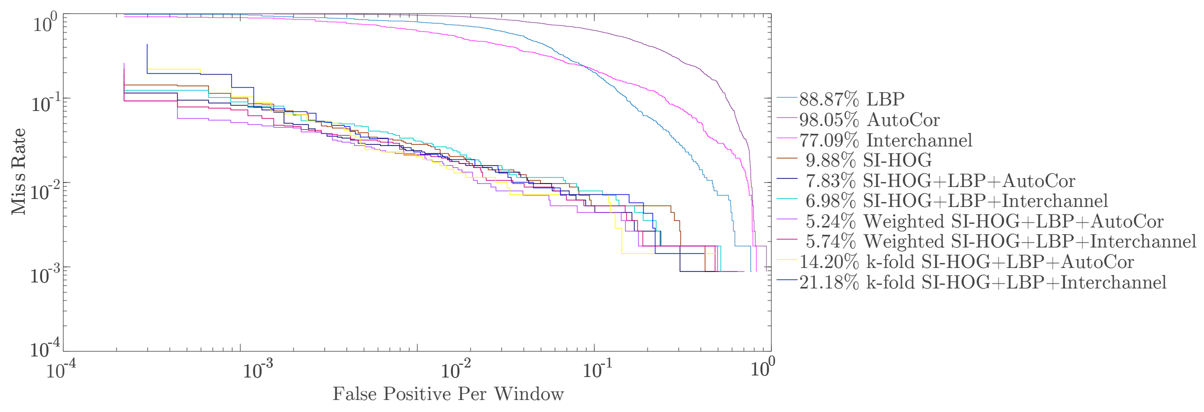


Figure 3.5 The DET Curve for Multiresolution INRIA Pedestrian Dataset

Table 3.1 Miss Rate values for Single and Multiresolution INRIA Pedestrian Dataset

Method	**Miss Rate (%)**	
	INRIA	
	Single Resolution	**Multi-resolution**
LBP	99.25	88.87
AutoCor	92.51	98.05
Interchannel	92.24	77.09
HOG-9	11.98	NA
HOG-18	57.24	NA
Ex-HOG	34.53	NA
SI-HOG	**7.47**	**9.88**
HOG-9+LBP	8.19	NA
HOG-9+AutoCor	6.51	NA
HOG-9+Interchannel	8.39	NA
HOG-9+LBP+AutoCor	4.35	NA
HOG-9+LBP+Interchannel	7.53	NA
SI-HOG+LBP+AutoCor	**3.49**	**7.83**
SI-HOG+LBP+Interchannel	**3.28**	**6.98**
Weighted SI-HOG+LBP+AutoCor	**3.26**	**5.24**
Weighted SI-HOG+LBP+Interchannel	**3.81**	**5.74**
k-fold SI-HOG+LBP+AutoCor	**1.55**	**14.2**
k-fold SI-HOG+LBP+Interchannel	**1.23**	**21.18**

3.3.2 NICTA Pedestrian Dataset

The proposed method is applied on the NICTA Pedestrian dataset to obtain the confusion matrix from the classification result. This yielded the ROC Curve and DET Curve from the confusion matrix result. The DET Curve for single and multiresolution NICTA is shown in Figure 3.6 and Figure 3.7 respectively. The miss rate is obtained at 10^{-3} FPPW and given in Table 3.2. In the single resolution case, the proposed method achieved the lowest miss rate of 20.56% for NICTA. The proposed method was the best-performing method when compared with other shape features as well as texture and color features. In the multiresolution case, the proposed method was the only shape feature-extraction method. With the addition of texture and color information, miss rates of 40.24% were achieved for NICTA. For NICTA, combining texture and color features with the proposed method further improved the miss rate. With the addition of texture and color information, miss rates of 14.65% and 40.24% is yielded for single and multiresolution respectively.

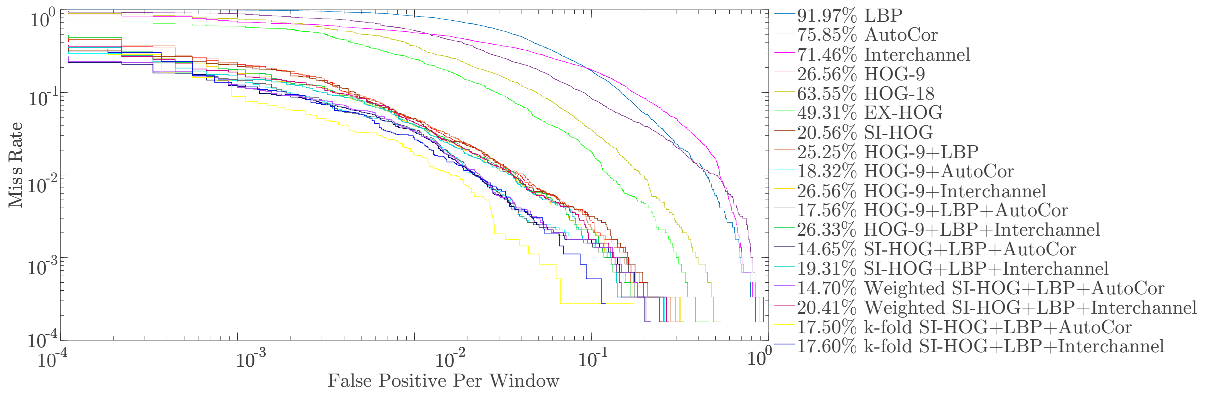


Figure 3.6 The DET Curve for Single Resolution NICTA Pedestrian Dataset

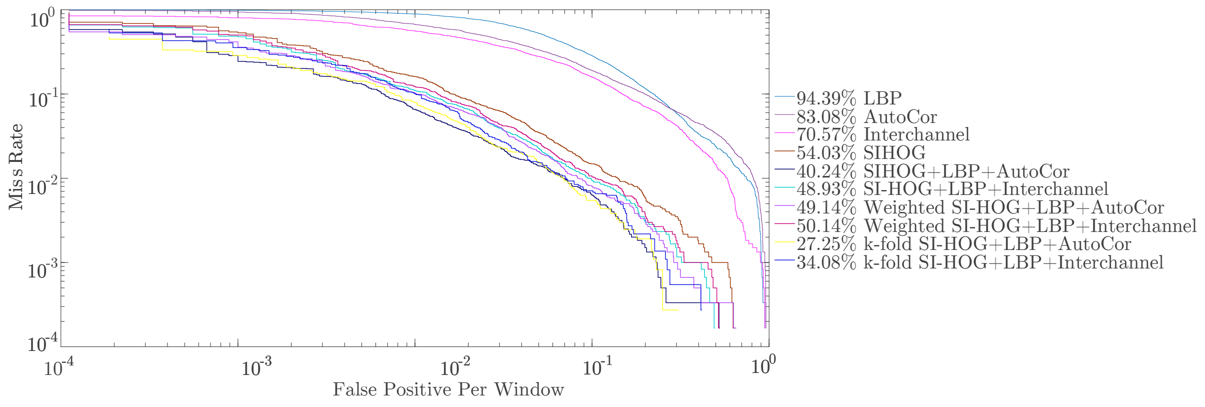


Figure 3.7 The DET Curve for Multiresolution NICTA Pedestrian Dataset

Table 3.2 Miss Rate values for Single and Multiresolution NICTA Pedestrian Dataset

Method	**Miss Rate (%)**	
	NICTA	
	Single Resolution	**Multi-resolution**
LBP	91.97	94.39
AutoCor	75.85	83.08
Interchannel	71.46	70.57
HOG-9	26.56	NA
HOG-18	63.55	NA
Ex-HOG	49.31	NA
SI-HOG	**20.56**	**54.03**
HOG-9+LBP	25.25	NA
HOG-9+AutoCor	18.32	NA
HOG-9+Interchannel	26.56	NA
HOG-9+LBP+AutoCor	17.56	NA
HOG-9+LBP+Interchannel	26.33	NA
SI-HOG+LBP+AutoCor	**14.65**	**40.24**
SI-HOG+LBP+Interchannel	**19.31**	**48.93**
Weighted SI-HOG+LBP+AutoCor	**14.7**	**49.14**
Weighted SI-HOG+LBP+Interchannel	**20.41**	**50.14**
k-fold SI-HOG+LBP+AutoCor	**17.5**	**27.25**
k-fold SI-HOG+LBP+Interchannel	**17.6**	**34.08**

3.3.3 Daimler Pedestrian Dataset

The proposed method is applied on the Daimler Pedestrian dataset and from the classification result and confusion matrix, the ROC Curve and DET Curve are obtained. The DET Curve for single and multiresolution Daimler is shown in Figure 3.8 and Figure 3.9 respectively. The miss rate is obtained at 10^{-2} FPPW and given in Table 3.3. As it is a grayscale dataset, the color feature is not considered here. In the single-resolution case, the proposed method achieved the lowest miss rate of 33.20%. The second lowest miss rate was achieved by HOG-9. Thus, the proposed method was the best-performing method when compared with other shape features as well as texture feature. In the multiresolution case, the proposed method was the only shape feature-extraction method. With the addition of texture information, a miss rate of 33.21% is yielded. For Daimler, combining texture feature with the proposed method further improved the miss rate. With the addition of texture information, miss rates of 32.67% and 33.31% is yielded for single and multiresolution respectively.

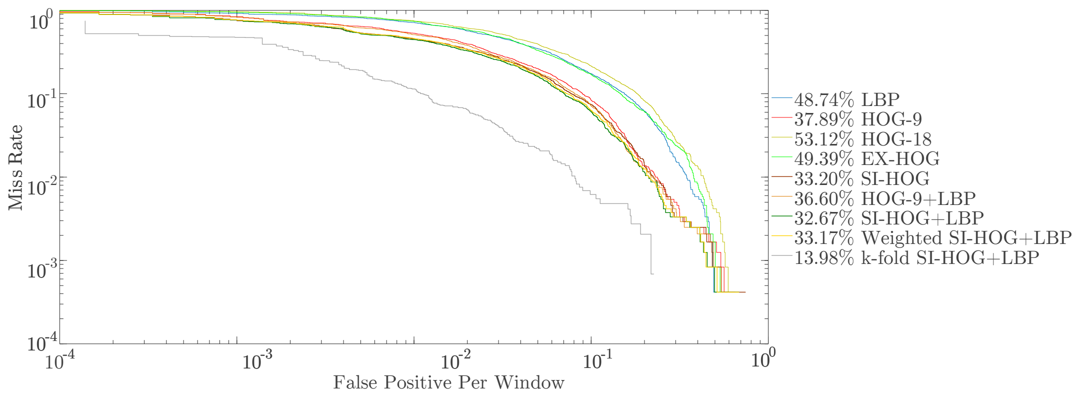


Figure 3.8 The DET Curve for Single Resolution Daimler Pedestrian Dataset

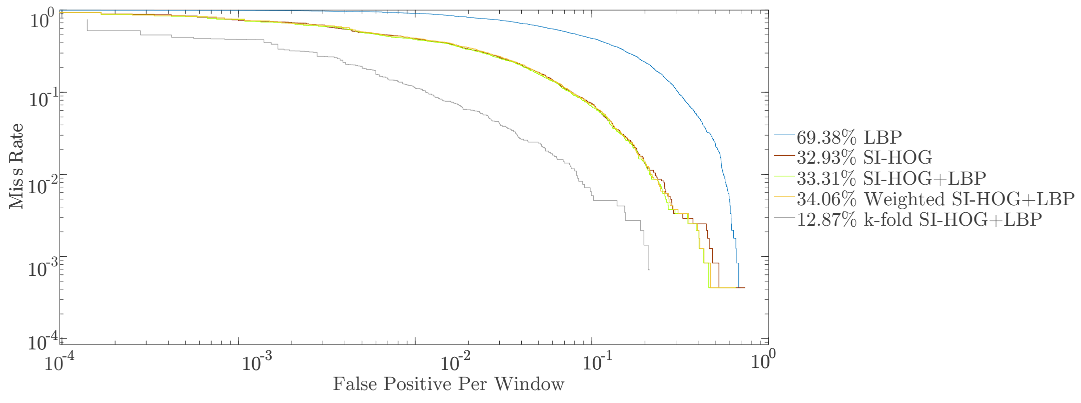


Figure 3.9 The DET Curve for Multiresolution Daimler Pedestrian Dataset

Table 3.3 Miss Rate values for Single and Multiresolution Pedestrian Dataset

Methods	**Miss Rate (%)**	
	Daimler	
	Single Resolution	**Multi-resolution**
LBP	48.74	69.38
HOG-9	37.89	NA
HOG-18	53.12	NA
Ex-HOG	49.39	NA
SI-HOG	**33.20**	**32.93**
HOG-9+LBP	36.60	NA
SI-HOG+LBP	**32.67**	**33.31**
Weighted SI-HOG+LBP	**33.17**	**34.06**
k-fold SI-HOG+LBP	**13.98**	**12.87**

3.3.4 Result of Fusion Strategies

Two fusion strategies namely weighted and k-fold SI-HOG+LBP+AutoCor and SI-HOG+LBP+Interchannel are evaluated for INRIA, NICTA and Daimler single and multiresolution datasets. In the case of weighted methods, the weight was taken with respect to the ratio of (TP+TN) to (TP+FN+FP+TN) of each method. The weight was multiplied by their respective train and test feature vectors to form weighted feature vectors. In the case of k-fold, the total of the train-test set was divided into k parts where k=5. Then the performance was evaluated on the train:test set in the ratio of 4:1 part. In the case of weighted fusion, there is an improvement in miss rate in the single and multiresolution INRIA dataset and in the single resolution NICTA dataset. In the case of k-fold fusion there is a significant improvement of miss rate in the single resolution INRIA and the multiresolution NICTA datasets and also in both the single and multiresolution Daimler datasets. The fusion strategies results are shown in their respective ROC, DET Curves and miss rate tables.

3.3.5 Statistical Analysis

In this work, there are 14 and 6 independent variables (k) in single and multiresolution dataset (N) respectively. The rank table for Friedman test is given in Table 3.4. The proposed concatenation i.e., SI-HOG+LBP+AutoCor and SI-HOG+LBP+Interchannel has the first and second rank respectively in the case of single resolution. The calculated chi-square is 25.557. The critical value of chi-square at a degree of freedom (k-1) 13 is 21.026. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. In the case of multiresolution both have the first rank. The calculated chi-square is 9.428. The critical value of chi-square at degree of freedom 5 is 11.070. As the calculated chi-square is less than the critical chi-square, the Null Hypothesis is failed to be rejected. The calculated F-distribution is 16.482. The critical value of F-distribution with degree of freedom (5,5) and α = 0.05 is 5.05. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

Table 3.4 Rank Table of Friedman Test for the Proposed Methods in Single and Multiresolution Dataset

Methods	**Single Resolution**		**Multiresolution**	
	Mean Rank	**Rank**	**Mean Rank**	**Rank**
LBP	14	12	5.5	4
AutoCor	13	11	5.5	4
Interchannel	12	10	4	3
HOG-9	8.75	7	NA	NA
HOG-18	11	9	NA	NA
Ex-HOG	10	8	NA	NA
SI-HOG	5	4	3	2
HOG-9+LBP	6.5	5	NA	NA
HOG-9+AutoCor 3.5	3.5	3	NA	NA
HOG-9+Interchannel	8.25	6	NA	NA
HOG-9+LBP+AutoCor	2.5	2	NA	NA
HOG-9+LBP+Interchannel	6.5	5	NA	NA
SI-HOG+LBP+AutoCor	**1.5**	**1**	**1.5**	**1**
SI-HOG+LBP+Interchannel	**2.5**	**2**	**1.5**	**1**

Friedman Test Analysis including the fusion strategies:

The Friedman test is conducted on the two fusion strategies namely weighted and k-fold SI-HOG+LBP+AutoCor and SI-HOG+LBP+Interchannel. In the case of single resolution INRIA and NICTA datasets, there are 18 algorithms whose Mean Rank yielded is [18, 17, 16, 12.75, 15, 14, 9, 10.5, 7, 12.25, 5.5, 10.5, 3, 5.5, 2.5, 7, 2.5, 3] as per the sequence in Table 3.1. It can be observed that except for the weighted SI-HOG+LBP+Interchannel all the 5 proposed methods are with high mean rank. The calculated chi-square is 32.88. The critical value of chi-square at a degree of freedom (k-1) 17 is 27.59; where k is the number of algorithms. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. In the case of multiresolution INRIA and NICTA datasets, there are 10 algorithms whose Mean Rank yielded is [9.5, 9.5, 8, 6, 3.5, 3.5, 3, 4, 3.5, 4.5] as per the sequence in Table 3.1. In this case, the 6 proposed methods are with top ranks as well. The calculated chi-square is 13.09. The critical value of chi-square at a degree of freedom 9 is 16.919. As the calculated chi-square is less than the critical chi-square, the Null Hypothesis is failed to be rejected. On application of F-distribution on the multiresolution datasets, the calculated F-distribution is 2.665. The critical value of F-distribution with degree of freedom

(9,9) and α = 0.05 is 3.17. There is rejection of Null Hypothesis here as well. It may be concluded that this is due to, among the 10 methods considered, the 6 fusions of SI-HOG are yielding similar mean rank.

3.4 Observations

By considering the proposed scale-space pyramid-based shape feature-extraction method i.e., SI-HOG on the three datasets the following improvements are observed. The performance is evaluated using three datasets, i.e., INRIA, NICTA, and Daimler, considering both single-resolution and multiresolution images.

- The SI-HOG overcomes the shortcoming of HOG, i.e., that it is not applicable to multiresolution images, by considering gradient information from different scales of an image, making it resolution-independent.
- Furthermore, the addition of texture and color information to SI-HOG for extracting a more detailed form of features is proposed.
- SI-HOG outperformed the existing LBP, AutoCor, Interchannel, HOG-9 bins, HOG-18 bins and ExHOG methods in both the single-resolution and multiresolution cases for all three datasets. When texture and color features were added, for INRIA and NICTA, SI-HOG+LBP+Interchannel and SI-HOG+LBP+AutoCor exhibited the best performance in both the single-resolution and multiresolution cases.
- For Daimler, the results did not vary significantly with the addition of texture features (SI-HOG+LBP).

Hence, SI-HOG is the best-performing shape feature among all the individual feature-extraction methods tested. The proposed method was also tested with two fusion methods: weighted and k-fold and the results are compared.

Chapter 4

A Multi-layer Feature Fused-Resnet Model for Pedestrian Detection

The hand-crafted features given in Chapter-3 lacked in the detection accuracy. To bridge the performance gap deep learning methods are used. The deep learning methods, especially CNNs, through the backpropagation error correction produces a better feature representation. Thus, improving the accuracy of the overall detection system. In this chapter, modification to a pre-trained CNN: ResNet18 is proposed to improve the feature extraction process of the network. The proposed network is named Multi-layer Feature Fused-ResNet (MF2-ResNet). Two ways to approach the pedestrian detection problem by using the MF2-ResNet is shown i.e., MF2-ResNet feature extraction with classifier and End-to-end MF2-ResNet.

4.1 Introduction

4.1.1 Base Convolutional Neural Network Features

The process of pedestrian detection is the same as any object detection i.e., feature extraction and then classification. The framework is given in Figure 4.1. Processing an image, feature extraction is done by hand-crafted features or by CNNs. The features produced in the previous step can be classified as either a pedestrian (positive) or a non-pedestrian (negative). Classification is achieved in two ways i.e., SVM and Fully Connected Neural Network. However, a significance performance gap exists among the hand-crafted and CNN features, which is addressed in the next contribution by using deep CNN methods. A Modification to a pre-trained CNN i.e., ResNet18 named Multi-layer Feature Fused-ResNet (MF2-ResNet) is proposed. The proposed MF2-ResNet is trained via transfer learning on the pedestrian dataset. The trained MF2-ResNet Features are used in two ways: 1) with SVM classifier 2) End-to-End network.

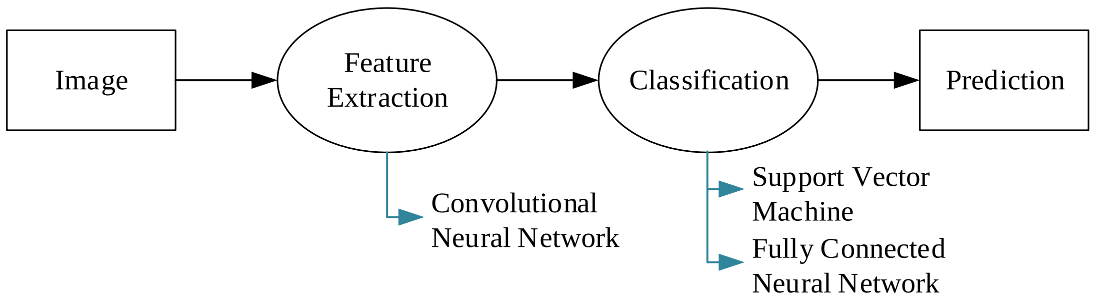


Figure 4.1 The framework of Pedestrian Detection using 1) CNN features with SVM Classifier and 2) End-to-End CNN

4.2 Methodology

The methodology is described by explaining the architecture of the proposed CNN MF2-ResNet in the Subsection 4.2.1. The training algorithm for the same is given in Subsection 4.2.2. The MF2-ResNet is used in two manners: Subsection 4.2.3 shows it with SVM and Subsection 4.2.4 shows it End-to-End.

4.2.1 Proposed Multi-layer Feature Fused-ResNet

The proposed MF2-ResNet is explained in this section by breaking the network configuration into three parts i.e., MF2-ResNet-1, MF2-ResNet-2 and MF2-ResNet-3.

MF2-ResNet-1

The ResNet18 architecture can be divided into four blocks considering the output feature maps. ResNet18 has three reduce shortcuts in blocks 2, 3 and 4. The reduce shortcut layer of block 2 takes input from the output feature map of block 1. The reduce shortcut layer of block 3 takes input from the output feature map of block 2 and so on. The proposed modification adds the reduce shortcut of (n-1)th block to the output feature map of the (n-1)th block and inputs it to the nth block reduce shortcut layer. This enables the successive reduce shortcut layers to apply convolution operations on previous block features as well.

MF2-ResNet-2

The input layer size of ResNet18 is 224×224×3. The output feature map's dimension of these four blocks is 56×56×64, 28×28×128, 14×14×256 and 7×7×512. The fully connected network in ResNet18 inputs only the last feature map i.e., 7×7×512. However, the output feature maps of other blocks can be used here to obtain the features from all the segments of the network to capture the features at varying levels. Here, a depth concatenation is used to concatenate the feature maps from blocks 2, 3 and 4. Max-pool operation is applied on block 2's feature map, to match its size to that of the feature map of block 3. A transpose convolution is applied to block 4's feature map to match its size to that of the feature map of block 3. The said operations were applied to block 2 and block 4 to avoid the loss of feature information that can be caused by a greater degree of upsampling or downsampling. The resultant feature map is of dimension 14×14×(128+256+512) i.e., 14×14×896.

MF2-ResNet-3

An inception module is added to the end or the highest-level feature map of the ResNet18 architecture. The higher-level feature maps are built on the features of lower-level feature maps. Therefore, the last feature map is a comprehensive feature representation of the image. With the further application of an inception module, a feature map is generated that is extracted by applying different receptive fields to the highest feature map. As the features which can be captured by a smaller receptive field can be missed by a filter of a larger receptive field and vice-versa. The inception module considered here applies filters of various receptive field sizes i.e., 1×1, 3×3 and 5×5 on the feature map of block 4, which enables the extraction of richer feature representation.

In the MF2-ResNet architecture, the inception module is applied on the output of Depth Concatenation I of MF2-ResNet-2. This makes the output feature map of MF2-ResNet to be 14×14×448. The representation of the MF2-ResNet architecture is shown in Figure 4.2.

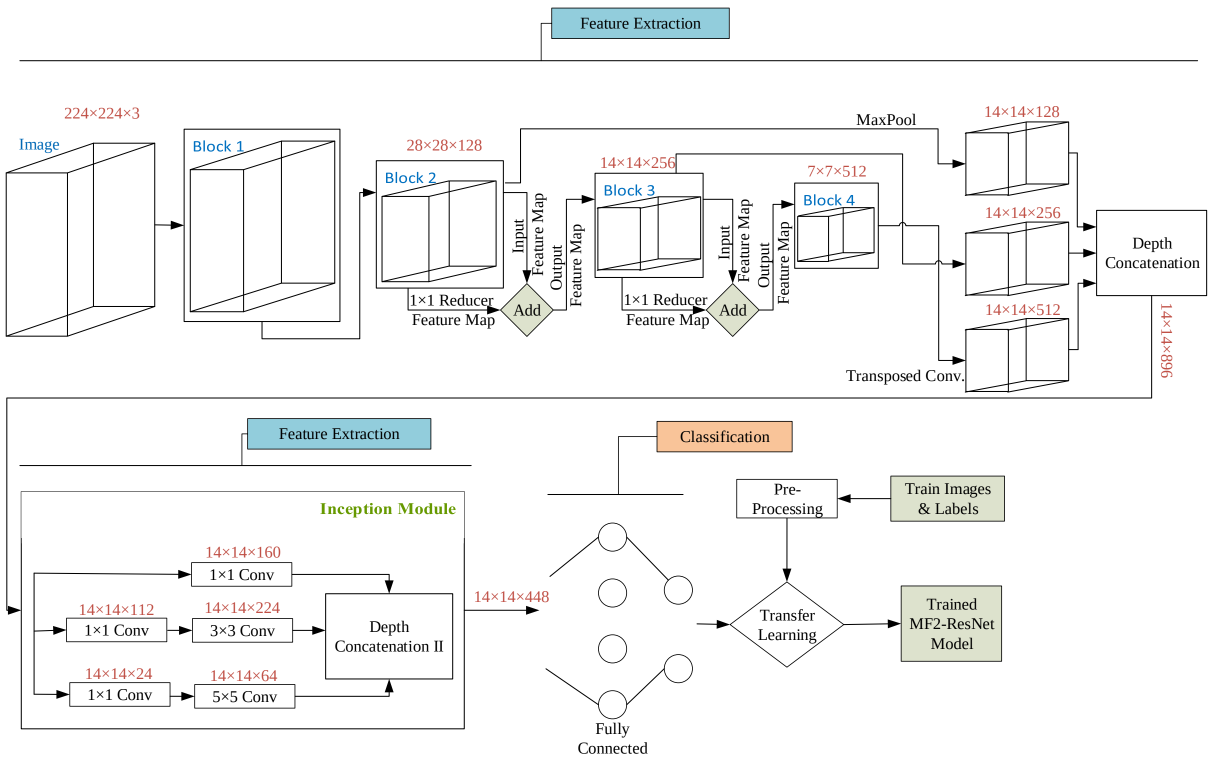


Figure 4.2 Block Diagram for Multi-layer Feature Fused-ResNet (MF2-ResNet)

4.2.2. MF2-ResNet Trained Model Algorithm

The training for MF2-ResNet can be performed in two steps: Pre-processing and Transfer Learning as described in this section.

Pre-processing

The images present in the dataset are of varying sizes. It is strongly required that all images should be of the same size. As each pre-trained CNN has a different size for its' input layer, the images in the dataset will be resized to MF2-ResNet's input layer size i.e., 224×224×3.

Transfer Learning

In a pre-trained CNNs, the network is trained on an enormous amount of data. The initial layers capture simple and basic features that can be useful for any desired task at hand. Even though, it should be re-trained according to the current dataset requirement. The advantage here is, training need not be started from scratch. In transfer learning, a few epochs of training on the target dataset with a small learning step are usually performed to adapt the convolutional network to the new dataset.

4.2.3 Multi-layer Feature Fused-Resnet Features + SVM

This method of classifying MF2-ResNet Features with SVM is performed in three steps: Pre-processing, Feature Extraction and Classification as explained in this section. The overall process for Multi-layer Feature Fused-ResNet Feature + SVM is shown in Figure 4.3.

Pre-processing

The images in the train and test set of the dataset are resized to that of the input layer's size of the proposed MF2-ResNet.

Feature Extraction

The pedestrian dataset has two class labels, i.e., pedestrian and non-pedestrian. The base ResNet18 used in the proposed MF2-ResNet used here was pre-trained on the ImageNet dataset thus had 1000 classes. After training the model, the trained MF2-ResNet's classification layer's label is two (pedestrian/non-pedestrian). The image given to the input layer is processed by each of the layers and feature descriptions known as feature maps are generated at each step. Feature maps extracted at a lower layer gives fundamental characteristics whereas, the higher-level layer builds on the feature maps of several lower layer which yields a superior feature representation. Therefore, we have extracted features from a higher-level layer i.e., the global average pooling layer. The feature vector size of MF2-ResNet is 1×448.

Classification

The features extracted from a dataset are separated based on train and test sets. The train and test sets are further separated into pedestrian and non-pedestrian. These features are given labels accordingly. SVM is used as a classifier because it is robust in the presence of bias in the training sample. The kernel function is selected based on datasets. The trained SVM model is used on the test features to obtain labels (pedestrian/non-pedestrian).

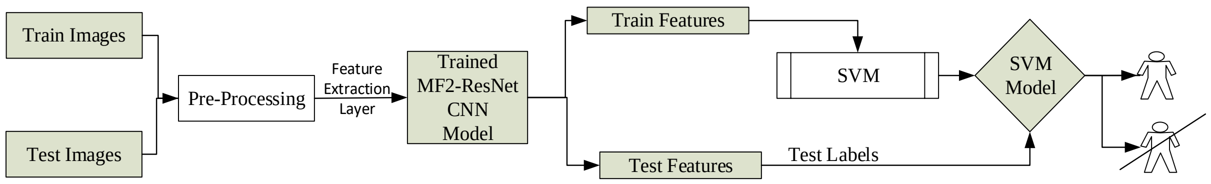


Figure 4.3 Block Diagram for Multi-layer Feature Fused-Resnet Features with SVM

4.2.4 End-to-End Multi-layer Feature Fused-Resnet

This method of End-to-End MF2-ResNet is performed in two steps: Pre-processing and Convolution and Fully connected Pipeline Network as explained in this section. The overall process for End-to-End Multi-layer Feature Fused-Resnet is shown in Figure 4.4.

Pre-processing

Different pre-trained CNNs have different input layer dimensions. As in the previous method, the input images are resized to MF2-ResNet's input layer size before putting it through the CNN pipeline.

Convolution and Fully connected Pipeline Network

Consequently, the fully connected layer has two classes after transfer learning, i.e., pedestrian (positive) and non-pedestrian (negative). The images in this method, are passed through convolutional layers and pooling layers. The feature map derived from the first stage of the trained model is moved to classification, the second stage of the trained model. The fully connected (FC) layers take the output of the preceding layer and match the features to comprehend as to which class it belongs. FC layer outputs the probability of the image belonging to each class. The image is assigned to the class label; having the highest FC layer probability.

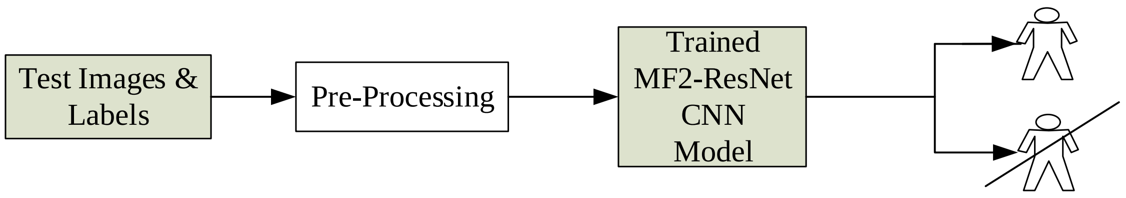


Figure 4.4 Block Diagram for End-to-End Multi-layer Feature Fused-Resnet Network

4.3 Experimental Results and Discussion

The proposed MF2-ResNet is evaluated individually in parts and also in a group of two on the INRIA, NICTA and Daimler Pedestrian datasets against four existing methods. The details of the datasets are given in Chapter-1. The methods are compared with existing pre-trained CNNs, i.e., AlexNet, ResNet18, Xception, and DarkNet. The training parameters such as batch size are set as 10, the learning rate is 10^{-4} and the number of epochs is set to 6. The optimizer used is *sgdm* for training. Subsection 4.3.1, 4.3.2 and 4.3.3 discusses the performance for the INRIA, NICTA and Daimler pedestrian datasets respectively. Subsection 4.3.4 shows the statistical analysis of the results.

4.3.1 INRIA Pedestrian Dataset

Results by using Multi-layer Feature Fused-Resnet Features + SVM

The proposed method is applied on the INRIA Pedestrian dataset to get the predicted labels from the SVM classifier. Upon comparing the predicted and groundtruth labels, the DET Curve is obtained. The DET Curve for Multi-layer Feature Fused-Resnet Features + SVM is shown in Figure 4.5. The corresponding miss rate values are given in Table 4.1. The proposed MF2-ResNet-2 and MF2-ResNet-3 individually are giving zero miss rate which signifies there is negligible or no misclassification. The proposed method MF2-ResNet and the proposed method group MF2-ResNet-1+2 is showing zero miss rate as well. It should be noted here that the proposed method MF2-ResNet achieved a hundred percent classification. The proposed method group MF2-ResNet-2+3 is also giving less miss rate from the existing pre-trained CNNs. In this dataset, as there is a fewer number of misclassifications, in some methods there isn't any value in the DET curve. As the DET curve plots error rates on both axes, the values are approximately zero; which resulted in a zero miss rate.

Results by using End-to-End Multi-layer Feature Fused-Resnet Network

The proposed method is applied on the INRIA Pedestrian dataset to get the predicted labels from the MF2-ResNet softmax layer. The predicted and groundtruth labels are compared to yield the DET Curve. The DET Curve for End-to-End Multi-layer Feature Fused-Resnet Network is shown in Figure 4.6 and the corresponding miss rate is given in Table 4.1. The zero miss rate is shown by individual proposed method MF2-ResNet-2 and MF2-ResNet-3, the proposed method MF2-ResNet and the group proposed method MF2-ResNet-1+2. It should be noted here that the group proposed method MF2-ResNet-1+2 achieved a hundred percent classification.

4.3.2 NICTA Pedestrian Dataset

Results by using Multi-layer Feature Fused-Resnet Features + SVM

The proposed method is applied on the NICTA Pedestrian dataset to predict the test dataset labels from the SVM classifier. Then the resultant predicted labels are compared to the groundtruth labels and the DET Curve is obtained. The DET Curve for Multi-layer Feature Fused-Resnet Features + SVM is shown in Figure 4.7 and the corresponding miss rate is given in Table 4.2. Except for individual proposed MF2-ResNet-1 and MF2-ResNet-3 & group proposed method MF2-ResNet-1+3, all others are performing better than the existing pre-trained CNNs. The proposed method group MF2-ResNet-2+3 is giving the least miss rate among all.

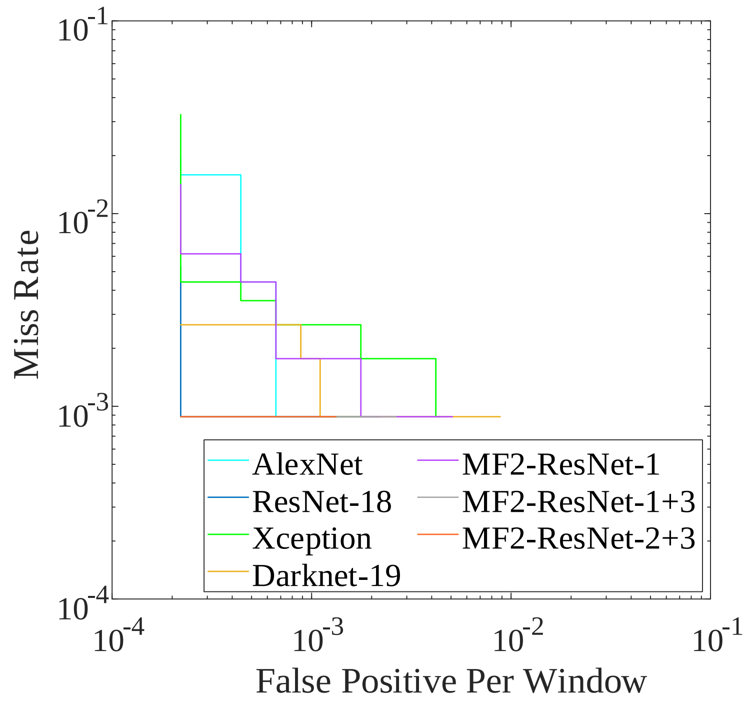


Figure 4.5 DET Curve for INRIA Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM

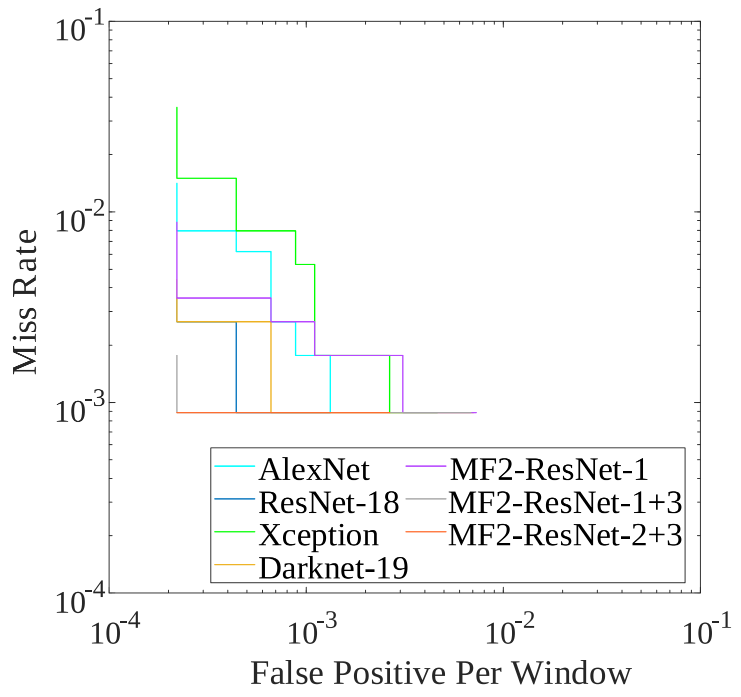


Figure 4.6 DET Curve for INRIA Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network

Table 4.1 Miss Rate for INRIA Pedestrian Dataset for the Proposed Methods

Methods	**Miss Rate (%)**	
	CNN Features+SVM	**End-to-End CNN**
AlexNet	0.00199	0.00281
ResNet-18	0.00088	0.00071
Xception	0.00283	0.00490
Darknet-19	0.00177	0.00133
MF2-ResNet-1	0.00252	0.00250
MF2-ResNet-2	0.00000	0.00000
MF2-ResNet-3	0.00000	0.00000
MF2-ResNet-1+2	0.00000	**0.00000**
MF2-ResNet-1+3	0.00088	0.00088
MF2-ResNet-2+3	0.00071	0.00088
MF2-ResNet-1+2+3	**0.00000**	0.00000

Results by using End-to-End Multi-layer Feature Fused-Resnet Network

The proposed method is applied on the NICTA Pedestrian dataset to predict the test dataset labels from the MF2-ResNet softmax layer. Upon comparing the resultant predicted labels and the groundtruth labels, the DET Curve is obtained. The DET Curve for End-to-End Multi-layer Feature Fused-Resnet Network is shown in Figure 4.8 and the corresponding miss rate is given in Table 4.2. Except for the individual proposed method MF2-ResNet-1 and MF2-ResNet-3 and group proposed method MF2-ResNet-1+3, all others are performing better than the existing pre-trained CNNs. The best performance i.e., least miss rate is achieved by the proposed method MF2-ResNet.

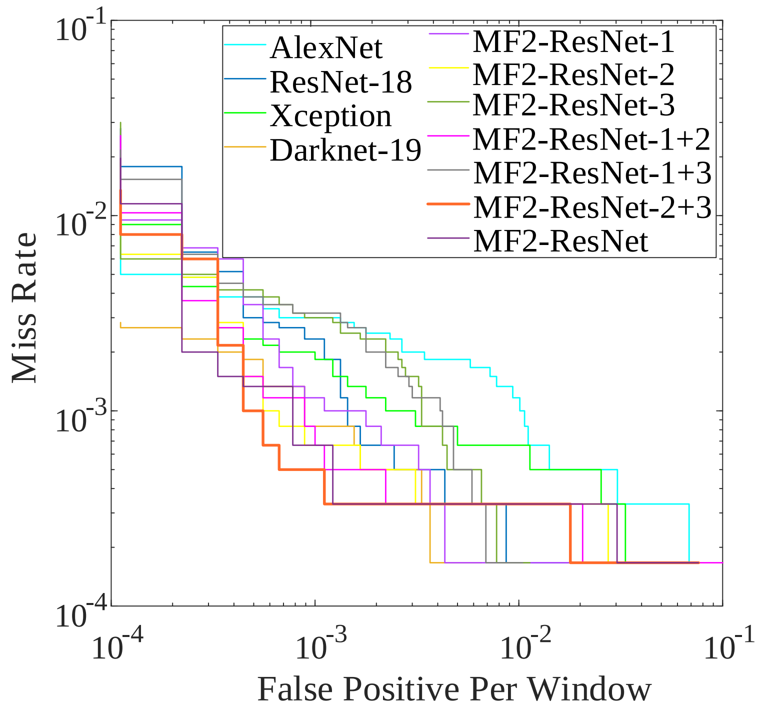


Figure 4.7 DET Curve for NICTA Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM

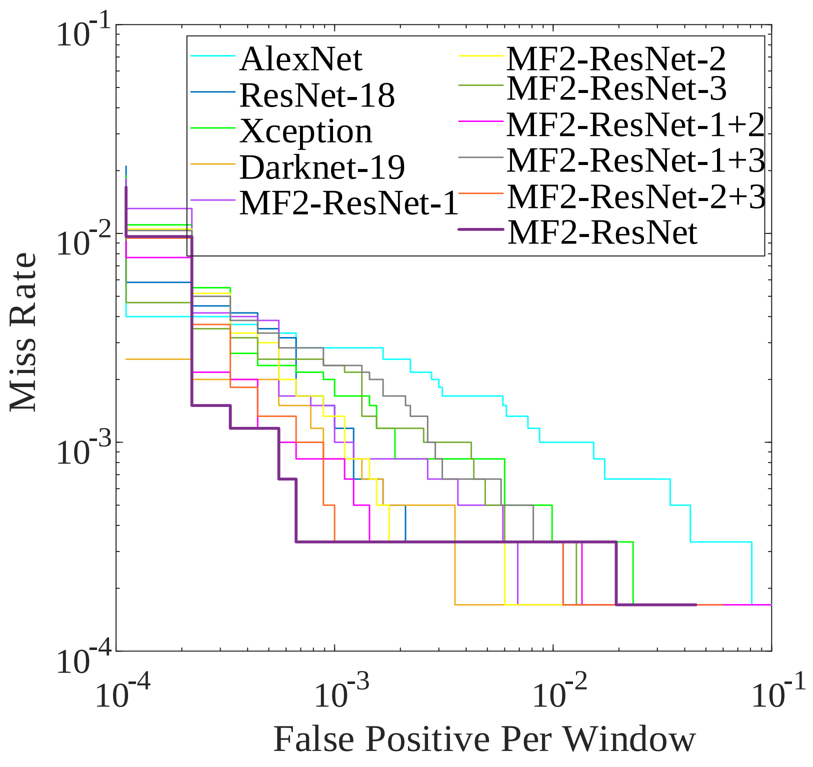


Figure 4.8 DET Curve for NICTA Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network

Table 4.2 Miss Rate for NICTA Pedestrian Dataset for the Proposed Methods

Methods	**Miss Rate (%)**	
	CNN Features+SVM	**End-to-End CNN**
AlexNet	0.00300	0.00283
ResNet-18	0.00233	0.00133
Xception	0.00192	0.00183
Darknet-19	0.00083	0.00083
MF2-ResNet-1	0.00117	0.00125
MF2-ResNet-2	0.00067	0.00083
MF2-ResNet-3	0.00300	0.00233
MF2-ResNet-1+2	0.00075	0.00083
MF2-ResNet-1+3	0.00317	0.00233
MF2-ResNet-2+3	**0.00050**	0.00042
MF2-ResNet-1+2+3	0.00067	**0.00033**

4.3.3 Daimler Pedestrian Dataset

Results by using Multi-layer Feature Fused-Resnet Features + SVM

The proposed method is applied on the Daimler Pedestrian dataset and the test dataset labels are predicted from the SVM classifier. The resultant predicted labels are compared to the groundtruth labels and the DET Curve is obtained. The DET Curve for Multi-layer Feature Fused-Resnet Features + SVM is shown in Figure 4.9 and the corresponding miss rate is given in Table 4.3. The least miss rate is given by the proposed method group MF2-ResNet-2+3.

Except for individual proposed MF2-ResNet-1 and MF2-ResNet-2 & the group proposed method MF2-ResNet-1+3, all others are performing better than the existing pre-trained CNNs.

Results by using End-to-End Multi-layer Feature Fused-Resnet Network

The proposed method is applied on the Daimler Pedestrian dataset and the test dataset labels are predicted from the MF2-ResNet softmax layer. The resultant predicted labels are compared to the groundtruth labels and the DET Curve is yielded. The DET Curve for End-to-End Multi-layer Feature Fused-Resnet Network is shown in Figure 4.10 and the corresponding miss rate is given in Table 4.3. The best performance is obtained by the group proposed method MF2-ResNet-2+3.

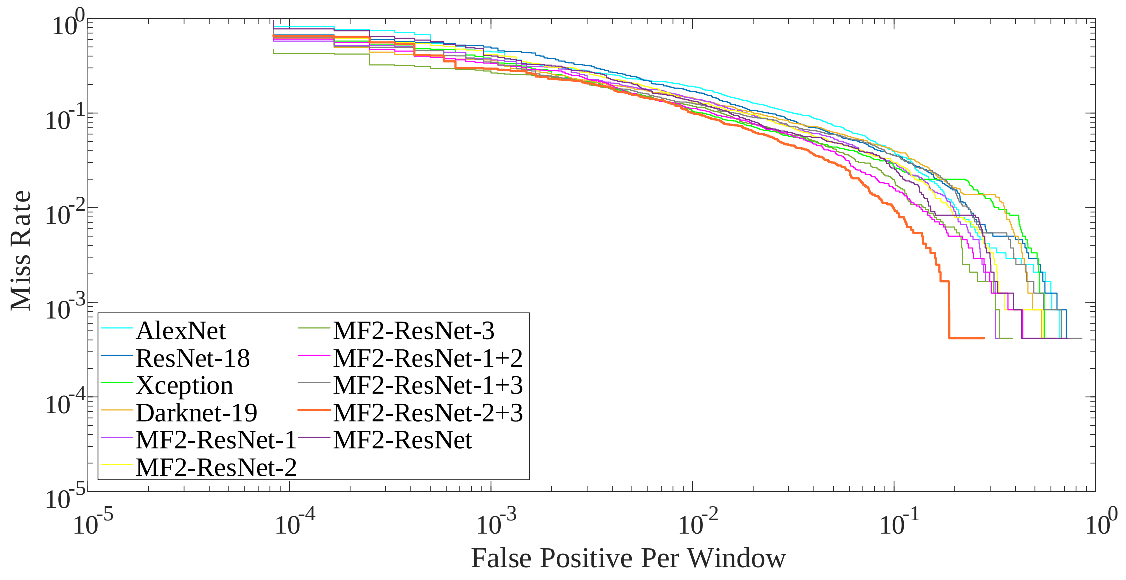


Figure 4.9 DET Curve for Daimler Pedestrian Dataset in Multi-layer Feature Fused-Resnet Features + SVM

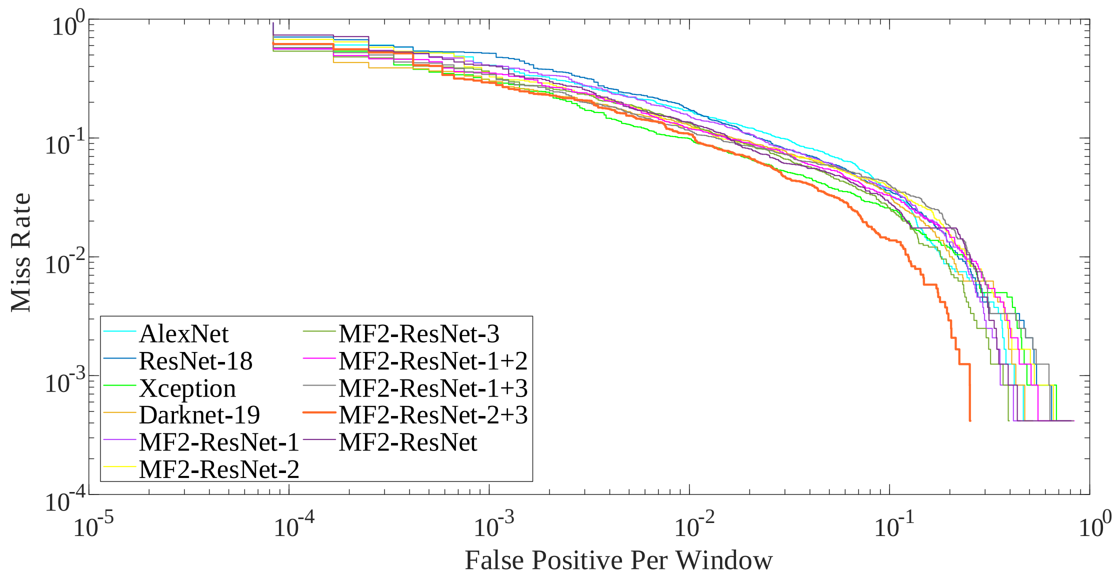


Figure 4.10 DET Curve for Daimler Pedestrian Dataset in End-to-End Multi-layer Feature Fused-Resnet Network

Table 4.3 Miss Rate for Daimler Pedestrian Dataset for the Proposed Methods

Methods	**Miss Rate (%)**	
	CNN Features+SVM	**End-to-End CNN**
AlexNet	0.03917	0.03458
ResNet-18	0.03583	0.03625
Xception	0.02750	0.02583
Darknet-19	0.04000	0.03354
MF2-ResNet-1	0.02958	0.03750
MF2-ResNet-2	0.03000	0.03833
MF2-ResNet-3	0.02000	0.02542
MF2-ResNet-1+2	0.01583	0.03292
MF2-ResNet-1+3	0.03583	0.04042
MF2-ResNet-2+3	**0.00958**	**0.01375**
MF2-ResNet-1+2+3	0.02625	0.02833

4.3.4 Statistical Analysis

In this work, there are 11 independent variables (k) and 3 datasets (N). The proposed method and its groups have the highest rank when compared among the other CNN methods. In case of MF2-ResNet Features + SVM, the calculated chi-square is 24.151. The critical value of chi-square at a degree of freedom (k-1) 10 is 18.310. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. The rank table for Friedman Test is given in Table 4.4. For, the F-distribution test, the calculated value is 8.258. The critical value of F-distribution with degree of freedom (10,20) [*k-1, (k-1)(N-1)*] and α = 0.05 is 2.347. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected. In case of End-to-End MF2-ResNet, the calculated chi-square is 14.545. And for, the F-distribution test, the calculated value is 1.882. As the calculated value of both chi-square and F-distribution is lesser than the respective critical value, the Null Hypothesis is failed to rejected. The rank table for Friedman Test is given in Table 4.5.

Table 4.4 Rank Table of Friedman Test for the Proposed Methods in Multi-layer Feature Fused-Resnet Features + SVM

CNN Features+SVM	r_{INRIA}	r_{NICTA}	$r_{Daimler}$	**Mean Rank**	**Rank**
AlexNet	9	9	10	9.333	11
ResNet-18	6.5	8	9	7.833	7
Xception	11	7	8.5	8.833	9
Darknet-19	8	5	11	8.000	8
MF2-ResNet-1	10	6	6	7.333	6
MF2-ResNet-2	2.5	2.5	7	4.000	4
MF2-ResNet-3	2.5	10	3	5.167	5
MF2-ResNet-1+2	2.5	4	2	2.833	2
MF2-ResNet-1+3	6.5	11	8.5	8.667	10
MF2-ResNet-2+3	5	1	1	2.333	1
MF2-ResNet-1+2+3	2.5	2.5	4	3.000	3

Table 4.5 Rank Table of Friedman Test for the Proposed Methods in End-to-End Multi-layer Feature Fused-Resnet Network

End-to-End CNN	r_{INRIA}	r_{NICTA}	$r_{Daimler}$	**Mean Rank**	**Rank**
AlexNet	10	11	7	9.333	11
ResNet-18	5	7	8	6.667	7
Xception	11	8	3	7.333	9
Darknet-19	8	4	6	6.000	6
MF2-ResNet-1	9	6	9	8.000	10
MF2-ResNet-2	2.5	4	10	5.500	5
MF2-ResNet-3	2.5	9.5	2	4.667	4
MF2-ResNet-1+2	2.5	4	5	3.833	3
MF2-ResNet-1+3	6.5	9.5	11	9.000	8
MF2-ResNet-2+3	6.5	2	1	3.167	2
MF2-ResNet-1+2+3	2.5	1	4	2.500	1

4.4 Observations

In this chapter, by applying the proposed CNN feature extraction method i.e., MF2-ResNet for pedestrian detection the following improvements are observed. The proposed methods are compared using three benchmark pedestrian datasets INRIA, NICTA and Daimler. The basic principle involved here is feature extraction and then classification. Two methods are given here: MF2-ResNet features with SVM and End-to-End MF2-ResNet network.

- In case of MF2-ResNet features with SVM, the miss rate observed is least when compared with the existing methods in all the three datasets.
- In case of End-to-End MF2-ResNet Network, in all the three datasets least miss rate is resulted.
- In the case of the INRIA Pedestrian dataset, a hundred percent classification was also obtained.
- It should be noted that even though the classification method is changed, the proposed method is always resulting in less misclassification for all the three datasets.

Chapter 5

Faster R-CNN based on Dilate-Condense ResNet and Multi-layer Feature Fused-Resnet Model for Pedestrian Detection

The pedestrian detection requires to localize the pedestrians in a given image. The CNNs used in previous contribution didn't generate location information. Therefore, a detection network is employed here to get the region proposals which assists in yielding bounding box information from an image. In this chapter, the two-stage detection network Faster R-CNN is considered because of its accuracy. Focusing on the feature extraction part of the network, modifications of the pre-trained CNN ResNet is proposed as the base network for the new and improved Faster R-CNN. The modifications are done by taking the pre-trained ResNet18. The proposed Faster R-CNN modifications named Faster R-CNN(Dilate-CondenseResNet) and Faster R-CNN(Multi-layer Feature Fused-Resnet) yields a detailed representation of the image as varying levels of the network are analyzed for feature extraction in the proposed methods.

5.1 Introduction

5.1.1 ResNet Feature Extraction Layer with Faster R-CNN

Till now, the pedestrian detection is being done in per-window evaluation system. To further address the pedestrian detection problem, the state-of-the-art detection method is employed in the next contribution. The Faster R-CNN detection paradigm is made up of first, a region proposal algorithm, employed to output the probable regions of the desired objects. Second, feature extraction by the pre-trained CNNs. Third, classification layer to predict the object's class. Lastly, regression stage to fine-tune the bounding boxes generated by the region proposal method.

Faster R-CNN speeds up the region proposal step by including a convolutional network for region proposal termed as the Region Proposal Network (RPN). It uses a pre-trained CNN as a backbone network which enables it to share weights with the CNN used in the detection process. In this work, ResNet18 is used as the base network. It has 18 conv layers of 3×3 receptive field. The layers for Faster R-CNN with base ResNet18 are shown in Table 5.1.

Table 5.1 Faster R-CNN base ResNet18 layers

	Layer Name	Filter Size/Strid	Channels	Input Layer	Output Size
	Input				224×224×3
	Conv1	7×7/2	64	Input	112×112×64
	Maxpool	3×3/2		Conv1	56×56×64
	Conv2_1	3×3/1	64	Maxpool	56×56×64
	Conv2_2	3×3/1	64	Conv2_1	56×56×64
×2	Add_1			Conv2_2	
				Maxpool /	
				{Add_1}	56×56×64
	Conv3_1	3×3/2/{1}	128	Add_1	28×28×128
	Conv3_2	3×3/1	128	Conv3_1	28×28×128
×2	Add_2			Conv3_2	
				Add_1 /	
				{Add_2}	28×28×128
	Conv4_1	3×3/2/{1}	256	Add_2	14×14×256
	Conv4_2	3×3/1	256	Conv4_2	14×14×256
×2	Add_3			Conv4_2	
				Add_2 /	
				{Add_3}	14×14×256
	RegPropNet			Add_3	RegProps
	ROIPooling			Add_3	
				RegProps	14×14×256
	Conv5_1	3×3/2/{1}	512	ROIPooling	14×14×256
×2	Conv5_2	3×3/1	256	Conv5_1	14×14×256
	Add_4			Conv5_2	
				ROIPooling	14×14×256

5.2 Methodology

In this section, the proposed network's architecture is discussed. The Subsection 5.2.1 explains the Dilate-Condense ResNet based Faster R-CNN [Faster R-CNN(DCResNet)] architecture details and the Subsection 5.2.2 explains the Multi-layer Feature Fused-Resnet based Faster R-CNN [Faster R-CNN(MF2-ResNet)] architecture details.

5.2.1 Dilate-Condense ResNet based Faster R-CNN

The proposed Faster R-CNN(DCResNet) is described in three components: Selection of Feature Map Blocks, Dilate and Condense 3×3 Network and Fusion of the Output Feature Map Blocks, in this section.

- *Selection of Feature Map Blocks:* The pre-trained CNN ResNet18 can be divided into four subparts depending on the distinct feature map size. These blocks are of size

56×56×64, 28×28×128, 14×14×256, and 7×7×512. The first three feature maps are processed in the next step. The division of feature maps is shown in Figure 5.2.

- *Dilate and Condense 3×3 Network:* Three dilate and condense 3×3 networks are attached in the network at feature map block I, II, and III. The characteristic of these blocks is to expand the input by a 1×1 conv. The expanded result is convolved with a receptive field of 3×3. This enables to further extract detailed features from the input. Lastly, it is reduced with another 1×1 conv. Hence, the output of these networks is feature maps which not only carry additional information but also have reduced dimension to enable faster computation. The three output feature map block derived in this step are of size 56×56×43, 28×28×85, and 14×14×128. The Dilate and Condense 3×3 Network I, II, and III blocks are shown in Figure 5.3, Figure 5.4, and Figure 5.5 respectively.
- *Fusion of the Output Feature Map Blocks:* The output generated by the dilate and condense 3×3 network in the previous step is fused to form one final feature map. First, as the three inputs to this step are of uneven size, they are made into the same size by applying a downsample and upsample operation on output feature map block I and III respectively. An average pool operation is performed on output feature map block I and a resize operation is performed on the output feature map block III. The fusion is accomplished by a depth concatenation, which concatenates the feature maps channel-wise. Therefore, the input to the depth concatenation becomes feature maps of size 28×28×43, 28×28×85, and 28×28×128. The fused feature map is of size 28×28×256.

The architecture of the proposed Faster R-CNN(DCResNet) is represented by a block diagram in Figure 5.1. The network layers' characteristics are shown in Table 5.2.

5.2.2 Multi-layer Feature Fused-Resnet based Faster R-CNN

The proposed Faster R-CNN(MF2-ResNet) is described in four components: Reduce-shortcut layer, Feature Map Fusion, Inception Module and Region Proposal Network in this section.

- *Reduce-shortcut layer:* The RPN network uses ResNet18 as the backbone. As in this case, it uses res4b_relu i.e., the output feature map of block 3, as a feature extraction layer when MF2-ResNet is used as the backbone CNN. Therefore, this allowed having only one reduced shortcut layer from block 2 to block 3 in the MF2-ResNet.

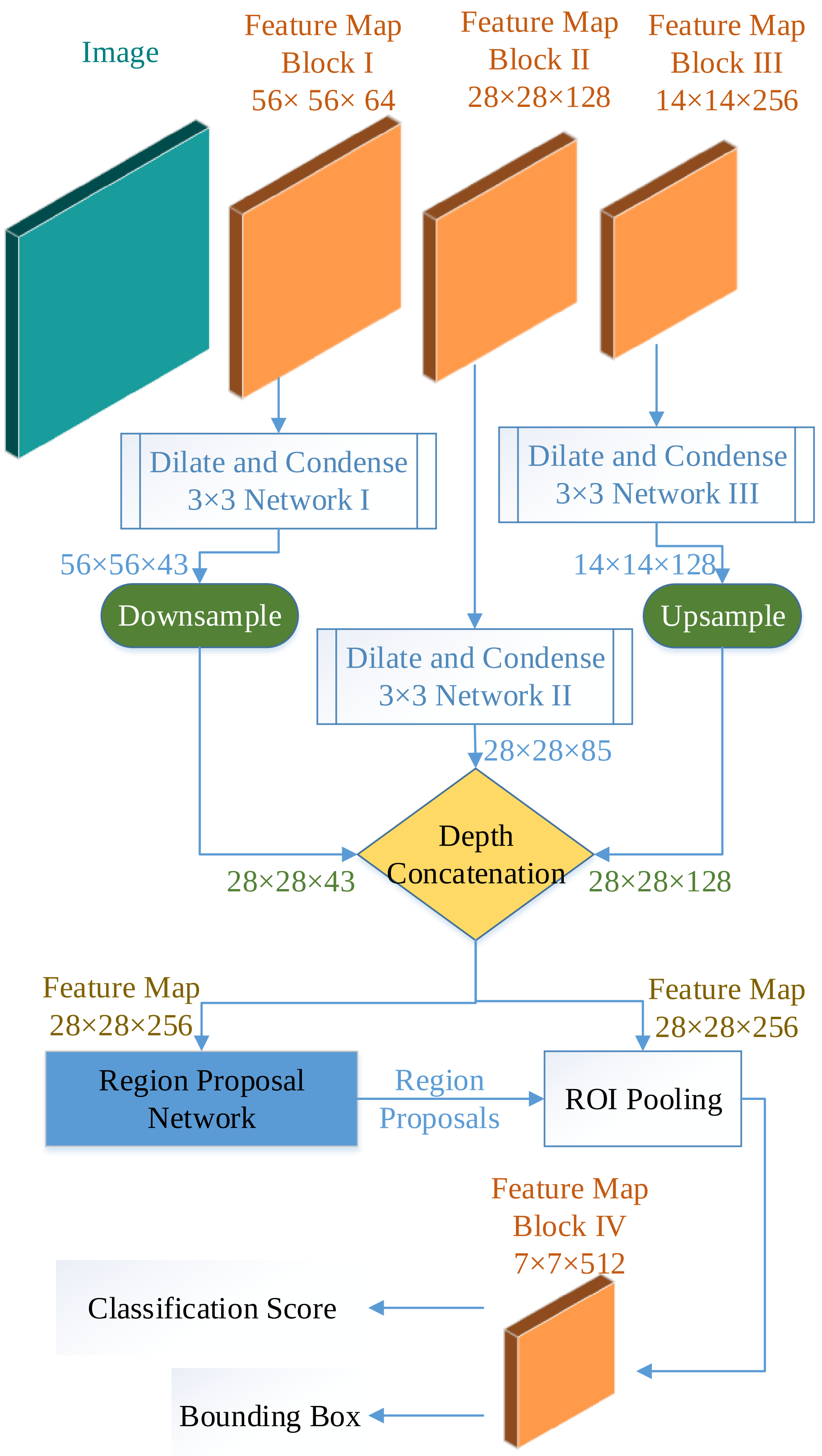


Image
Feature Map Block I 56× 56× 64
Feature Map Block II 28×28×128
Feature Map Block III 14×14×256
Dilate and Condense 3×3 Network I
Dilate and Condense 3×3 Network III
56×56×43
14×14×128
Downsample
Upsample
Dilate and Condense 3×3 Network II
28×28×85
Depth Concatenation
28×28×43
28×28×128
Feature Map 28×28×256
Feature Map 28×28×256
Region Proposal Network
Region Proposals
ROI Pooling
Feature Map Block IV 7×7×512
Classification Score
Bounding Box

Figure 5.1 The Block Diagram of the Proposed Faster R-CNN with the modified DCResNet as the base

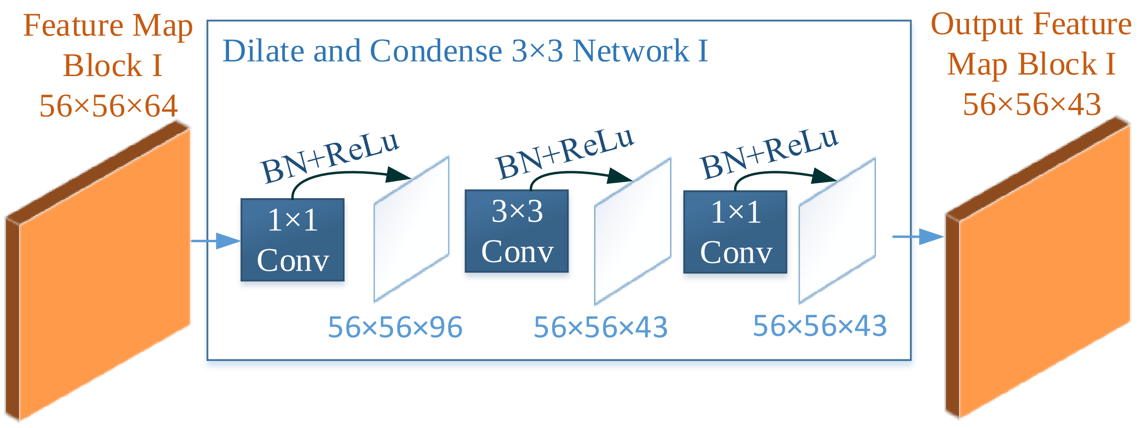


Figure 5.2 The structure of Dilute ad Condense 3×3 Network for I, input for the Feature Map Block I

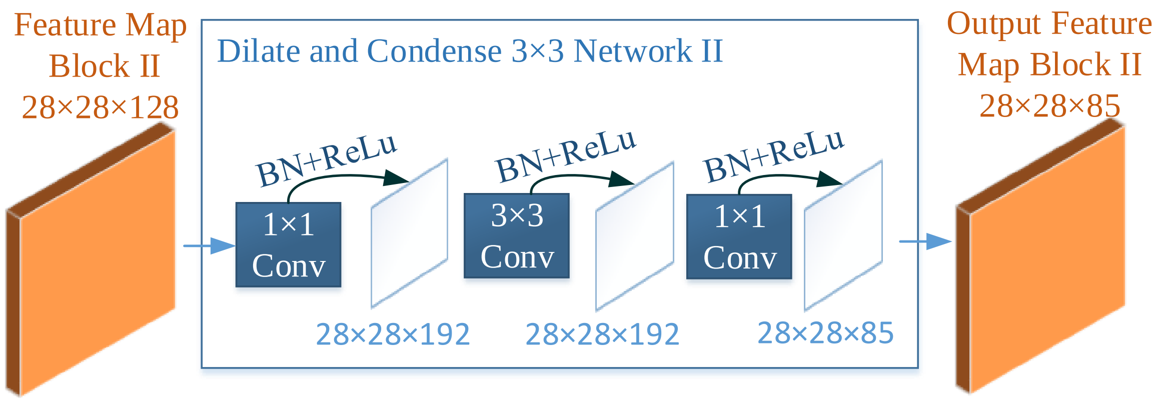


Figure 5.3 The structure of Dilute ad Condense 3×3 Network for II, input for the Feature Map Block II

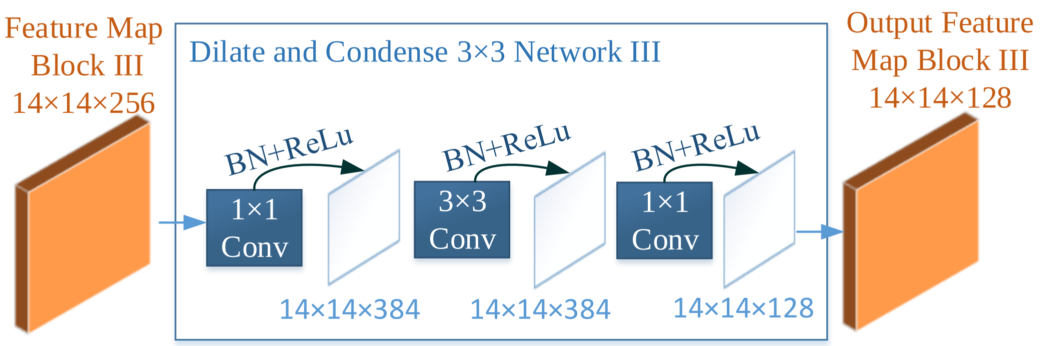


Figure 5.4 The structure of Dilute ad Condense 3×3 Network for III, input for the Feature Map Block III

Table 5.2 The Proposed FASTER R-CNN(DCRESNET) network's layer and characteristics

	Layer Name	**Filter Size/Stride**	**Channels**	**Input Layer**	**Output Size**
	Input				224×224×3
	Conv1	7×7/2	64	Input	112×112×64
	Maxpool	3×3/2		Conv1	56×56×64
	Conv2_1	3×3/1	64	Maxpool	56×56×64
	Conv2_2	3×3/1	64	Conv2_1	56×56×64
×2	Add_1			Conv2_2	
				Maxpool /	
				{Add_1}	56×56×64
	Conv3_1	3×3/2/{1}	128	Add_1	28×28×128
	Conv3_2	3×3/1	128	Conv3_1	28×28×128
×2	Add_2			Conv3_2	
				Add_1 /	
				{Add_2}	28×28×128
	Conv4_1	3×3/2/{1}	256	Add_2	14×14×256
	Conv4_2	3×3/1	256	Conv4_2	14×14×256
×2	Add_3			Conv4_2	
				Add_2 /	
				{Add_3}	14×14×256
	ConvDC1_1	1×1/1	96	Add_1	56×56×96
	ConvDC1_2	3×3/1	96	ConvDC1_1	56×56×96
	ConvDC1_3	1×1/1	43	ConvDC1_2	56×56×43
	AveragePool	3×3/2		ConvDC1_3	28×28×43
	ConvDC2_1	1×1/1	96	Add_2	28×28×192
	ConvDC2_2	3×3/1	96	ConvDC2_1	28×28×192
	ConvDC2_3	1×1/1	43	ConvDC2_2	28×28×85
	ConvDC3_1	1×1/1	384	Add_3	14×14×384
	ConvDC3_2	3×3/1	384	ConvDC3_1	14×14×384
	ConvDC3_3	1×1/1	128	ConvDC3_2	14×14×128
	Resize	2		ConvDC3_3	28×28×128
	DepthConcat			ConvDC1_3	
				ConvDC2_3	
				ConvDC3_3	28×28×256
	RegPropNet			DepthConcat	RegProps
	ROIPooling			DepthConcat	
				RegProps	14×14×256
	Conv5_1	3×3/2/{1}	512	ROIPooling	14×14×256
	Conv5_2	3×3/1	256	Conv5_1	14×14×256
×2	Add_4			Conv5_2	
				ROIPooling	
				/{Add_4}	14×14×256

- *Feature Map Fusion:* The Depth Concatenation I use features from block 1, 2 and 3 for MF2-ResNet. Max-pool operation is applied on block 1's feature map, to match its size to that of the feature map of block 2. A transpose convolution is applied to block 3's feature map to match its size to that of the feature map of block 2.
- *Inception Module:* The inception module applies filters of various receptive field sizes i.e., 1×1, 3×3 and 5×5 to the input feature map from block 3 for MF2-ResNet.
- *Region Proposal Network:* The feature map from the Depth Concatenation II layer is given as input to generate a region where a pedestrian might be present, which is the location in the input image. This is done by using two sibling convolution layers. The task of one layer is to classify and the other's task is to improve the co-ordinates of anchor boxes. *Anchors* of various sizes are placed on the input image corresponding to a point in the feature map received as input. The network checks whether the anchor boxes at any point corresponds to a pedestrian.

The working of the proposed Faster R-CNN(MF2ResNet) is represented by a block diagram in Figure 5.5. The network layers' characteristics are shown in Table 5.3.

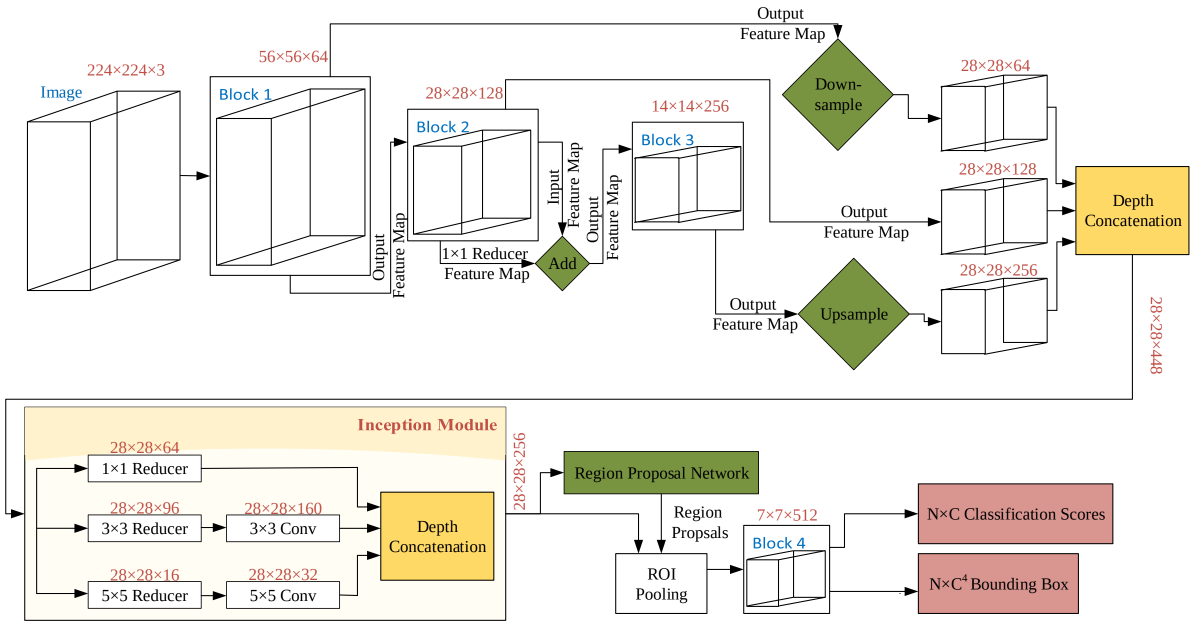


Figure 5.5 The Block Diagram of the Proposed Faster R-CNN with the modified MF2ResNet as the base

Table 5.3 The proposed FASTER R-CNN(MF2RESNET) network's layer and characteristics

	Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
	Input				224×224×3
	Conv1	7×7/2	64	Input	112×112×64
	Maxpool	3×3/2		Conv1	56×56×64
	Conv2_1	3×3/1	64	Maxpool	56×56×64
	Conv2_2	3×3/1	64	Conv2_1	56×56×64
×2	Add_1			Conv2_2	
				Maxpool /	
				{Add_1}	56×56×64
	Conv3_1	3×3/2/{1}	128	Add_1	28×28×128
	Conv3_2	3×3/1	128	Conv3_1	28×28×128
×2	Add_2			Conv3_2	
				Add_1 / {Add_2}	
					28×28×128
	Add_P			Conv3_1	
				Add_2	28×28×128
	Conv4_1	3×3/2/{1}	256	Add_P	14×14×256
	Conv4_2	3×3/1	256	Conv4_2	14×14×256
×2	Add_3			Conv4_2	
				Add_P / {Add_3}	
					14×14×256
	MaxPool	3x3/2		Add_1	28×28×64
	Transpose Conv.	3x3/2		Add_3	28×28×256
	DepthConcat_I			MaxPool	
				Add_2	
				Transpose Conv.	28×28×448
	Reducer1×1	1×1/1	64	DepthConcat_I	28×28×64
	Reducer3×3	1×1/1	96	DepthConcat_I	28×28×96
	Reducer5×5	1×1/1	16	DepthConcat_I	28×28×16
	Conv3×3	3×3/1	160	Reducer3×3	28×28×160
	Conv5×5	5×5/1	32	Reducer5×5	28×28×32
	DepthConcat_II			Reducer1×1	
				Conv3×3	
				Conv5×5	28×28×256
	RegPropNet			DepthConcat_II	RegProps
	ROIPooling			DepthConcat_II	
				RegProps	14×14×256
	Conv5_1	3×3/2/{1}	512	ROIPooling	14×14×256
	Conv5_2	3×3/1	256	Conv5_1	14×14×256
×2	Add_4			Conv5_2	
				ROIPooling	
				/{Add_4}	14×14×256

5.3 Experimental Results and Discussion

The proposed Faster R-CNN modifications are compared with four state-of-the-art detection methods Faster R-CNN, YOLOv2, YOLOv3, and SSD on two pedestrian datasets: INRIA Pedestrian and PASCAL VOC 2012. Four pre-trained CNN AlexNet, ResNet18, SqueezeNet, and MobileNetv2 are used as base for Faster R-CNN and YOLOv2. Training parameters such as batch size are set to 2, the learning rate is 10^{-3} and the number of epochs is 10 with *sgdm* optimizer. Subsection 5.3.1 and 5.3.2 discusses the performance for the INRIA pedestrian and PASCAL VOC 2012 respectively. Subsection 5.3.3 shows the result's statistical analysis.

5.3.1 INRIA Pedestrian Dataset

The proposed FasterRCNN methods generate bounding box with confidence score for the test images of the INRIA Pedestrian dataset. Comparing the generated prediction and the groundtruth bounding box, DET Curve and P-R Curve are obtained. The DET and P-R Curve are shown in Figure 5.6 and Figure 5.7 respectively. The LAMR and AP values are given in Table 5.4. The proposed Faster R-CNN(DCResNet) method has achieved a minimum LAMR and AP improvement of 8.18% and 5.81% respectively when compared to the Faster R-CNN methods. And it has achieved a minimum LAMR and AP improvement of 2.48% and 2.97% respectively when compared to the YOLOv2 and YOLOv3 methods. The proposed Faster R-CNN(MF2ResNet) method has achieved a minimum LAMR and AP improvement of 16.58% and 11.41% respectively when compared to the Faster R-CNN methods. And it has achieved a minimum LAMR and AP improvement of 10.88% and 8.57% respectively when compared to the YOLOv2 and YOLOv3 methods. Some sample output images for Faster R-CNN(DCResNet) and Faster RCNN(MF2ResNet) are shown in Figure 5.8 and Figure 5.9 respectively.

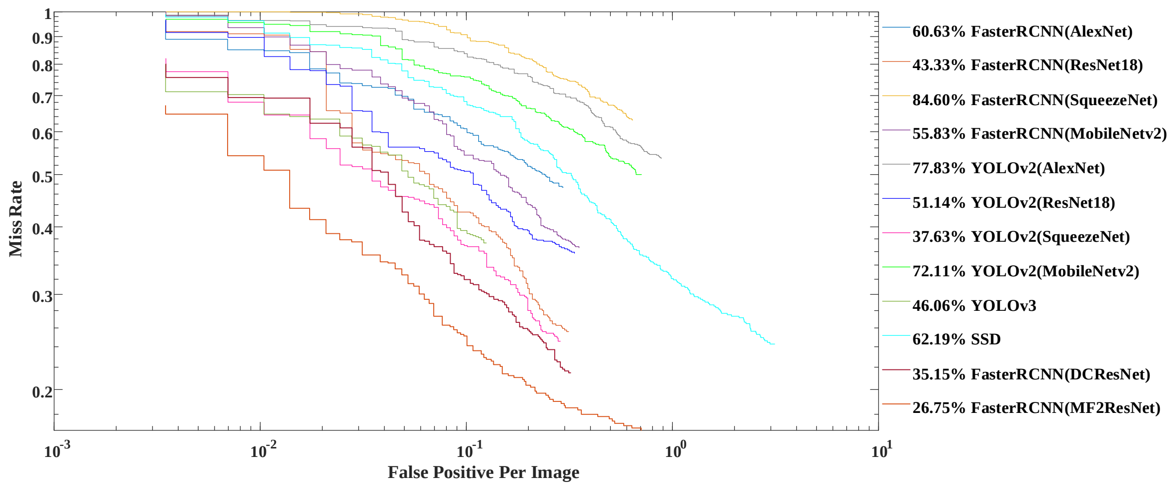


Figure 5.6 DET Curve for INRIA Pedestrian Dataset

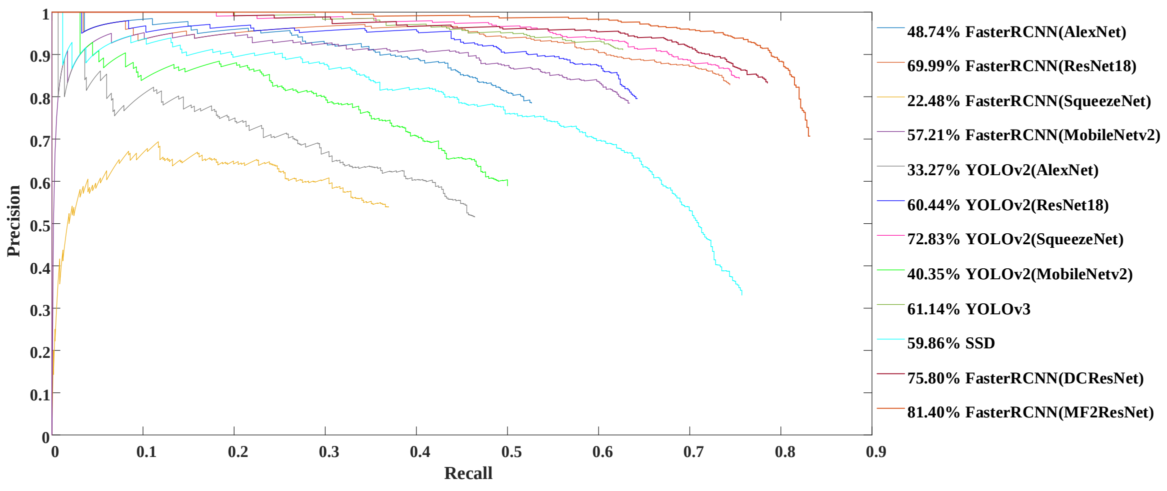


Figure 5.7 P-R Curve for INRIA Pedestrian Dataset

Table 5.4 LAMR and AP values for INRIA Pedestrian Dataset

Methods	**INRIA**	
	LAMR	**AP**
FasterRCNN(AlexNet)	60.63	48.74
FasterRCNN(ResNet18)	43.33	69.99
FasterRCNN(SqueezeNet)	84.60	22.48
FasterRCNN(MobileNetv2)	55.83	57.21
YOLOv2(AlexNet)	77.83	33.27
YOLOv2(ResNet18)	51.14	60.44
YOLOv2(SqueezeNet)	37.63	72.83
YOLOv2(MobileNetv2)	72.11	40.35
YOLOv3	46.06	61.14
SSD	62.19	59.86
FasterRCNN(DCResNet)	**35.15**	**75.80**
FasterRCNN(MF2ResNet)	**26.75**	**81.40**

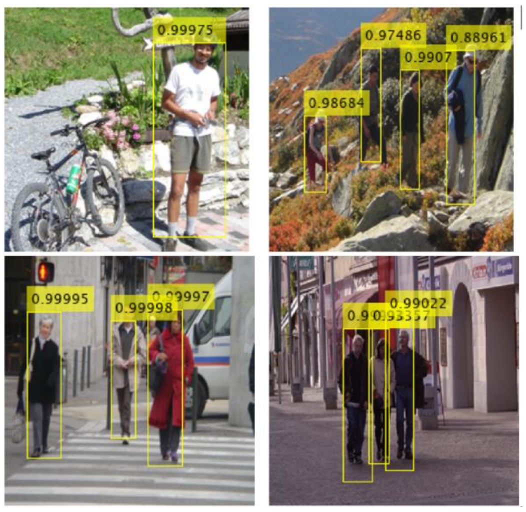


Figure 5.8 Sample INRIA Pedestrian output images with score and bounding boxes for the Proposed Faster R-CNN(DCResNet)

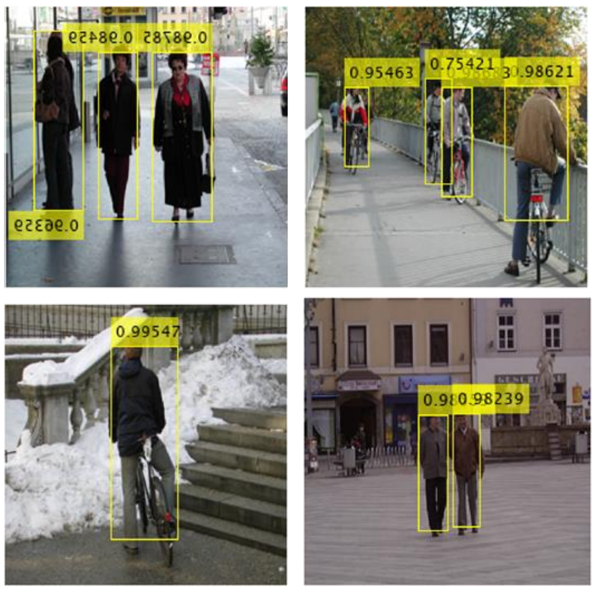


Figure 5.9 Sample INRIA Pedestrian output images with score and bounding boxes for the Proposed Faster R-CNN(MF2ResNet)

5.3.2 PASCAL VOC 2012 Dataset

The proposed FasterRCNN methods predicts the bounding box with confidence score for the PASCAL VOC 2012 test images. The generated prediction and the groundtruth bounding box are compared to yield the DET Curve and P-R Curve. The DET and P-R Curve are shown in Figure 5.10 and Figure 5.11 respectively. The LAMR and AP values are given in Table 5.5. The proposed Faster R-CNN(DCResNet) has achieved a minimum LAMR and AP improvement of 2.89% and 5.22% respectively when compared to the Faster RCNN methods. And it has achieved a minimum LAMR and AP improvement of 1.63% and 1.71% respectively when compared to the YOLOv2 and YOLOv3 methods. The proposed Faster R-CNN(MF2ResNet) has achieved a minimum LAMR and AP improvement of 1.57% and 3.29% respectively when compared to the Faster RCNN methods. And it has achieved a minimum LAMR and AP improvement of 0.31% and 0.38% respectively when compared to the YOLOv2 and YOLOv3 methods. Some sample output images for Faster R-CNN(DCResNet) and Faster R-CNN(MF2ResNet) are shown in Figure 5.12 and Figure 5.13 respectively.

5.3.3 Statistical Analysis

In this work, 11(k) independent variables or the detection methods and 2(N) datasets are involved. The rank table for Faster R-CNN(DCResNet) and Faster R-CNN(MF2ResNet) are given in Table 5.6 and Table 5.7 respectively.

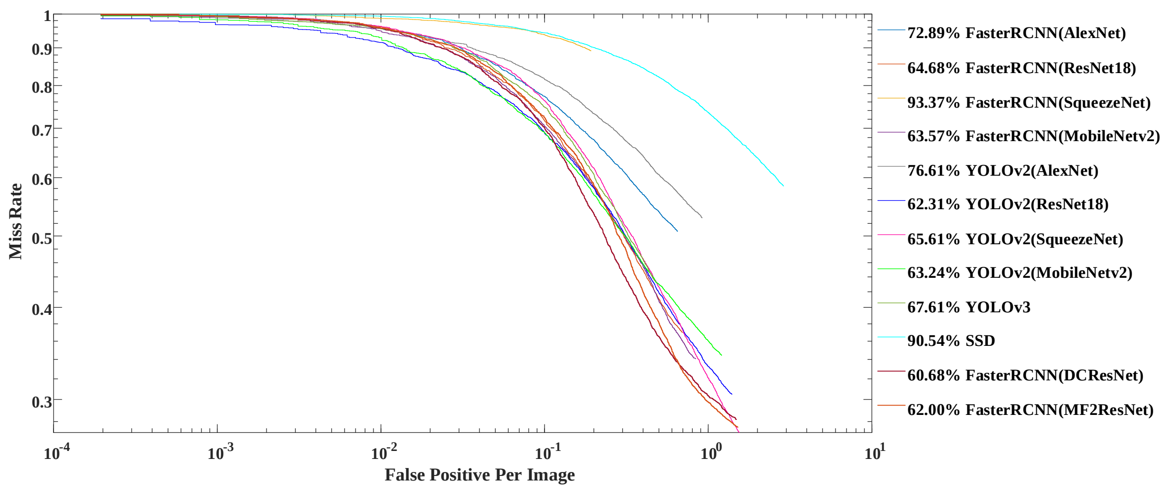


Figure 5.10 DET Curve for PASCAL VOC 2012 Dataset

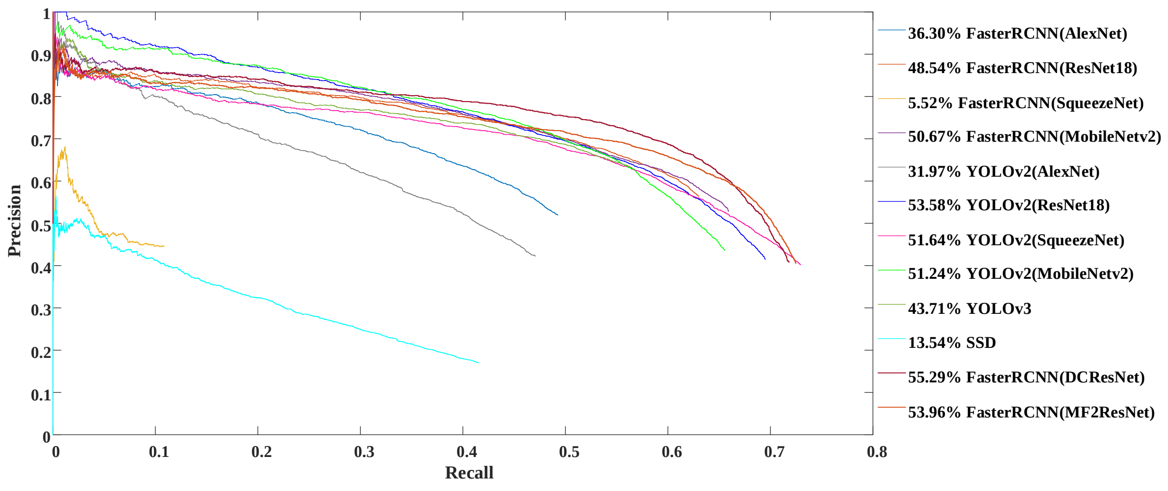


Figure 5.11 P-R Curve for PASCAL VOC 2012 Dataset

Table 5.5 LAMR and AP values for PASCAL VOC 2012 Dataset

Methods	**PASCAL VOC 2012**	
	LAMR	**AP**
FasterRCNN(AlexNet)	72.89	36.30
FasterRCNN(ResNet18)	64.68	48.54
FasterRCNN(SqueezeNet)	93.37	5.52
FasterRCNN(MobileNetv2)	63.57	50.67
YOLOv2(AlexNet)	76.61	31.97
YOLOv2(ResNet18)	62.31	53.58
YOLOv2(SqueezeNet)	65.61	51.64
YOLOv2(MobileNetv2)	63.24	51.24
YOLOv3	67.61	43.71
SSD	90.54	13.54
FasterRCNN(DCResNet)	**60.68**	**55.29**
FasterRCNN(MF2ResNet)	**62.00**	**53.96**

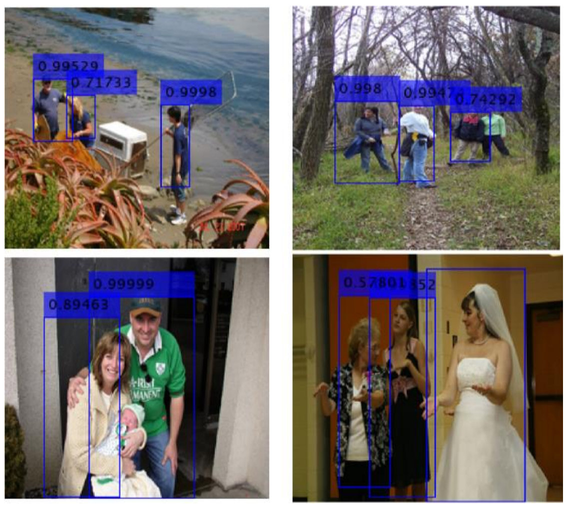


Figure 5.12 Sample PASCAL VOC 2012 output images with score and bounding boxes for the Proposed Faster R-CNN(DCResNet)

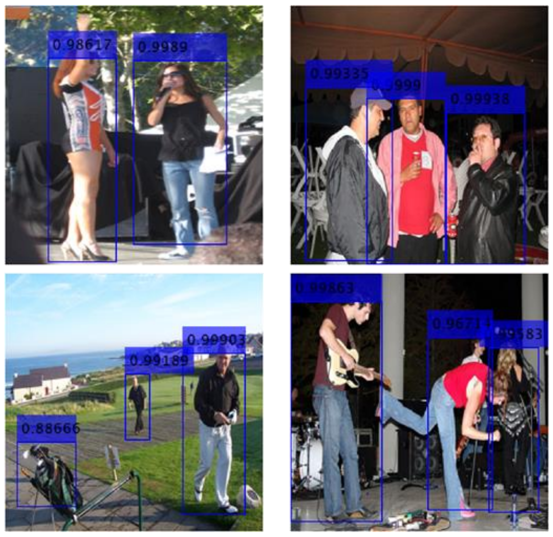


Figure 5.13 Sample PASCAL VOC 2012 output images with score and bounding boxes for the Proposed Faster R-CNN(MF2ResNet)

The proposed methods Faster R-CNN(DCResNet) and Faster R-CNN(MF2ResNet) takes the highest rank in the test. For conducting the test, first, the chi-square value is calculated which is 16.18. Then the F-distribution is calculated to be 4.235. At the degree of freedom (10,10), the critical value of F-distribution with α = 0.05 is 2.978. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

Table 5.6 Faster R-CNN(DCResNet) F-DISTRIBUTION Test Analysis

Methods	r_{INRIA}	$r_{PASCAVOC2012}$	**Mean Rank**	**Rank**
FasterRCNN(AlexNet)	7	8	7.50	7
FasterRCNN(ResNet18)	3	5	4.00	3
FasterRCNN(SqueezeNet)	11	11	11.00	10
FasterRCNN(MobileNetv2)	6	4	5.00	4
YOLOv2(AlexNet)	10	9	9.50	9
YOLOv2(ResNet18)	5	2	3.50	2
YOLOv2(SqueezeNet)	2	6	4.00	3
YOLOv2(MobileNetv2)	9	3	6.00	6
YOLOv3	4	7	5.50	5
SSD	8	10	9.00	8
FasterRCNN(DCResNet)	**1**	**1**	**1.00**	**1**

Table 5.7 Faster R-CNN(MF2ResNet) F-DISTRIBUTION Test Analysis

Methods	r_{INRIA}	$r_{PASCAVOC2012}$	**Mean Rank**	**Rank**
FasterRCNN(AlexNet)	7	8	7.50	7
FasterRCNN(ResNet18)	3	5	4.00	3
FasterRCNN(SqueezeNet)	11	11	11.00	10
FasterRCNN(MobileNetv2)	6	4	5.00	4
YOLOv2(AlexNet)	10	9	9.50	9
YOLOv2(ResNet18)	5	2	3.50	2
YOLOv2(SqueezeNet)	2	6	4.00	3
YOLOv2(MobileNetv2)	9	3	6.00	6
YOLOv3	4	7	5.50	5
SSD	8	10	9.00	8
FasterRCNN(MF2ResNet)	**1**	**1**	**1.00**	**1**

5.4 Observations

In this chapter, with the two proposed modifications of CNN: ResNet18, which serves as a base for Faster R-CNN, the particular observations are given here after applying on two datasets.

- As the feature map is formed by the concatenation of the processed output feature map of the proposed networks, it enabled a thorough feature extraction process to give a detailed hierarchical representation of the image.
- The proposed methods have shown substantial improvement on both benchmark datasets INRIA Pedestrian and PASCAL VOC 2012.
- The proposed methods have yielded the least LAMR and greatest AP when compared to Faster R-CNN, YOLOv2, YOLOv3, and SSD detection methods for both the datasets.

Chapter 6

YOLO based on DarkNet: Depth-wise Separable, Inception Depth-wise & Fire and SqueezeNet: Multiscale-Multilevel for Pedestrian Detection

An important aspect in the task of pedestrian detection is generation of region proposals from the input image. In previous contribution, the two-stage detection network: Faster RCNN was employed. The Faster R-CNN involved more computation overhead as it performed bounding box generation and classification in separate pipeline. To address this shortcoming, in this work, the single-stage network is used which performs the bounding box and classification in a single pipeline. This reduces the computation overhead and also saves the execution time. In this chapter, three base networks: YOLOv2(DarkNet19), YOLOv2(DarkNet53) and YOLOv3(SqueezeNet) are used. To increase the detection rate of the network, we have focused on enhancing the feature extraction network. As the feature map of only one (topmost) level is used; it can be further enhanced by also considering features from different levels of the base network. Techniques such as fire modules, inception modules and depth-wise separable convolution modules are incorporated within the base networks to assist in yielding a more detailed feature. The proposed YOLOv2 methods are named Depth-wiseSeparableModule-InceptionDepthwiseModule-YOLO (DSM-IDM-YOLO), InceptionDepth-wiseYOLOv2 and FireYOLOv2. The proposed YOLOv3 is method is named Multiscale-Multilevel-SqueezeNetYOLOv3 (MS-ML-SNYOLOv3).

6.1 Introduction

6.1.1 Depth-wise Separable Convolution Module

A depth-wise separable convolution consists of a) depth-wise convolution, followed by b) convolution of size 1×1. In the depth-wise convolution, each filter, of size h×h, will have one channel, i.e., h×h×1. Here, unlike the convolution of h×h×D, where D is the number of channels or depth, the D kernels are applied separately. Each D kernel convolves with one channel of the input feature map (of size H×W×D). To get the output feature map, the (H-h+1)×(W-h+1)×1 map are stacked together to form a depth of D. The second step of applying a 1×1 convolution is to increase the depth of the feature map which is obtained. Lastly, N number of 1×1×D convolutions are applied to get the final feature map of (H-h+1)×(W-h+1)×N. The overall process of depthwise separable convolution is shown in Figure 6.1.

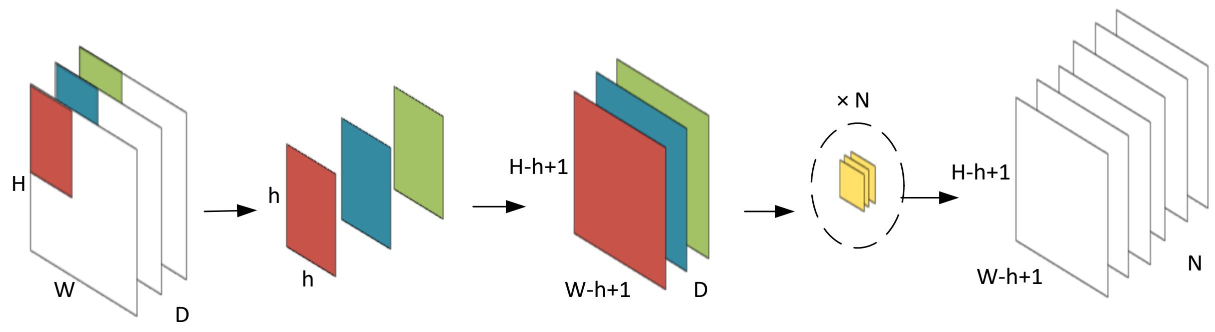


Figure 6.1 The overall feature map formation in Depth-wise Separable Convolution

6.1.2 Inception Module

A traditional inception module captures the features by using receptive fields of various sizes. The varying receptive fields are helpful in not missing out on any of the features of the object. The structure is shown in Figure 6.2.

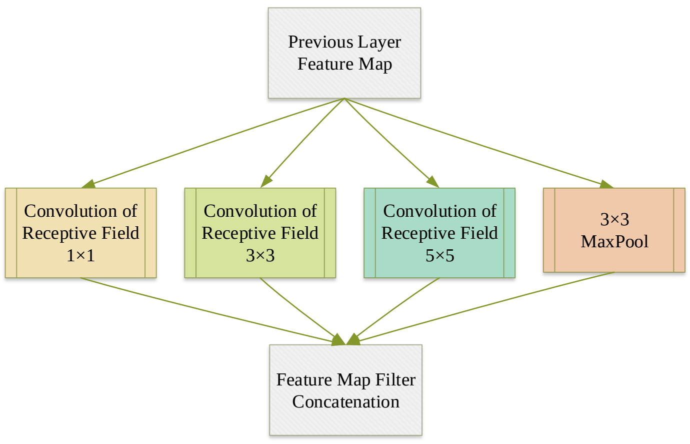

Previous Layer Feature Map
Convolution of Receptive Field 1×1
Convolution of Receptive Field 3×3
Convolution of Receptive Field 5×5
3×3 MaxPool
Feature Map Filter Concatenation

Figure 6.2 General Idea behind a traditional Inception Module

6.1.3 Squeeze and Expand Module

A Fire module comprises a squeeze convolution layer (which has only 1×1 filters), feeding into an expand layer that has a mix of 1×1 and 3×3 convolution filters. Its architecture is given in Figure 6.3. It serves two purposes:

- *Purpose 1:* Replace 3×3 filters with 1×1 filters. This reduces the number of training parameters.
- *Purpose 2:* Decrease the number of input channels to 3×3 filters. This reduces the number of channels which in turn implies less computation.

The squeeze layer decreases the number of input channels to 3×3 filters. The number of filters per fire module is gradually increased from the start to the end of the network.

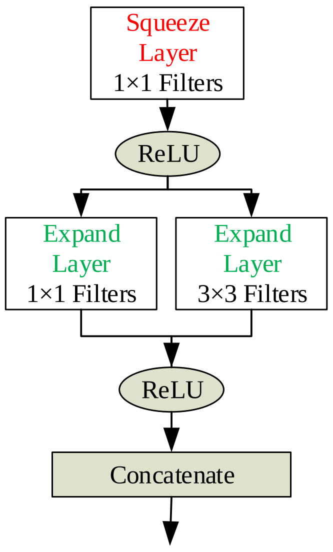


Figure 6.3 Fire Module

6.2 Methodology

In this section, the proposed YOLOv2 and YOLOv3 architecture is described in detail. The proposed YOLOv2 methods, i.e., DSM-IDM-YOLO, InceptionDepth-wiseYOLOv2 and FireYOLOv2 components are explained in Subsection 6.2.1, 6.2.2 and 6.2.3 respectively. The proposed YOLOv3 method, i.e., MS-ML-SNYOLOv3 architecture is explained in Subsection 6.2.4.

6.2.1 Depth-wiseSeparableModule-InceptionDepthwiseModule-YOLO

The proposed Depth-wiseSeparableModule-InceptionDepthwiseModule-YOLO is described in five components: Feature Blocks, Fusion, Depth-wise Separable Module I, Depth-wise Separable Module II and Inception Depth-wise Module, in this section.

- *Feature Blocks:* The DarkNet19 is divided into 6 blocks according to the output feature map which is of sizes 256×256×32, 128×128×64, 64×64×128, 32×32×256, 16×16×512 and 8×8×1024. The last three blocks are taken for the next component processing. The purpose here is to process the feature maps at different scales to capture more information about the presence of an object in the image. The representation of these blocks is shown in Figure 6.4.
- *Fusion:* The output from the three feature blocks is to be concatenated depthwise to form a feature map. However, as the feature maps differ in size, feature block four is applied with a MaxPool layer to yield a 16×16×256 feature map. Next, the output of feature block six is applied with a Rescale layer to generate a 16×16×1024 feature map. Then, the depth concatenation layer fuses the three feature maps, i.e., 16×16×256,

16×16×512, and 16×16×1024. The output size of this layer is 16×16×1792. The resultant feature is a detailed feature map from different scales. This process is shown under the 'Fusion' part of Figure 6.4.

- *Depth-wise Separable Module I:* This module takes the fused feature map of size 16×16×1792. The feature is applied with a group of 3×3×1 depth-wise convolution followed by a 1×1 convolution which projects or reduces the feature map. This step is done to reduce the dimension before going for further processing. The feature map is then processed again by a group of 3×3×1 depth-wise convolution. This action is preceded by a 1×1 convolution which expands the feature map and followed by a 1×1 convolution which projects the feature map. This expansion of feature map before a 3×3 convolution operation facilitates a detailed feature calculation. A skip connection is made in the network by adding the two projected convolution feature map. The output of this module is a feature map of size 16×16×896. This process is represented in Figure 6.5.
- *Depth-wise Separable Module II:* This module follows the depth-wise separable module I. The input here is a 16×16×896 feature map. The feature map is expanded and is followed by a group of 3×3×1 depth-wise convolution, which yields a detailed feature map. The resultant feature map is projected to reduce the dimension of the feature map. Here, a skip connection is made by adding the input and the projected feature map. The output of this module is a feature map of size 16×16×896. This process is shown in Figure 6.6.
- *Inception Depth-wise Module:* This module follows the depth-wise separable module II. In this module, three different depth-wise convolutions are applied. As some features can be missed by a receptive field and captured by another receptive field, three receptive fields are considered in the depth-wise convolution. They are a group of 1×1×1, 3×3×1, and 5×5×1 depth-wise convolution. These are preceded by 1×1 reducers. The inception depth-wise module yields a comprehensive feature map wherein various receptive fields are considered. The output of this module is a feature map of size 16×16×1280. This process is shown in Figure 6.7.

The DSM-IDM-YOLO provides rich hierarchical feature information owing to the depth-wise separable convolution and inception depth-wise convolution modules applied to fused feature maps from different levels of the network. In addition, a dropout layer (50%) is added at the end to avoid overfitting as the resultant feature map is quite dense. The detailed architecture is shown in Figure 6.4. Table 6.1 shows the DSM-IDM-YOLO feature formation layers characteristics. There are 32 layers in this network.

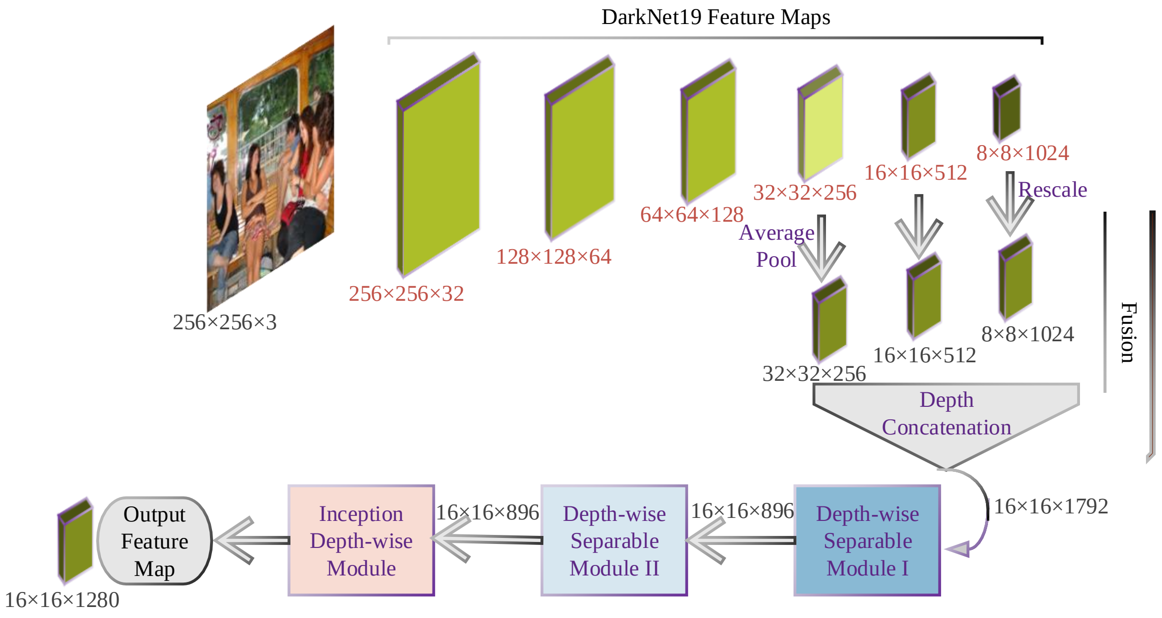


Figure 6.4 Block Diagram of the Proposed DSM-IDM-YOLO

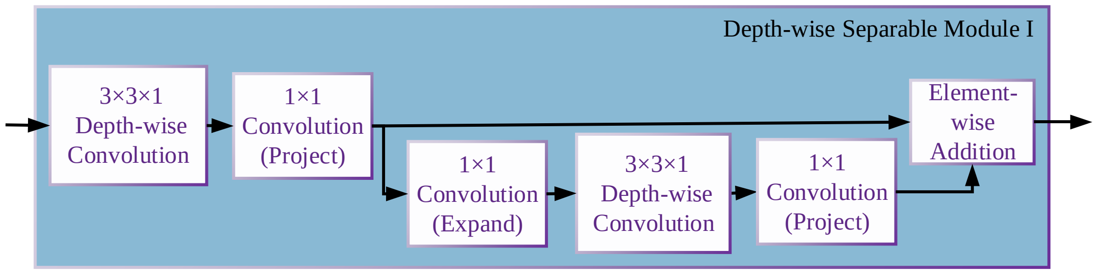

Depth-wise Separable Module I
3×3×1 Depth-wise Convolution
1×1 Convolution (Project)
1×1 Convolution (Expand)
3×3×1 Depth-wise Convolution
1×1 Convolution (Project)
Element-wise Addition

Figure 6.5 Block Diagram of Depth-wise Separable Module I of DSM-IDM-YOLO

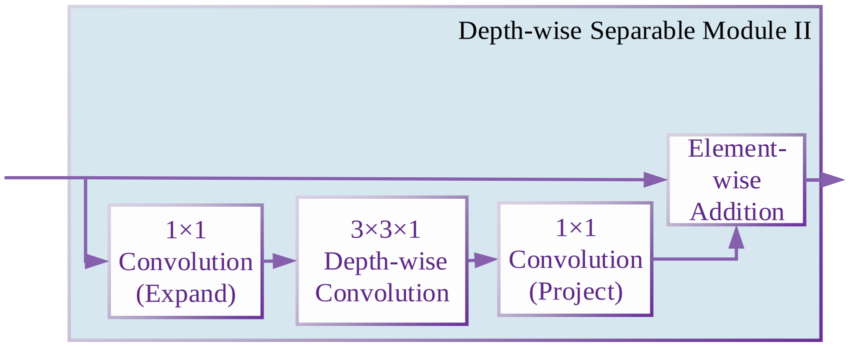

Depth-wise Separable Module II
Element-wise Addition
1×1 Convolution (Expand)
3×3×1 Depth-wise Convolution
1×1 Convolution (Project)

Figure 6.6 Block Diagram of Depth-wise Separable Module II of DSM-IDM-YOLO

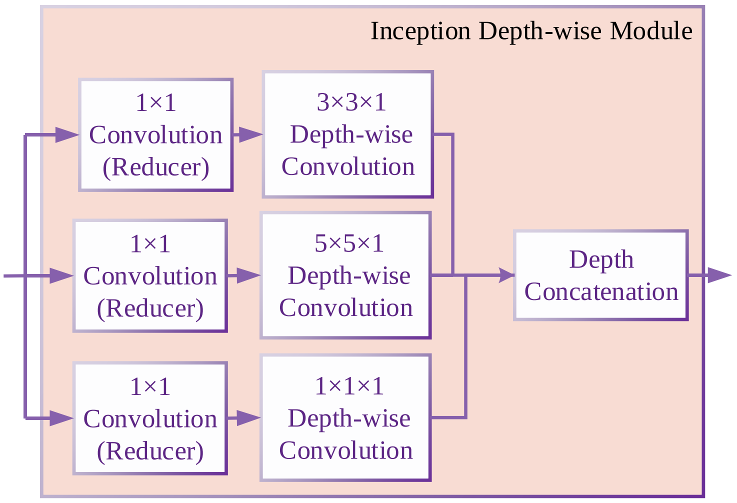

Inception Depth-wise Module
1×1 Convolution (Reducer)
3×3×1 Depth-wise Convolution
1×1 Convolution (Reducer)
5×5×1 Depth-wise Convolution
1×1 Convolution (Reducer)
1×1×1 Depth-wise Convolution
Depth Concatenation

Figure 6.7 Block Diagram of Inception Depth-wise Module of DSM-IDM-YOLO

6.2.2 InceptionDepth-wiseYOLOv2

In this work, an inception module involving depth-wise convolution is proposed:

a) The input feature map goes through dimension reduction. The dimension reduction is applied in different ratios for the 1×1, 3×3, and 5×5 receptive fields.

b) The feature map is applied with the depth-wise convolutions.

c) A batch normalization and ReLu Layer are added after both the reducer convolution layer and dept-wise convolution layer.

d) The feature maps are concatenated channel-wise to get the output.

The overall process is shown in Figure 6.8. The proposed InceptionDepth-wiseYOLOv2 is based on this idea.

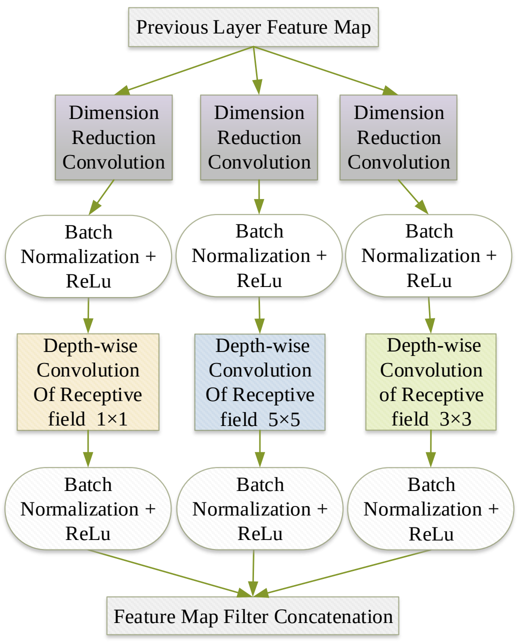

Previous Layer Feature Map
Dimension Reduction Convolution
Dimension Reduction Convolution
Dimension Reduction Convolution
Batch Normalization + ReLu
Batch Normalization + ReLu
Batch Normalization + ReLu
Depth-wise Convolution Of Receptive field 1×1
Depth-wise Convolution Of Receptive field 5×5
Depth-wise Convolution of Receptive field 3×3
Batch Normalization + ReLu
Batch Normalization + ReLu
Batch Normalization + ReLu
Feature Map Filter Concatenation

Figure 6.8 Idea behind the Proposed Inception Depth-wise Convolution Module

Table 6.1 DSM-IDM-YOLO Feature Layers architecture and characteristics

	Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
	Input				256×256×3
	Conv1	3×3/1	32	Input	256×256×32
	Maxpool_1	2×2/2		Conv1	128×128×32
	Conv2	3×3/1	64	Maxpool_1	128×128×64
	Maxpool_2	2×2/2		Conv2	64×64×64
	Conv3	1×1/1	128	Maxpool_2	64×64×128
	Conv4	3×3/1	64	Conv3	64×64×64
	Conv5	1×1/1	128	Conv4	64×64×128
	Maxpool_3	2×2/2		Conv5	32×32×128
	Conv6	1×1/1	256	Maxpool_3	32×32×256
	Conv7	3×3/1	128	Conv6	32×32×128
	Conv8	1×1/1	256	Conv7	32×32×256
	Maxpool_4	2×2/2		Conv8	16×16×256
	Conv6	1×1/1	512	Maxpool_4	16×16×512
	Conv7	3×3/1	256	Conv6	16×16×256
	Conv8	1×1/1	512	Conv7	16×16×512
	Conv9	3×3/1	256	Conv8	16×16×256
	Conv10	1×1/1	512	Conv9	16×16×512
	Maxpool_5	2×2/2		Conv10	8×8×512
	Conv11	1×1/1	1024	Maxpool_5	8×8×1024
	Conv12	3×3/1	512	Conv11	8×8×512
	Conv13	1×1/1	1024	Conv12	8×8×1024
	Conv14	1×1/1	512	Conv13	8×8×512
	Conv15	3×3/1	1024	Conv14	8×8×1024
Fusion	AveragePool	3×3/2		Conv8	16×16×256
	Rescale			Conv15	16×16×1024
	DepthConcatenationI			AveragePool	
				Conv10	
				Rescale	16×16×1792
Depth-wise Separable Module I	Depth-wiseConv1	3×3/1	1	DepthConcatenationI	16×16×1792
	ProjectConv1	1×1/1	896	Depth-wiseConv1	16×16×896
	ExpandConv1	1×1/1	1024	ProjectConv1	16×16×1024
	Depth-wiseConv2	3×3/1	1	ExpandConv1	16×16×1024
	ProjectConv2	1×1/1	896	Depth-wiseConv2	16×16×896
	Add_1			ProjectConv1	
				ProjectConv2	16×16×896
Depth-wise Separable Module II	ExpandConv2	1×1/1	1024	Add_2	16×16×1024
	Depth-wiseConv3	3×3/1	1	ExpandConv2	16×16×1024
	ProjectConv3	1×1/1	896	Depth-wiseConv3	16×16×896
	Add_2			Add_1	
				ProjectConv3	16×16×896
Inception Depth-wise Module	ReducerConv1	1×1/1	512	Add_2	16×16×512
	Depth-wiseConv4	3×3/1	1	ReducerConv1	16×16×512
	ReducerConv2	1×1/1	512	Add_2	16×16×512
	Depth-wiseConv5	1×1/1	1	ReducerConv2	16×16×512
	ReducerConv3	1×1/1	256	Add_2	16×16×256
	Depth-wiseConv6	5×5/1	1	ReducerConv3	16×16×256
	DepthConcatenationII			Depth-wiseConv4	
				Depth-wiseConv5	
				Depth-wiseConv6	16×16×1280

The Proposed InceptionDepth-wiseYOLOv2

The InceptionDepth-wiseYOLOv2 architecture flow can be explained with the following three components: Feature Map Blocks, Inception Depth-wise Convolution Module and Feature Fusion, in this section.

- *Feature Map Blocks:* The DarkNet53 is divided into five blocks according to the output feature map, which is of sizes 256×256×3, 128×128×64, 64×64×128, 32×32×256, 16×16×512, and 8×8×1024. The last three blocks are taken for the subsequent component processing. The purpose here is to process the feature maps at different scales to capture more information about the presence of an object in the image. The representation of these blocks is shown in Figure 6.9.
- *Inception Depth-wise Convolution Module:* In this module, three depth-wise convolutions with different receptive fields are applied. Some features can be missed by a receptive field and captured by another receptive field; three receptive fields are considered in the depth-wise convolution. They are a group of 1×1×1, 3×3×1, and 5×5×1 depth-wise convolution. Convolution of 1×1 dimension reducers precedes these. The inception depth-wise module yields a comprehensive feature map wherein various receptive fields are considered. The dimension reducers for the receptive fields 1×1, 3×3, and 5×5 are in the ratio of 2:2:1. The output feature map of modules I, II, and III are 32×32×256, 16×16×512, and 8×8×1024, as shown in Figure 6.9.
- *Feature Fusion:* The output from the three inception depth-wise convolution modules is concatenated channel-wise to form a feature map. However, as the feature maps differ in size, the output of inception depth-wise convolution module I is applied with a MaxPool layer to yield a 16×16×256 feature map. The inception depth-wise convolution module III output is applied with a Rescale layer to yield a 16×16×1024 feature map. The depth concatenation layer fuses the three feature maps, i.e., 16×16×256, 16×16×512, and 16×16×1024. The output size of this layer is 16×16×1792. The resultant feature is a detailed feature map from different scales. This process is shown under the 'Feature Fusion' part of Figure 6.9.

The InceptionDepth-wiseYOLOv2 provides rich hierarchical feature information owing to the inception depth-wise convolution modules applied at different levels of the network. In addition, a dropout layer (50%) is added at the end to avoid overfitting as the resultant feature map is quite dense. The resultant feature map is then processed by YOLOv2 Detection Layers.

The detailed architecture is shown in Figure 6.9. Table 6.2 shows the InceptionDepth-wiseYOLOv2 feature formation layers characteristics.

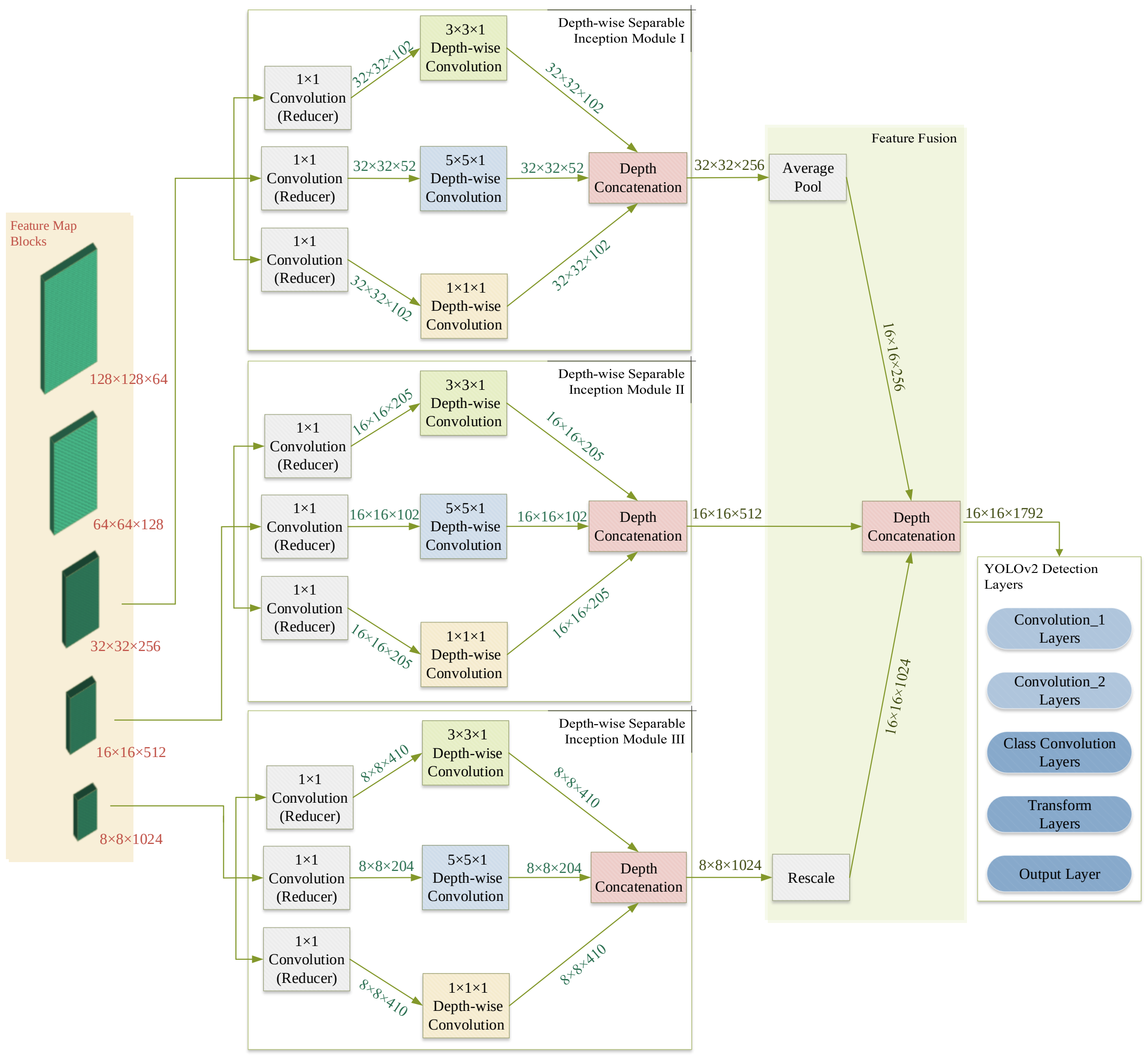


Figure 6.9 Block Diagram of InceptionDepth-wiseYOLOv2

6.2.3 FireYOLOv2

The FireYOLOv2 architecture flow can be explained with three components: Feature Blocks, Fire Modules and Fusion, in this section. A lightweight version of the proposed network is also given.

- *Feature Blocks:* The DarkNet53 is divided into 5 blocks according to the output feature map which is of sizes 256×256×3, 128×128×64, 64×64×128, 32×32×256, 16×16×512 and 8×8×1024. The last three blocks are taken for the next component processing. The purpose here is to process the feature maps at different scales to capture more information about the presence of an object in the image.

Table 6.2 InceptionDepth-wiseYOLOv2 Feature Layers architecture and characteristics

	Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
	Input				256×256×3
	conv1	3×3/1	32	Input	256×256×32
	conv2	3×3/2	64	conv1	128×128×64
1×	conv3	1×1/1	32	conv2	128×128×32
	conv4	3×3/1	64		128×128×64
	Residual_1				128×128×64
	conv5	3×3/2	128	Residual_1	64×64×128
2×	conv6	1×1/1	64	conv5	64×64×64
	conv7	3×3/1	128		64×64×128
	Residual_2				64×64×128
	conv10	3×3/2	256	Residual_2	32×32×256
8×	conv11	1×1/1	128	conv10	32×32×128
	conv12	3×3/1	256		32×32×256
	Residual_3				32×32×256
	conv27	3×3/2	512	Residual_3	16×16×512
8×	conv28	1×1/1	256	conv27	16×16×256
	conv29	3×3/1	512		16×16×512
	Residual_4				16×16×512
	conv44	3×3/2	1024	Residual_4	8×8×1024
4×	conv45	1×1/1	512	conv44	8×8×512
	conv46	3×3/1	1024		8×8×1024
Proposed Modification	Residual_5				8×8×1024
	ReducerConv1_3×3	1×1/1	102	Residual_3	32×32×102
	Depth-wiseConv1_3	3×3/1	1	ReducerConv1_3×3	32×32×102
	ReducerConv1_5×5	1×1/1	52	Residual_3	32×32×52
Inception Depth-wise Module I	Depth-wiseConv1_5	5×5/1	1	ReducerConv1_5×5	32×32×52
	ReducerConv1_1×1	1×1/1	102	Residual_3	32×32×102
	Depth-wiseConv1_1	1×1/1	1	ReducerConv1_1×1	32×32×102
	DepthConcatenationI			Depth-wiseConv1_3	
				Depth-wiseConv1_5	
				Depth-wiseConv1_1	32×32×256
	ReducerConv2_3×3	1×1/1	205	Residual_4	16×16×205
	Depth-wiseConv2_3	3×3/1	1	ReducerConv2_3×3	16×16×205
	ReducerConv2_5×5	1×1/1	102	Residual_4	16×16×102
Inception Depth-wise Module II	Depth-wiseConv2_5	5×5/1	1	ReducerConv2_5×5	16×16×102
	ReducerConv2_1×1	1×1/1	205	Residual_4	16×16×205
	Depth-wiseConv2_1	1×1/1	1	ReducerConv2_1×1	16×16×205
	DepthConcatenationII			Depth-wiseConv2_3	
				Depth-wiseConv2_5	
				Depth-wiseConv2_1	16×16×512
	ReducerConv3_3×3	1×1/1	410	Residual_5	8×8×410
	Depth-wiseConv3_3	3×3/1	1	ReducerConv3_3×3	8×8×410
	ReducerConv3_5×5	1×1/1	204	Residual_5	8×8×204
Inception Depth-wise Module III	Depth-wiseConv3_5	5×5/1	1	ReducerConv3_5×5	8×8×204
	ReducerConv3_1×1	1×1/1	410	Residual_5	8×8×410
	Depth-wiseConv3_1	1×1/1	1	ReducerConv3_1×1	8×8×410
	DepthConcatenationIII			Depth-wiseConv3_3	
				Depth-wiseConv3_5	
				Depth-wiseConv3_1	8×8×1024
	AveragePool	3×3/2		DepthConcatenationI	16×16×256
Feature Map Fusion	Rescale			DepthConcatenationIII	16×16×1024
	DepthConcatenationIV			AveragePool	
				DepthConcatenationII	
				Rescale	16×16×1792
	Dropout (50%)			DepthConcatenationIV	16×16×1792

- *Fire Modules:* Three fire modules are attached to the output of the last three blocks. The squeeze convolution filters and the expand convolution filters are in the ratio of 0.125. In the expand section, 50% of filters are of size 3×3. The expand section's 1×1 and 3×3 conv filters are depth concatenated to get the output. The size of the output feature map of FireModule1, FireModule2 and FireModule3 is 32×32×256, 16×16×512 and 8×8×1024 respectively. The purpose of these fire modules is to apply more conv filters on the input feature maps to get a more fine-tuned output. The squeeze layer helps in reducing the dimension and the expand layer's role is to process the filters with more conv filters with receptive fields 1×1 and 3×3.
- *Fusion:* The output from the three fire modules is to be concatenated depthwise to form a feature map. However, as the feature maps differ in size; the output of fire module 1 is applied with a MaxPool layer to yield a 16×16×256 feature map. The output of fire module 3 is applied with a Transposed Conv layer to yield a 16×16×1024 feature map. The depth concatenation layer fuses the three feature maps i.e., 16×16×256, 16×16×512 and 16×16×1024. The output size of this layer is 16×16×1792. The resultant feature is a detailed feature map from different scales.

The FireYOLOv2 provides rich hierarchical feature information owing to the fire modules applied at different levels of the network. A dropout layer (50%) is added at the end to avoid overfitting as the resultant feature map is quite dense. The detailed architecture is shown in Figure 6.10. Table 6.3 shows the FireYOLOv2 feature formation layers characteristics.

LightWeight FireYOLOv2

A LightWeight version of FireYOLOv2 is proposed in this work. In this version, the number of convolution filters in both the squeeze and expand layers of the fire modules are reduced by a factor of 2. Consequently, the output feature map size of FireModule1, FireModule2 and FireModule3 are 32×32×128, 16×16×256 and 8×8×512 respectively. Following the same procedure for *Fusion* as in FireYOLOv2, the output feature map is of size 16×16×896.

6.2.4 Multiscale-Multilevel-SqueezeNetYOLOv3

The Multiscale-Multilevel-SqueezeNetYOLOv3 architecture flow can be explained with four components: The Feature Maps, Squeeze and Expand Blocks, Fusion and YOLOv3 Detection, in this section. The hyperparameters associated with this network is also discussed.

- *The Feature Maps:* The SqueezeNet base has five feature map divisions. The feature maps have dimensions 113×113×64, 56×56×64, 28×28×128, 14×14×256, and 14×14×512 respectively. There are two, three and four Squeeze and Expand blocks after feature map 2, 3, and 4 respectively. The feature map block 2 and 3 i.e., 56×56×64 and 28×28×128 are selected for further processing to incorporate it in the detection pipeline. This is done to achieve feature details at each level of the network. These details are reflected in the 'SqueezeNet' section of Figure 6.11.

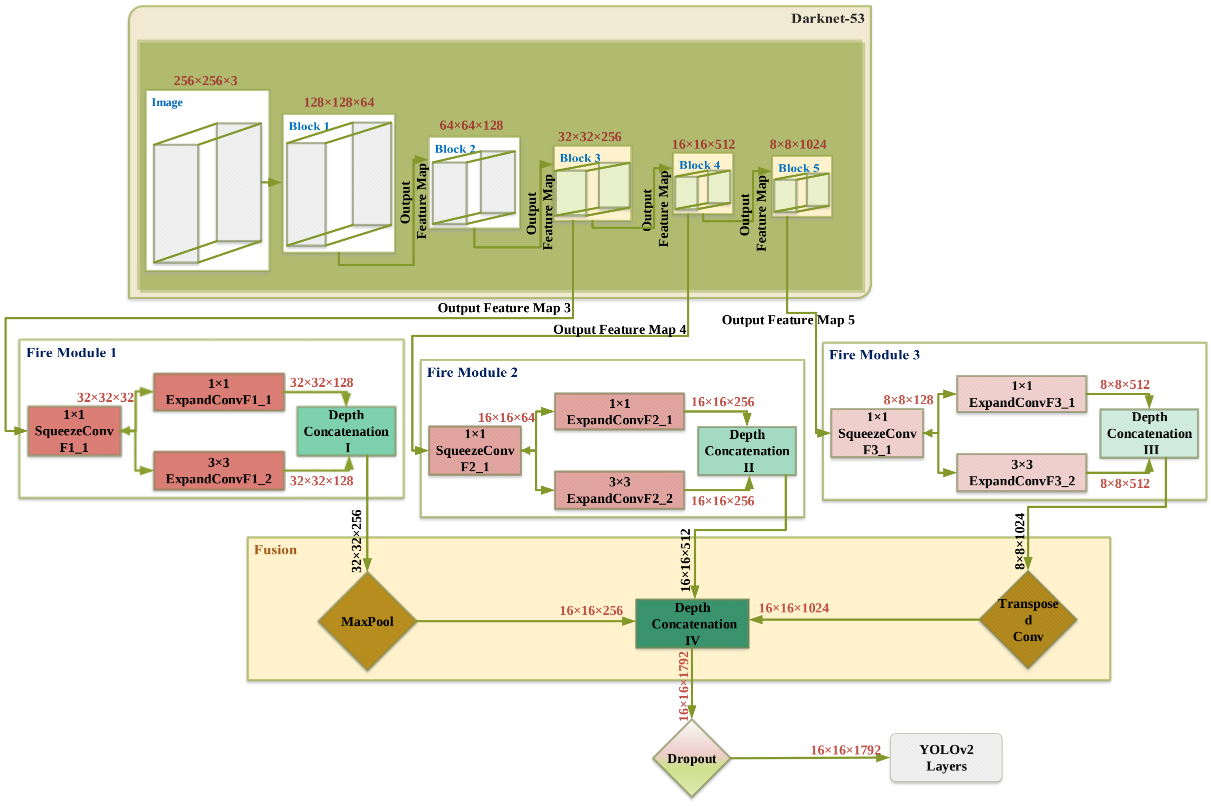


Figure 6.10 Block Diagram of FireYOLOv2

- *Squeeze & Expand Blocks:* The proposed two Squeeze & Expand blocks process input from feature maps 2 and 3. The input goes to a squeeze conv which is of receptive field 1×1. In the Expand section, there are 3 conv of receptive field 1×1, 3×3, and 5×5. 50% of the filters are allotted to the 1×1 filter in the expand section. The remaining 50% is divided between 3×3 and 5×5 filters in the ratio of 3:1. The various receptive fields capture the image features at different scales. The representation can be seen in the 'Proposed Modification' part of Figure 6.11.
- *Fusion:* The feature map generated from proposed Squeeze & Expand Block 1 and 2 is of size 56×56×N_1 and 28×28×N_2. To fuse these two, they are made into the same size. To achieve this task an average pool is applied with stride 2 and size 3×3 upon the feature map generated from Block 1. The resultant is a feature map of size 28×28×N_1. The next step is to apply a depth concatenation on Block 1 and Block 2 feature map, which yields a 28×28×(N_1+ N_2) size feature map. This gives us a combined feature representation of the different levels

as well as of the different scale of the network. The representation can be seen in the ‘Fusion’ part of Figure 6.11.

Table 6.3 FireYOLOv2 Feature Layers architecture and characteristics

	Layer Name	**Filter Size/Stride**	**Channels**	**Input Layer**	**Output Size**
	Input				256×256×3
	conv1	3×3/1	32	Input	256×256×32
	conv2	3×3/2	64	conv1	128×128×64
1×	conv3	1×1/1	32	conv2	128×128×32
	conv4	3×3/1	64		128×128×64
	Residual_1				128×128×64
	conv5	3×3/2	128	Residual_1	64×64×128
2×	conv6	1×1/1	64	conv5	64×64×64
	conv7	3×3/1	128		64×64×128
	Residual_2				64×64×128
	conv10	3×3/2	256	Residual_2	32×32×256
8×	conv11	1×1/1	128	conv10	32×32×128
	conv12	3×3/1	256		32×32×256
	Residual_3				32×32×256
	conv_S_1	1×1/1	32	Residual_3	32×32×32
	conv_Ex_1_1	1×1/1	128	conv_S_1	32×32×128
Fire Module-1	conv_Ex_1_2	1×1/1	128	conv_S_1	32×32×128
	depthconcat_1			conv_Ex_1_1, conv_Ex_1_2	32×32×256
	maxpool	3×3/2		depthconcat_1	16×16×256
	conv27	3×3/2	512	Residual_3	16×16×512
8×	conv28	1×1/1	256	conv27	16×16×256
	conv29	3×3/1	512		16×16×512
	Residual_4				16×16×512
	conv_S_2	1×1/1	64	Residual_4	16×16×64
	conv_Ex_2_1	1×1/1	256	conv_S_2	16×16×256
Fire Module-2	conv_Ex_2_2	1×1/1	256	conv_S_2	16×16×256
	depthconcat_2			conv_Ex_2_1, conv_Ex_2_2	16×16×512
	conv44	3×3/2	1024	Residual_4	8×8×1024
4×	conv45	1×1/1	512	conv44	8×8×512
	conv46	3×3/1	1024		8×8×1024
	Residual_5				8×8×1024
	conv_S_3	1×1/1	128	Residual_4	8×8×128
	conv_Ex_3_1	1×1/1	512	conv_S_3	8×8×512
Fire Module-3	conv_Ex_3_2	1×1/1	512	conv_S_3	8×8×512
	depthconcat_3			conv_Ex_3_1, conv_Ex_3_2	8×8×512
	transposedConv	3×3/2		depthconcat_3	16×16×512
	depthconcat_4			depthconcat_1, depthconcat_2, depthconcat_3	16×16×1792
	droupout			depthconcat_4	16×16×1792

- *YOLOv3 Detection:* The YOLOv3 detection uses two feature maps. First, the Feature Map Block 5 of size 14×14×512. This is converted to 14×14×256 by detection conv 1 and then to the output tensor 14×14×18. The second feature map is taken by concatenating the Fused result with detection conv 1. To complete this step, the detection conv 1 is upsampled to 28×28×256 and then depth concatenated with the Fused result of 28×28×(N_1+ N_2). The resultant is a feature map of 28×28×(N_1+ N_2+256). This is converted to 28×28×128 by detection conv 2 and lastly, the output tensor 28×28×18 is yielded. The representation can be seen in the 'YOLOv3 Detection' part of Figure 6.11.

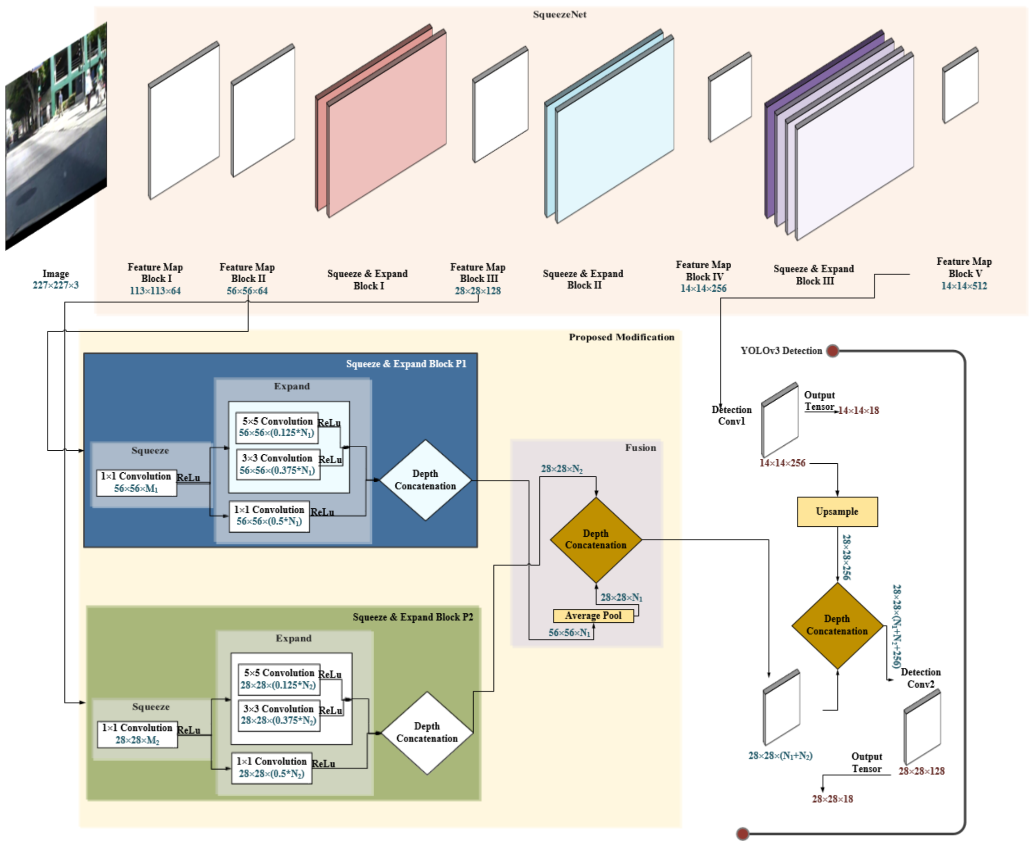

SqueezeNet
Image 227×227×3
Feature Map Block I 113×113×64
Feature Map Block II 56×56×64
Squeeze & Expand Block I
Feature Map Block III 28×28×128
Squeeze & Expand Block II
Feature Map Block IV 14×14×256
Squeeze & Expand Block III
Feature Map Block V 14×14×512
Proposed Modification
Squeeze & Expand Block P1
Expand
5×5 Convolution 56×56×(0.125*N1) ReLu
3×3 Convolution 56×56×(0.375*N1) ReLu
Squeeze
1×1 Convolution 56×56×M1 ReLu
1×1 Convolution 56×56×(0.5*N1) ReLu
Depth Concatenation
Fusion
28×28×N2
Depth Concatenation
28×28×N1
Average Pool
56×56×N1
Squeeze & Expand Block P2
Expand
5×5 Convolution 28×28×(0.125*N2) ReLu
3×3 Convolution 28×28×(0.375*N2) ReLu
Squeeze
1×1 Convolution 28×28×M2 ReLu
1×1 Convolution 28×28×(0.5*N2) ReLu
Depth Concatenation
YOLOv3 Detection
Detection Conv1
Output Tensor 14×14×18
14×14×256
Upsample
28×28×256
Depth Concatenation
28×28×(N1+N2+256)
Detection Conv2
28×28×(N1+N2)
Output Tensor
28×28×128
28×28×18

Figure 6.11 Block Diagram of the Proposed MS-ML-SNYOLOv3. The modification made are shown in the 'Proposed Modification' section

Hyperparameters

- It is to be noted that the filter distribution in Expand section is made in the ratio of 0.5, 0.375 and 0.125 for 1×1, 3×3, and 5×5 respectively. The number of filters in Squeeze section of P1 and P2 blocks are represented by M_1 and M_2 respectively. And for the expand section they are represented by N_1 and N_2 respectively.
- The ratio of the filters between Expand and Squeeze section i.e., between N_1 and M_1 & N_2 and M_2 can be varied according to the requirement. In simple small datasets, the ratio can be less while in complex and large datasets, the ratio can be more to avoid overfitting and underfitting respectively.

Table 6.4 Layers and Feature Map characteristics of proposed MS-ML-SNYOLOv3

	Layer Name	Filter Size/Stride	Channels	Input Layer	Output Size
	Input				227×227×3
	Conv1	3×3/2	64	Input	113×113×64
	MaxPool1	3×3/2		Conv1	56×56×64
Squeeze & Expand Block I	SqueezeConv1	1×1/1	16	MaxPool1	56×56×16
	ExpandConv1_1×1	1×1/1	64	SqueezeConv1	56×56×64
	ExpandConv1_3×3	3×3/1	64	SqueezeConv1	56×56×64
	DepthConcat1			ExpandConv1_1×1	
				ExpandConv1_3×3	56×56×128
	SqueezeConv2	1×1/1	16	DepthConcat1	56×56×16
	ExpandConv2_1×1	1×1/1	64	SqueezeConv2	56×56×64
	ExpandConv2_3×3	3×3/1	64	SqueezeConv2	56×56×64
	DepthConcat2			ExpandConv2_1×1	
				ExpandConv2_3×3	56×56×128
	MaxPool2	3×3/2		DepthConcat2	28×28×128
Squeeze & Expand Block II	SqueezeConv3	1×1/1	32	MaxPool2	28×28×32
	ExpandConv3_1×1	1×1/1	128	SqueezeConv3	28×28×128
	ExpandConv3_3×3	3×3/1	128	SqueezeConv3	28×28×128
	DepthConcat3			ExpandConv3_1×1	
				ExpandConv3_3×3	28×28×256
	SqueezeConv4	1×1/1	32	DepthConcat3	28×28×32
	ExpandConv4_1×1	1×1/1	128	SqueezeConv4	28×28×128
	ExpandConv4_3×3	3×3/1	128	SqueezeConv4	28×28×128
	DepthConcat4			ExpandConv4_1×1	
				ExpandConv4_3×3	28×28×256
	MaxPool3	3×3/2		DepthConcat4	14×14×128
Squeeze & Expand Block III	SqueezeConv5	1×1/1	48	MaxPool3	14×14×48
	ExpandConv5_1×1	1×1/1	192	SqueezeConv5	14×14×192
	ExpandConv5_3×3	3×3/1	192	SqueezeConv5	14×14×192
	DepthConcat5			ExpandConv5_1×1	
				ExpandConv5_3×3	14×14×384
	SqueezeConv6	1×1/1	48	DepthConcat5	14×14×48
	ExpandConv6_1×1	1×1/1	192	SqueezeConv6	14×14×192
	ExpandConv6_3×3	3×3/1	192	SqueezeConv6	14×14×192
	DepthConcat6			ExpandConv6_1×1	
				ExpandConv6_3×3	14×14×384
	SqueezeConv7	1×1/1	64	DepthConcat6	14×14×64
	ExpandConv7_1×1	1×1/1	256	SqueezeConv7	14×14×256
	ExpandConv7_3×3	3×3/1	256	SqueezeConv7	14×14×256
	DepthConcat7			ExpandConv7_1×1	
				ExpandConv7_3×3	14×14×512
	SqueezeConv8	1×1/1	64	DepthConcat7	14×14×64
	ExpandConv8_1×1	1×1/1	256	SqueezeConv8	14×14×256
	ExpandConv8_3×3	3×3/1	256	SqueezeConv8	14×14×256
	DepthConcat8			ExpandConv8_1×1	
				ExpandConv8_3×3	14×14×512
Proposed Squeeze & Expand Block I	SqueezeConvP1	1×1/1	32	DepthConcat2	56×56×32
	ExpandConvP1_1×1	1×1/1	192	SqueezeConvP1	56×56×192
	ExpandConvP1_3×3	3×3/1	144	SqueezeConvP1	56×56×144
	ExpandConvP1_5×5	5×5/1	48	SqueezeConvP1	56×56×48
	DepthConcatP1			ExpandConvP1_1×1	
				ExpandConvP1_3×3	
				ExpandConvP1_5×5	56×56×384
	AveragePool	3×3/2		DepthConcatP1	28×28×384
Proposed Squeeze & Expand Block I	SqueezeConvP2	1×1/1	48	DepthConcat4	28×28×48
	ExpandConvP2_1×1	1×1/1	288	SqueezeConvP1	28×28×288
	ExpandConvP2_3×3	3×3/1	216	SqueezeConvP1	28×28×216
	ExpandConvP2_5×5	5×5/1	72	SqueezeConvP1	28×28×72
	DepthConcatP2			ExpandConvP1_1×1	
				ExpandConvP1_3×3	
				ExpandConvP1_5×5	28×28×576
	Conv1Detection1	3×3/1	256	DepthConcat8	14×14×256
	OutputTensor1	1×1/1	18	Conv1Detection1	14×14×18
	Upsample	2		Conv1Detection1	28×28×256
	DepthConcatDetection			DepthConcatP1	
				DepthConcatP2	
				Upsample	28×28×1216
	Conv1Detection2	3×3/1	128	DepthConcatDetection	28×28×128
	OutputTensor2	1×1/1	18	Conv1Detection1	28×28×18

6.3 Experimental Results and Discussion

The proposed YOLOv2 methods i.e., DSM-IDM-YOLO, InceptionDepth-wiseYOLOv2 and FireYOLOv2 are evaluated using three standard pedestrian datasets, INRIA Pedestrian dataset, PASCAL VOC 2012 dataset and Caltech Pedestrian dataset. The proposed methods are compared against four state-of-the-art detection methods, i.e., FasterRCNN, YOLOv2, YOLOv3, and SSD. Nine different base networks are considered for YOLOv2 for comparison of results. They are AlexNet, ResNet18, ResNet50, Inceptionv3, Xception, SqueezeNet, MobileNetv2, DarkNet19 and DarkNet53. For the three datasets, per-image evaluation is employed. For INRIA, training parameters such as batch size is set to 8, the learning rate 10^{-4}, the number of epochs to 30 and the number of anchors is 4. Whereas for PASCAL VOC 2012, training parameters such as batch size is set to 16, the learning rate 10^{-4}, the number of epochs to 30 and the number of anchors is 4. Caltech pedestrian dataset is trained with batch size 8 and learning rate 10^{-4} for 50 epochs with 6 anchors. The optimizer used is *adam* for training in all the datasets.

The proposed YOLOv3 method i.e., MS-ML-SNYOLOv3 is evaluated using two standard pedestrian datasets, the INRIA Pedestrian dataset and the Caltech Pedestrian dataset. The proposed method is compared against state-of-the-art methods i.e., FasterRCNN, YOLOv2, and YOLOv3 with various base networks and SSD. In YOLOv2, base networks AlexNet, ResNet18, ResNet50, Inceptionv3, Xception, SqueezeNet, MobileNetV2, DarkNet19, and DarkNet53 are considered. For YOLOv3, two base networks, ResNet18 and SqueezeNet are considered. For the INRIA Pedestrian dataset, the training hyperparameters such as batch size, the learning rate, the number of epochs, and the number of anchors is set to 16, 10^{-3}, 70, and 6 respectively. For the Caltech Pedestrian dataset, the training hyperparameters batch size, the learning rate, the number of epochs, and the number of anchors is set to 16, 10^{-3}, 100, and 6 respectively. The ratio of the filters between expand and squeeze section is 8 and 12 respectively for INRIA and Caltech Pedestrian dataset. *SGDM* optimizer was used to train the datasets.

The model is trained using NVIDIA Geforce RTX 2070 16GB GPU on MATLAB R2021a. The prediction of models is yielded in a bounding box format. Subsection 6.3.1, 6.3.2 and 6.3.3 show the comparison for the proposed methods with INRIA Pedestrian dataset, PASCAL VOC 2012 dataset and Caltech Pedestrian dataset respectively. Section 6.3.4 shows the statistical test analysis for the proposed methods.

6.3.1 INRIA Pedestrian Dataset

Proposed YOLOv2 Methods

The proposed YOLOv2 methods generate bounding box and confidence score. This is compared with the groundtruth values to yield the DET Curve and P-R Curve. The DET Curve and P-R Curve for the YOLOv2 methods are shown in Figure 6.12 and Figure 6.13 respectively. The corresponding LAMR and AP values are shown in Table 6.5. The proposed DSM-IDM-YOLO has achieved the least miss rate and highest precision with 27.68% and 79.84%, respectively. A minimum improvement of 2.97% and 5.74% is attained by DSM-IDM-YOLO w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.14. The proposed InceptionDepth-wiseYOLOv2 has achieved the least miss rate and highest precision with 21.92% and 82.95%, respectively. A minimum improvement of 8.73% and 7.51% is attained by InceptionDepth-wiseYOLOv2 w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.15. The proposed FireYOLOv2 has achieved the least miss rate and highest precision with 19.60% and 85.27% respectively. LightWeight FireYOLOv2 has the second least miss rate of 26.30% and second highest precision of 82.45%. An improvement of 11.05% and 9.83% is attained by FireYOLOv2 w.r.t. LAMR and AP respectively whereas for LightWeight FireYOLOv2 an improvement of 4.35% and 7.01% respectively is gained. Some sample groundtruth and output images for comparison is shown in Figure 6.16.

Scale Analysis

The INRIA Pedestrian dataset is dominated by pedestrians of 'Large' scale i.e., height > 80. On evaluation of the height of the predicted pedestrians the following observation are noted. The Scale Analysis is represented in Figure 6.17.

- The DSM-IDM-YOLO detects the third greatest number of 'Large' scale pedestrians which is 485.
- The InceptionDepth-wiseYOLOv2 detects the second greatest number of 'Large' scale pedestrians, which is 496.
- The FireYOLOv2 detects the greatest number of 'Large' scale pedestrians which is 512.

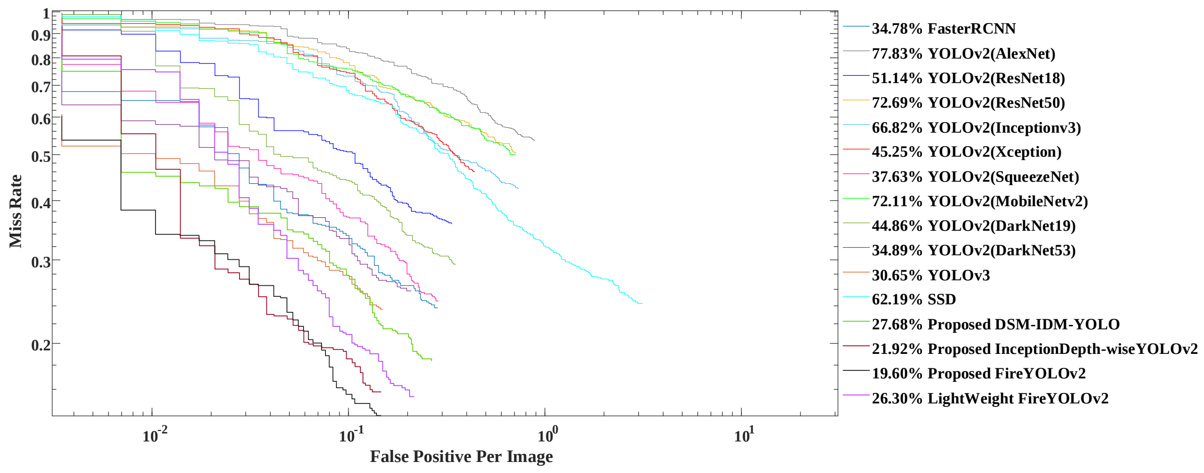


Figure 6.12 DET Curve of INRIA Pedestrian Dataset for YOLOv2 Methods

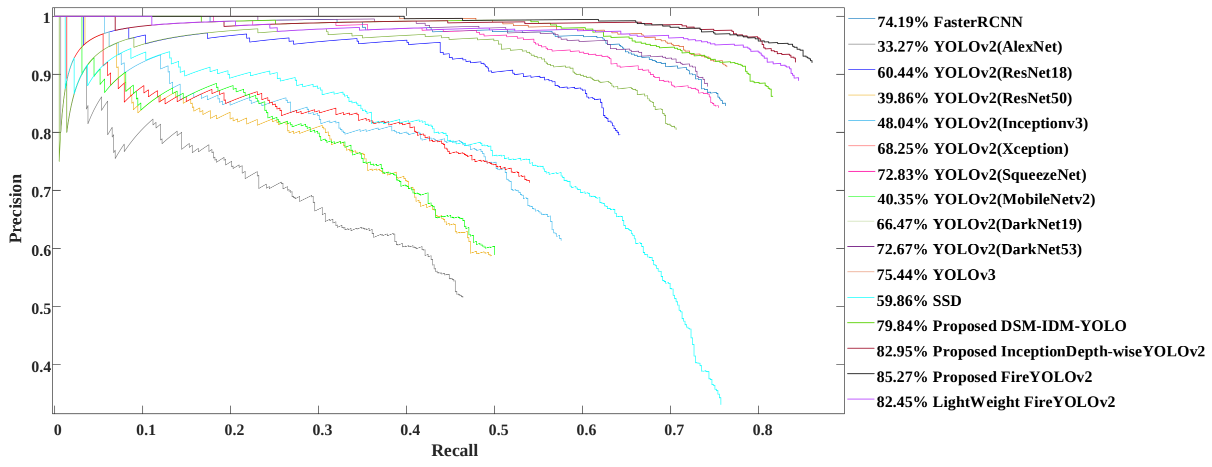


Figure 6.13 P-R Curve of INRIA Pedestrian Dataset for YOLOv2 Methods

Table 6.5 LAMR and AP values of INRIA Pedestrian Dataset for YOLOv2 Methods

Methods	**INRIA**	
	LAMR	**AP**
FasterRCNN	34.78	74.19
YOLOv2(AlexNet)	77.83	33.27
YOLOv2(ResNet18)	51.14	60.44
YOLOv2(ResNet50)	72.69	39.86
YOLOv2(Inceptionv3)	66.82	48.04
YOLOv2(Xception)	45.25	68.25
YOLOv2(SqueezeNet)	37.63	72.83
YOLOv2(MobileNetv2)	72.11	40.35
YOLOv2(DarkNet19)	44.86	66.47
YOLOv2(DarkNet53)	34.89	72.67
YOLOv3	30.65	75.44
SSD	62.19	59.86
Proposed DSM-IDM-YOLO	**27.68**	**79.84**
Proposed InceptionDepth-wiseYOLOv2	**21.92**	**82.95**
Proposed FireYOLOv2	**19.60**	**85.27**
LightWeight FireYOLOv2	**26.30**	**82.45**

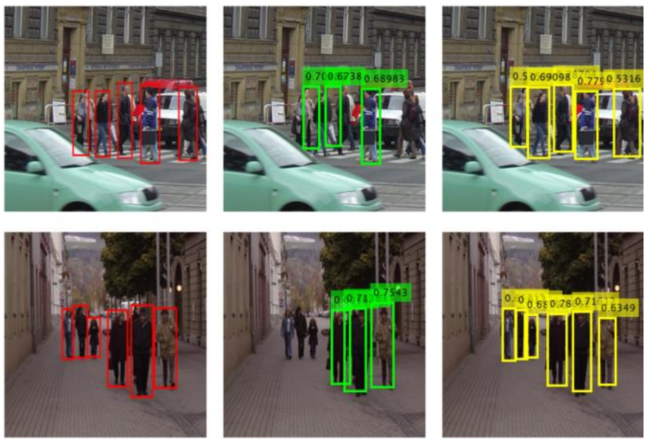


Figure 6.14 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO

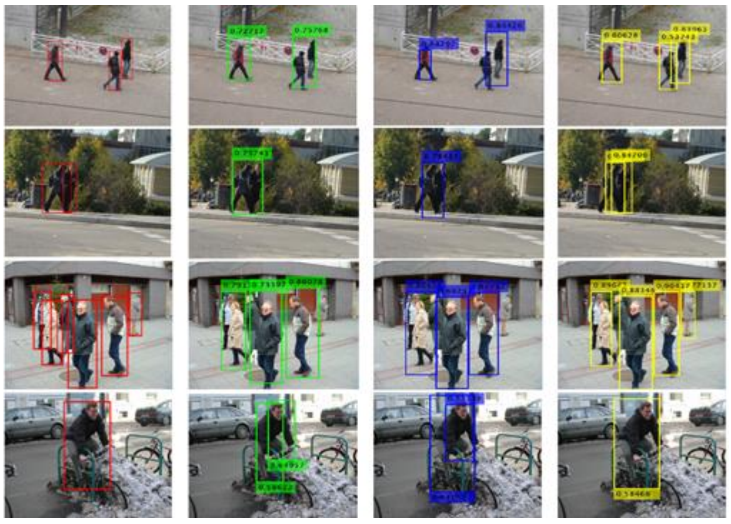


Figure 6.15 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2

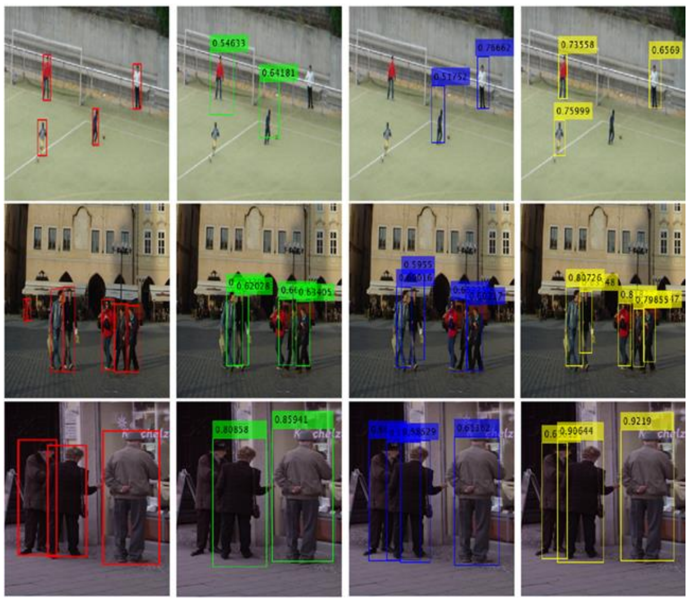


Figure 6.16 INRIA Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2

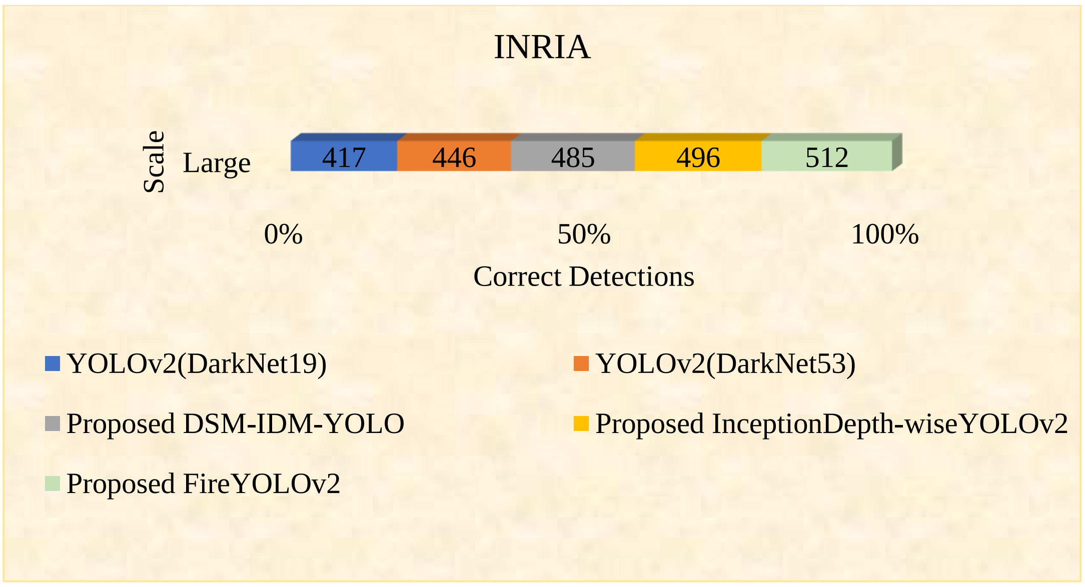


Figure 6.17 Scale Analysis of INRIA Pedestrian Dataset for YOLOv2 Methods

Proposed YOLOv3 Method

The proposed YOLOv3 method generates bounding box along with respective confidence score. When these are compared with the groundtruth values the DET Curve and P-R Curve are yielded. The DET Curve and P-R Curve for the YOLOv3 methods are shown in Figure 6.18 and Figure 6.19 respectively. The corresponding LAMR and AP values are shown in Table 6.6. The proposed MS-ML-SNYOLOv3 obtains a LAMR of 28.36% and AP of 76.89%. It has achieved a minimum performance gain of 2.29% and 1.45% in miss rate and precision respectively.

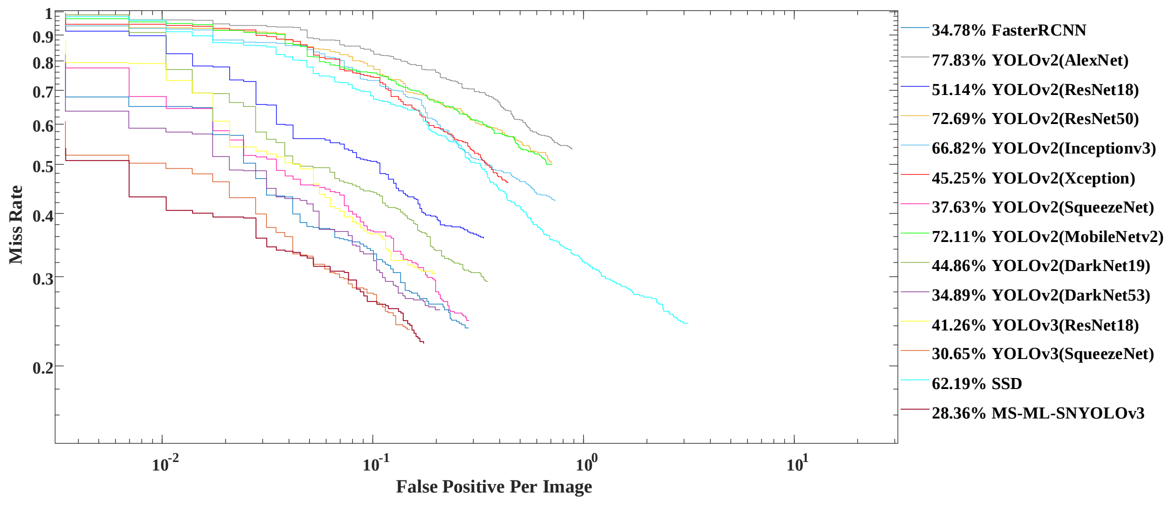


Figure 6.18 DET Curve of INRIA Pedestrian Dataset for YOLOv3 Method

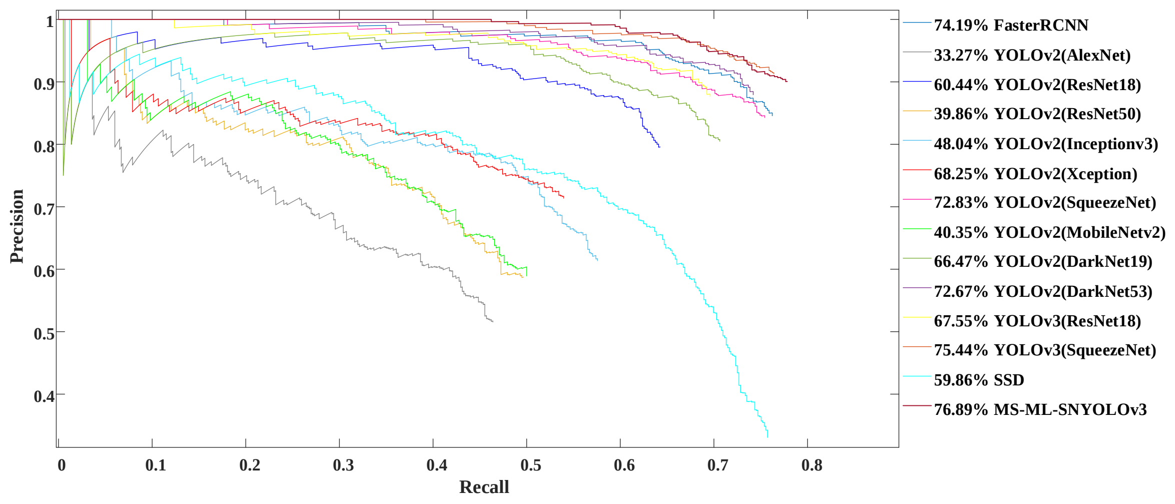


Figure 6.19 P-R Curve of INRIA Pedestrian Dataset for YOLOv3 Method

Table 6.6 LAMR and AP values of INRIA Pedestrian Dataset for YOLOv3 Method

Methods	**INRIA**	
	LAMR	**AP**
FasterRCNN	34.78	74.19
YOLOv2(AlexNet)	77.83	33.27
YOLOv2(ResNet18)	51.14	60.44
YOLOv2(ResNet50)	72.69	39.86
YOLOv2(Inceptionv3)	66.82	48.04
YOLOv2(Xception)	45.25	68.25
YOLOv2(SqueezeNet)	37.63	72.83
YOLOv2(MobileNetv2)	72.11	40.35
YOLOv2(DarkNet19)	44.86	66.47
YOLOv2(DarkNet53)	34.89	72.67
YOLOv3(ResNet18)	41.26	67.55
YOLOv3(SqueezeNet)	30.65	75.44
SSD	62.19	59.86
MS-ML-SNYOLOv3	**28.36**	**76.89**

6.3.2 PASCAL VOC 2012 Dataset

Proposed YOLOv2 Methods

The proposed YOLOv2 methods predicts the bounding boxes along with respective confidence scores. On comparison with the groundtruth values the DET Curve and P-R Curve are obtained. The DET Curve and P-R Curve for the YOLOv2 methods are shown in Figure 6.20 and Figure 6.21 respectively. The corresponding LAMR and AP values are shown in Table 6.7. The proposed DSM-IDM-YOLO has achieved the least miss rate and highest precision with 53.13% and 66.12%, respectively. A minimum improvement of 0.82% and 1.18% is attained by DSM-IDM-YOLO w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.22. The proposed InceptionDepth-wiseYOLOv2 has achieved the least miss rate and highest precision with 52.21% and 67.04%, respectively. A minimum improvement of 1.74% and 2.1% is attained by InceptionDepth-wiseYOLOv2 w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.23. The proposed FireYOLOv2 has achieved the least miss rate and highest precision with 50.70% and 68.64% respectively. LightWeight FireYOLOv2 has the second least miss rate of 53.23% and second highest precision of 65.88%. An improvement of 3.25% and 3.7% is attained by FireYOLOv2 w.r.t. LAMR and AP respectively whereas for LightWeight FireYOLOv2 there is a slight improvement of 0.72% and 0.94% respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.24.

Scale Analysis:

The PASCAL VOC 2012 'person' dataset is also dominated by pedestrians of 'Large' scale, i.e., height > 80. However, pedestrians with 'Medium' scale, i.e., 30 < height < 80, and 'Small' scale, i.e., height < 30 are also present. The scale of the predicted pedestrians is analyzed and the following result is observed. The Scale Analysis is represented in Figure 6.25.

- The DSM-IDM-YOLO detects the greatest number of pedestrians on all scales, 6086 'Large', 230 'Medium' and 2 'Small'.
- The InceptionDepth-wiseYOLOv2 detects the greatest number of pedestrians in all the scales, 6179 'Large' and 286 'Medium' and 4 'Small'.
- The FireYOLOv2 detects the greatest number of pedestrians in all the scales which is 6173 'Large', 293 'Medium' and 3 'Small'.

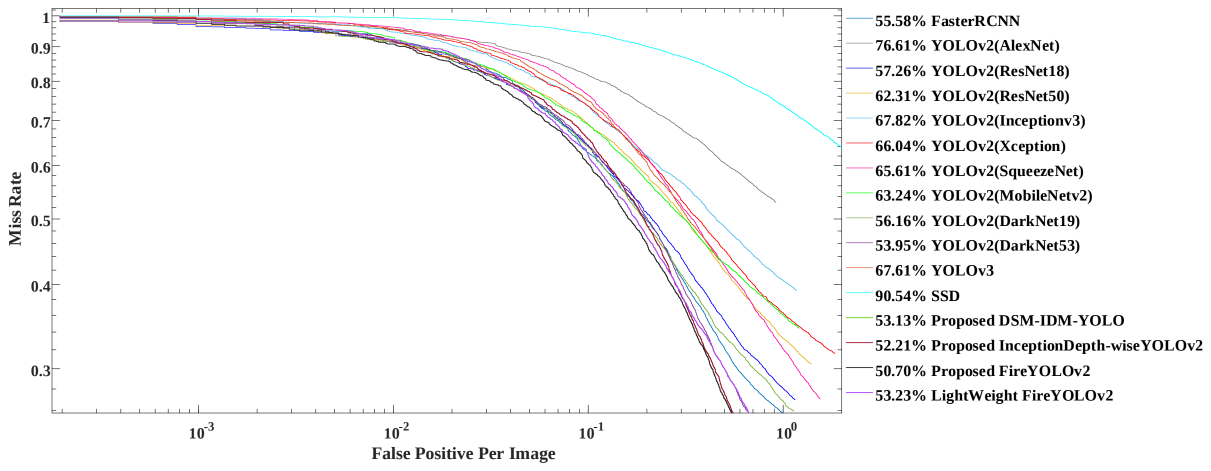


Figure 6.20 DET Curve of PASCAL VOC 2012 Dataset for YOLOv2 Methods

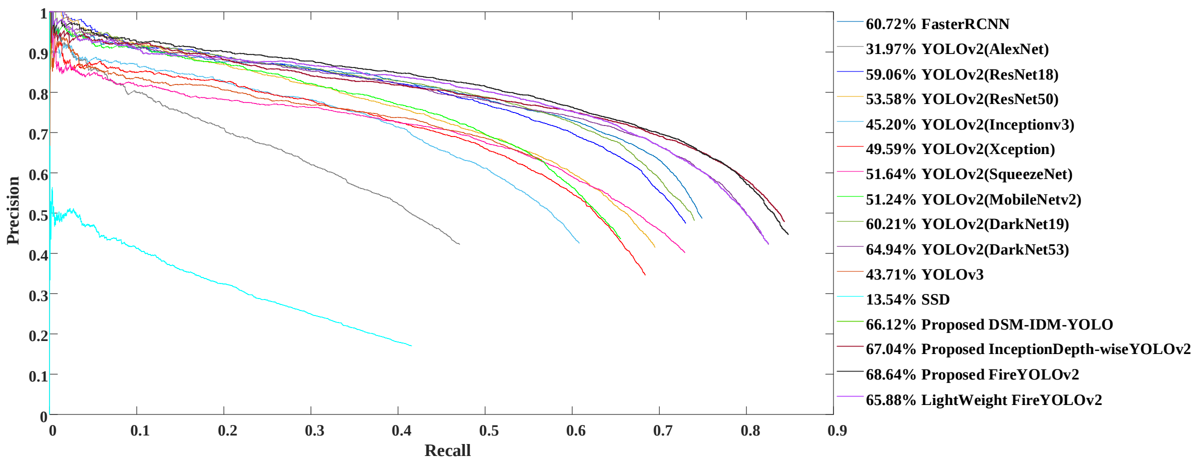


Figure 6.21 P-R Curve of PASCAL VOC 2012 Dataset for YOLOv2 Methods

Table 6.7 LAMR and AP values of PASCAL VOC 2012 Dataset for YOLOv2 Methods

Methods	**PASCAL VOC 2012**	
	LAMR	**AP**
FasterRCNN	55.58	60.72
YOLOv2(AlexNet)	76.61	31.97
YOLOv2(ResNet18)	57.26	59.06
YOLOv2(ResNet50)	62.31	53.58
YOLOv2(Inceptionv3)	67.82	45.20
YOLOv2(Xception)	66.04	49.59
YOLOv2(SqueezeNet)	65.61	51.64
YOLOv2(MobileNetv2)	63.24	51.24
YOLOv2(DarkNet19)	56.16	60.21
YOLOv2(DarkNet53)	53.95	64.94
YOLOv3	67.61	43.71
SSD	90.54	13.54
Proposed DSM-IDM-YOLO	**53.13**	**66.12**
Proposed InceptionDepth-wiseYOLOv2	**52.21**	**67.04**
Proposed FireYOLOv2	**50.70**	**68.64**
LightWeight FireYOLOv2	**53.23**	**65.88**

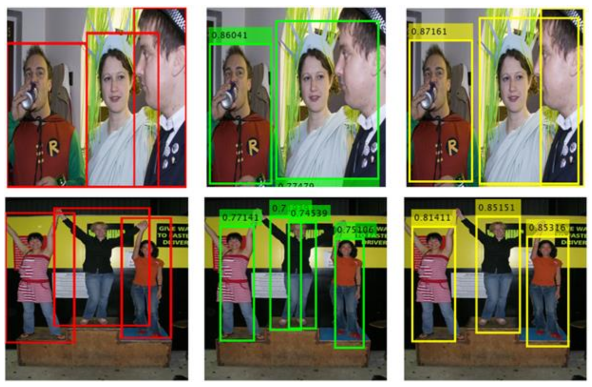


Figure 6.22 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO

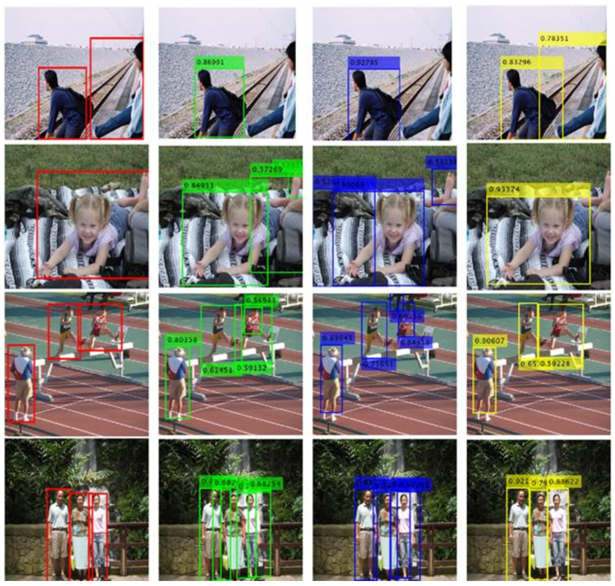


Figure 6.23 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2

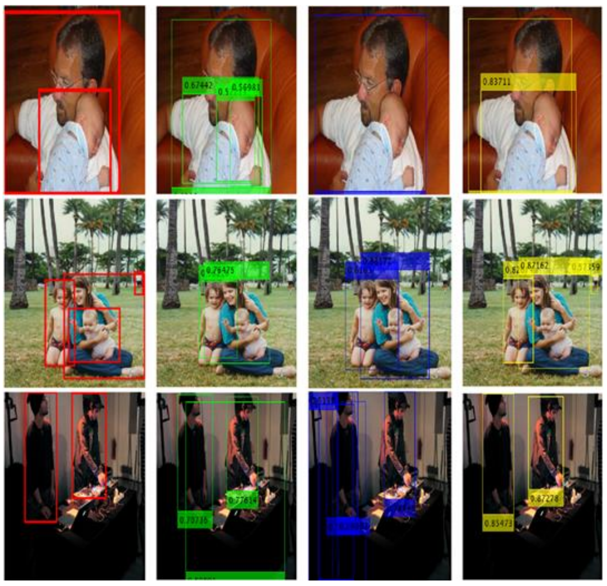


Figure 6.24 PASCAL VOC 2012 Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2

6.3.3 Caltech Pedestrian Dataset

Proposed YOLOv2 Methods

The proposed YOLOv2 methods predicts the bounding boxes along with respective confidence scores. When compared with the groundtruth values the DET Curve and P-R Curve are obtained. The DET Curve and P-R Curve for the YOLOv2 methods are shown in Figure 6.26 and Figure 6.27 respectively. The corresponding LAMR and AP values are shown in Table 6.8. The proposed DSM-IDM-YOLO has achieved the least miss rate and highest precision with 72.36% and 37.20%, respectively. A minimum improvement of 6.3% and 11.13% is attained by DSM-IDM-YOLO w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.28. The proposed InceptionDepth-wiseYOLOv2 has achieved the least miss rate and highest precision with 67.32% and 43.84%, respectively. A minimum improvement of 11.34% and 17.77% is attained by InceptionDepth-wiseYOLOv2 w.r.t. LAMR and AP, respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.29. The proposed FireYOLOv2 has achieved the least miss rate and highest precision with 66.89% and 42.54% respectively. LightWeight FireYOLOv2 has the second least miss rate of 71.16% and second highest precision of 40.29%. An improvement of 11.77% and 16.47% is attained by FireYOLOv2 w.r.t. LAMR and AP respectively whereas for LightWeight FireYOLOv2 there is an improvement of 7.5% and 14.22% respectively. Some sample groundtruth and output images for comparison is shown in Figure 6.30.

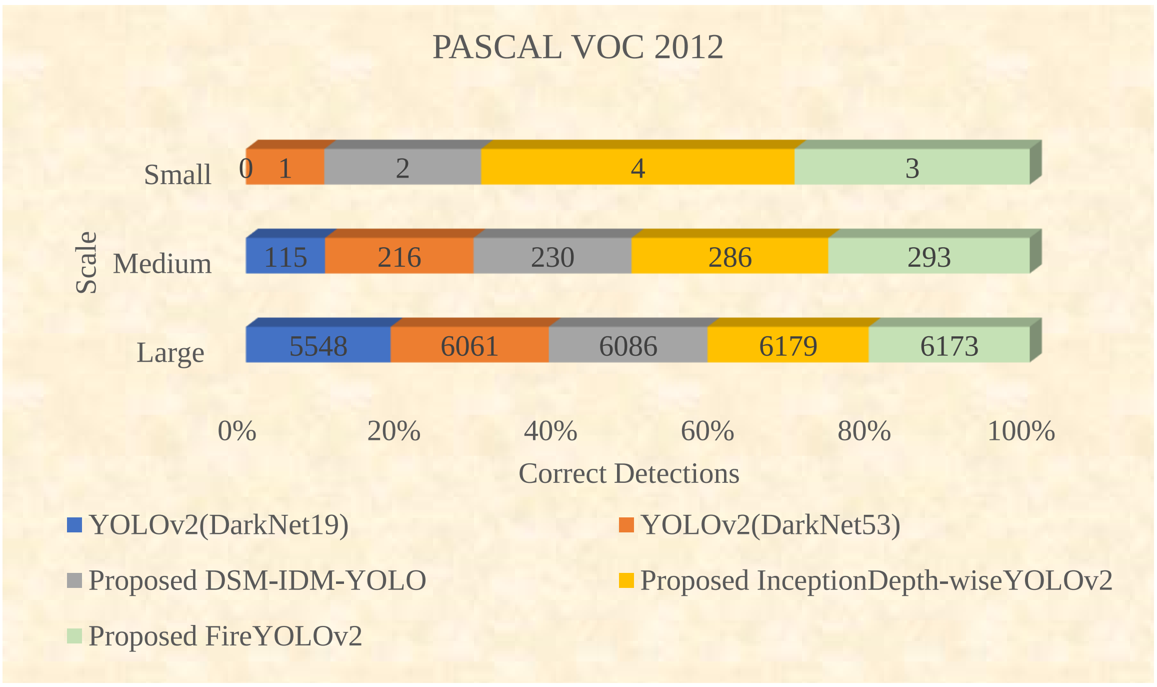


Figure 6.25 Scale Analysis of PASCAL VOC 2012 Dataset for YOLOv2 Methods

Scale Analysis:

The Caltech Pedestrian dataset's Reasonable set is made up of pedestrians of 'Large' scale i.e., height > 80 and 'Medium' scale i.e., 30 < height < 80. Upon examining the scale of the predicted pedestrians, the following points are noted. The Scale Analysis is represented in Figure 6.31.

- The DSM-IDM-YOLO detects the greatest number of pedestrians on all scales, 274 'Large' and 275 'Medium'.
- The InceptionDepth-wiseYOLOv2 detects the greatest number of pedestrians in all the scales, 304 'Large' and 340 'Medium'.

- The FireYOLOv2 detects the greatest number of pedestrians in all the scales which is 268 'Large' and 335 'Medium'.

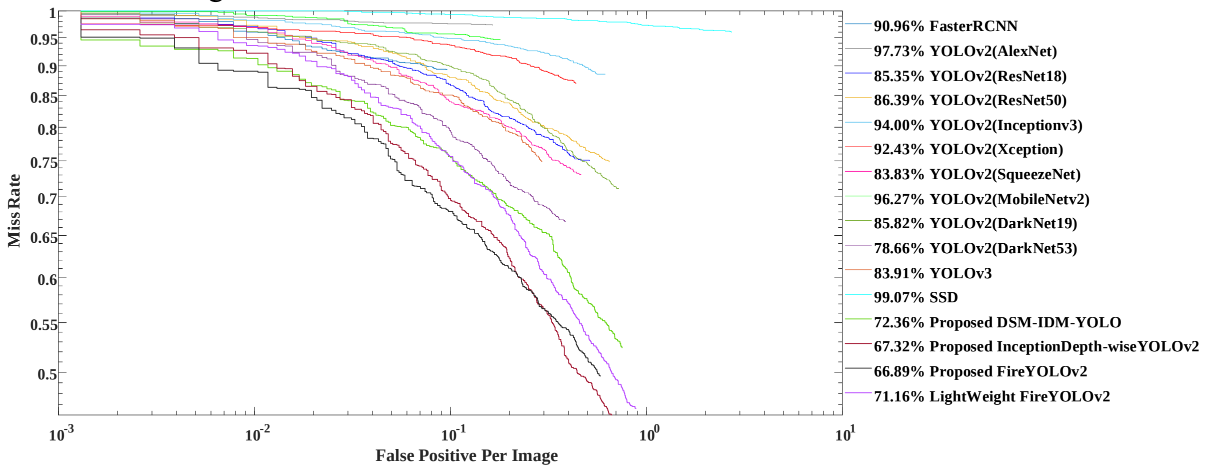


Figure 6.26 DET Curve of Caltech Pedestrian Dataset for YOLOv2 Methods

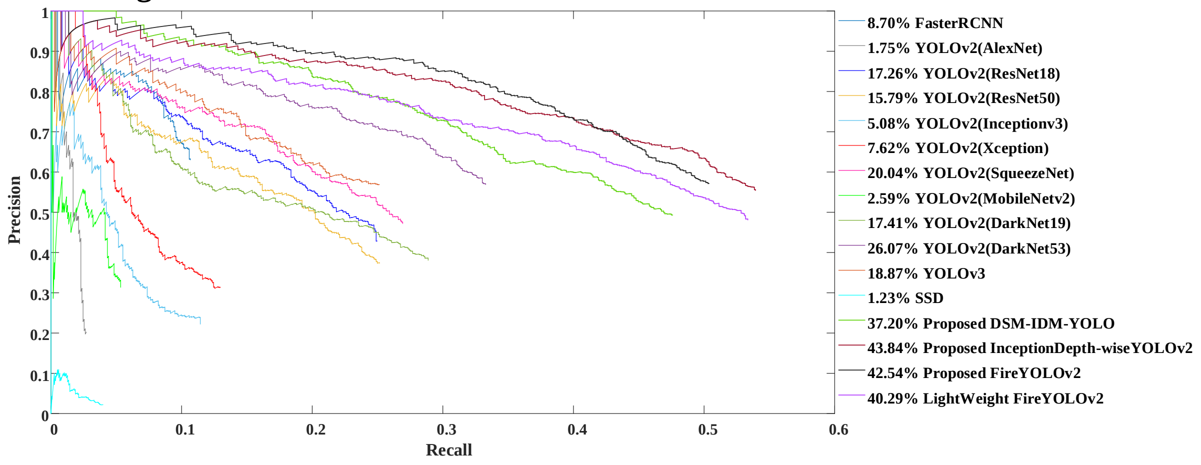


Figure 6.27 P-R Curve of Caltech Pedestrian Dataset for YOLOv2 Methods

Table 6.8 LAMR and AP values of Caltech Pedestrian Dataset for YOLOv2 Methods

Methods	**Caltech**	
	LAMR	**AP**
FasterRCNN	90.96	8.70
YOLOv2(AlexNet)	97.73	1.75
YOLOv2(ResNet18)	85.35	17.26
YOLOv2(ResNet50)	86.39	15.79
YOLOv2(Inceptionv3)	94.00	5.08
YOLOv2(Xception)	92.43	7.62
YOLOv2(SqueezeNet)	83.83	20.04
YOLOv2(MobileNetv2)	96.27	2.59
YOLOv2(DarkNet19)	85.82	17.41
YOLOv2(DarkNet53)	78.66	26.07
YOLOv3	83.91	18.87
SSD	99.07	1.23
Proposed DSM-IDM-YOLO	**72.36**	**37.20**
Proposed InceptionDepth-wiseYOLOv2	**67.32**	**43.84**
Proposed FireYOLOv2	**66.89**	**42.54**
LightWeight FireYOLOv2	**71.16**	**40.29**

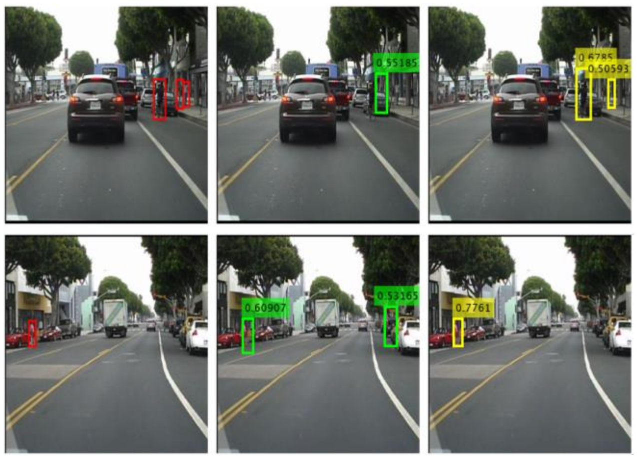


Figure 6.28 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19) and Proposed DSM-IDM-YOLO

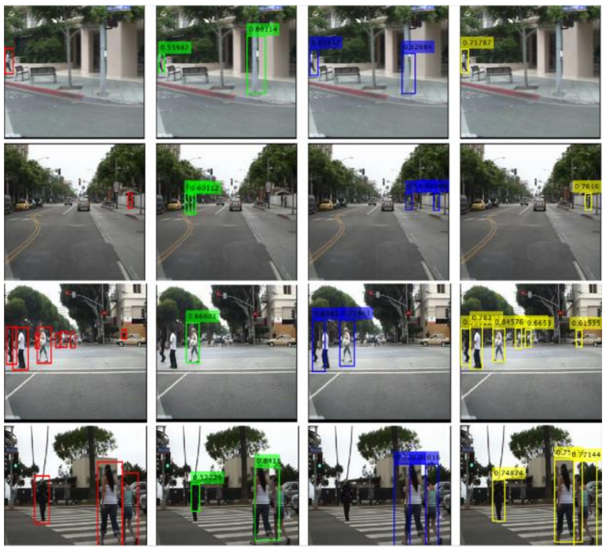


Figure 6.29 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed InceptionDepth-wiseYOLOv2

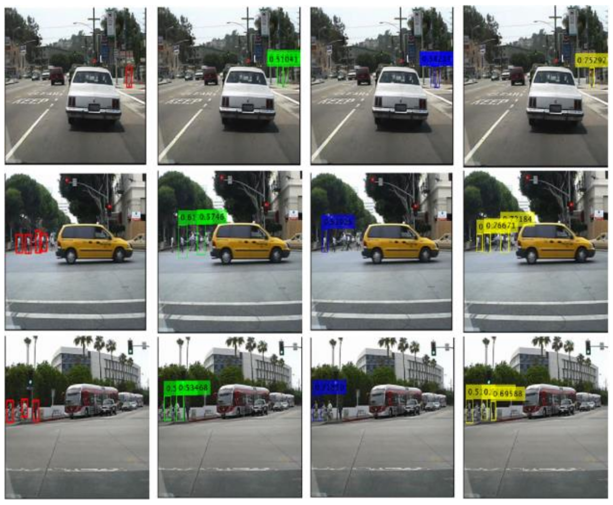


Figure 6.30 Caltech Pedestrian Dataset. Bounding box (L-R) Groundtruth and predicted by YOLOv2(DarkNet19), YOLOv2(DarkNet53) and Proposed FireYOLOv2

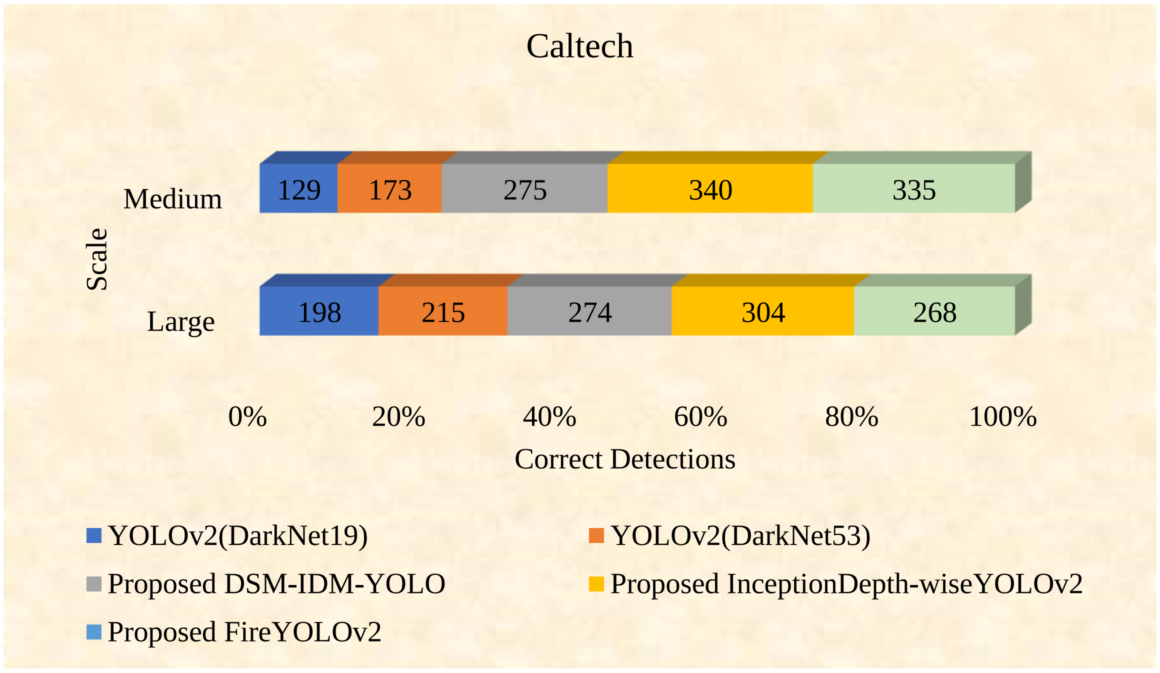


Figure 6.31 Scale Analysis of Caltech Pedestrian Dataset for YOLOv2 Methods

Proposed YOLOv3 Method

The proposed YOLOv3 method predicts the bounding box information and their respective confidence scores. On comparison with the groundtruth values the DET Curve and P-R Curve are obtained. The DET Curve and P-R Curve for the YOLOv3 methods are shown in Figure 6.32 and Figure 6.33 respectively. The corresponding LAMR and AP values are shown in Table 6.9. The proposed MS-ML-SNYOLOv3 obtains a LAMR of 76.26% and AP of 28.45%. It has achieved a minimum performance gain of 1.89% and 2.01% in miss rate and precision respectively.

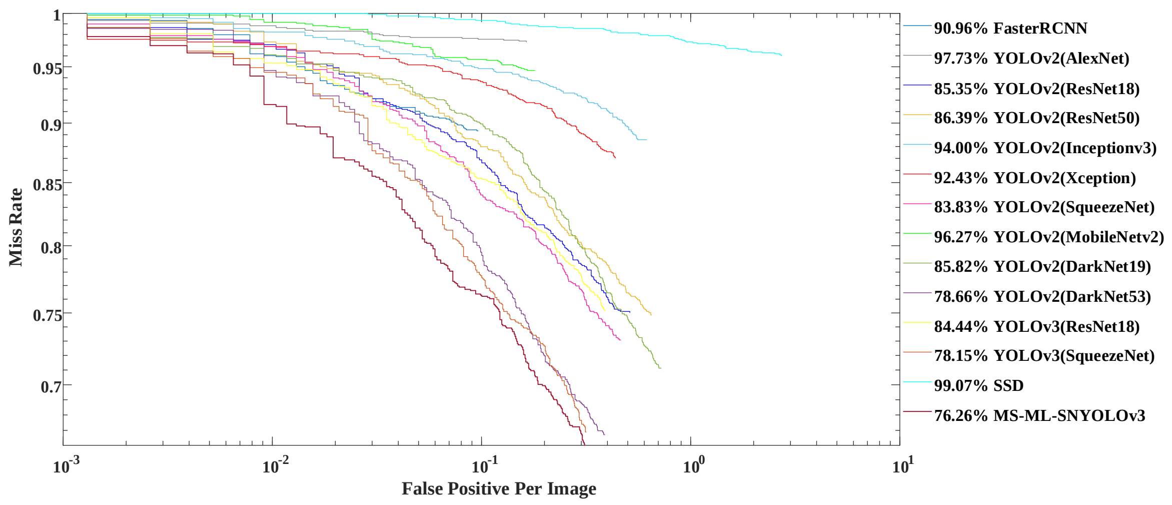


Figure 6.32 DET Curve of Caltech Pedestrian Dataset for YOLOv3 Method

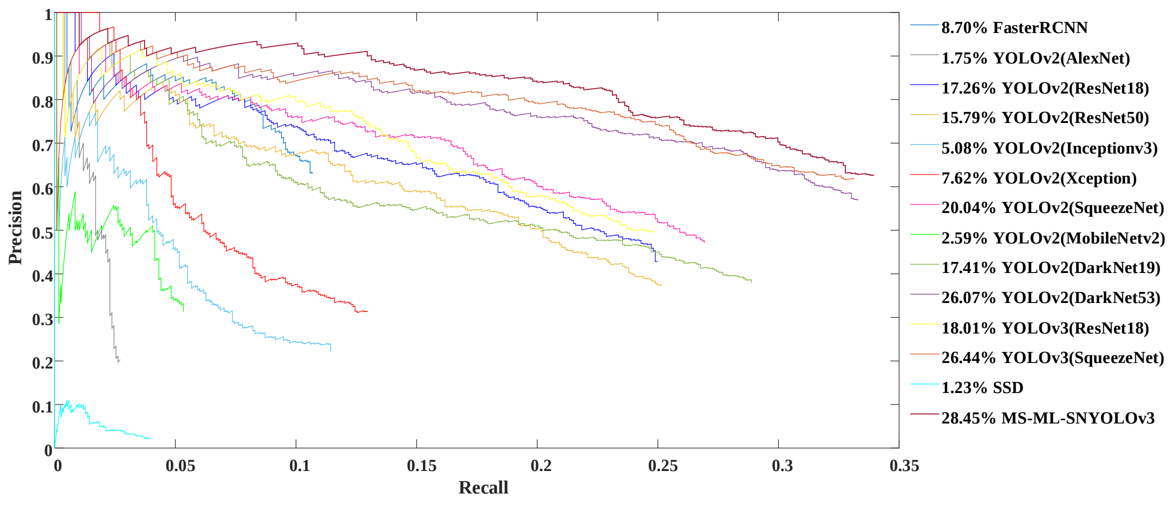


Figure 6.33 P-R Curve of Caltech Pedestrian Dataset for YOLOv3 Method

Table 6.9 LAMR and AP values of Caltech Pedestrian Dataset for YOLOv3 Method

Methods	**Caltech**	
	LAMR	**AP**
FasterRCNN	90.96	8.70
YOLOv2(AlexNet)	97.73	1.75
YOLOv2(ResNet18)	85.35	17.26
YOLOv2(ResNet50)	86.39	15.79
YOLOv2(Inceptionv3)	94.00	5.08
YOLOv2(Xception)	92.43	7.62
YOLOv2(SqueezeNet)	83.83	20.04
YOLOv2(MobileNetv2)	96.27	2.59
YOLOv2(DarkNet19)	85.82	17.41
YOLOv2(DarkNet53)	78.66	26.07
YOLOv3(ResNet18)	84.44	18.01
YOLOv3(SqueezeNet)	78.15	26.44
SSD	99.07	1.23
MS-ML-SNYOLOv3	**76.26**	**28.45**

6.3.4 Statistical Analysis

Proposed YOLOv2 Methods

In the first two proposed YOLOv2 modification, there are 13 independent variables, i.e., the methods (k) and 3 datasets (N). In the third proposed YOLOv2 modification, there are 14 independent variables, i.e., the methods (k) and 3 datasets (N). The proposed YOLOv2 methods DSM-IDM-YOLO and InceptionDepth-wiseYOLOv2 has the first highest rank when compared individually with two-stage FasterRCNN and single-stage YOLOv2(AlexNet), YOLOv2(ResNet18), YOLOv2(ResNet50), YOLOv2(Inceptionv3), YOLOv2(Xception), YOLOv2(SqueezeNet), YOLOv2(MobileNetv2), YOLOv2(DarkNet19), YOLOv2(DarkNet53), YOLOv3 and SSD. The rank tables for DSM-IDM-YOLO and

InceptionDepth-wiseYOLOv2 are shown in Table 6.10 and Table 6.11 respectively. The proposed YOLOv2 methods FireYOLOv2 and LightWeightFireYOLOv2 has the first and second rank when compared with two-stage FasterRCNN and single-stage YOLOv2(AlexNet), YOLOv2(ResNet18), YOLOv2(ResNet50), YOLOv2(Inceptionv3), YOLOv2(Xception), YOLOv2(SqueezeNet), YOLOv2(MobileNetv2), YOLOv2(DarkNet19), YOLOv2(DarkNet53), YOLOv3 and SSD. The rank table for FireYOLOv2 and LightWeightFireYOLOv2 is shown in Table 6.12.

The DSM-IDM-YOLO method:

As per the statistical analysis procedure discussed in Chapter-1, first, the chi-square is calculated to be 27.851. The critical value of chi-square at a degree of freedom (=k-1) 12 is 21.026. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. The calculated F-distribution is 6.835. The critical value of F-distribution with the degree of freedom k-1 and (k-1)(N-1) (12, 24) and α = 0.05 is 2.183. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

Table 6.10 Rank Table of Friedman Test for the Proposed DSM-IDM-YOLO

Methods	**INRIA**		**PASCAL VOC 2012**		**Caltech**		**Mean Rank**	**Rank**
	Miss Rate	***r***	**Miss Rate**	***r***	**Miss Rate**	***r***		
FasterRCNN	34.78	3	55.58	3	90.96	8	4.66	3
YOLOv2(AlexNet)	77.83	13	76.61	12	97.73	12	12.33	10
YOLOv2(ResNet18)	51.14	8	57.26	5	85.35	5	6.00	5
YOLOv2(ResNet50)	72.69	12	62.31	6	86.39	7	8.33	6
YOLOv2(Inceptionv3)	66.82	10	67.82	11	94.00	10	10.33	8
YOLOv2(Xception)	45.25	7	66.04	9	92.43	9	8.33	6
YOLOv2(SqueezeNet)	37.63	5	65.61	8	83.83	3	5.33	4
YOLOv2(MobileNetv2)	72.11	11	63.24	7	96.27	11	9.66	7
YOLOv2(DarkNet19)	44.86	6	56.16	4	85.82	6	5.33	4
YOLOv2(DarkNet53)	34.89	4	53.95	2	78.66	2	2.66	2
YOLOv3	30.65	2	67.61	10	83.91	4	5.33	4
SSD	62.19	9	90.54	13	99.07	13	11.66	9
Proposed DSM-IDM-YOLO	**27.68**	**1**	**53.13**	**1**	**72.36**	**1**	**1.00**	**1**

The InceptionDepth-wiseYOLOv2 method:

Following the procedure of statistical analysis discussed in Chapter-1, the chi-square value is calculated to be 27.851. The critical value of chi-square at a degree of freedom (=k-1) 12 is 21.026. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. The calculated F-distribution is 6.835. The critical value of F-distribution with the degree of freedom k-1 and (k-1)(N-1) (12, 24) and α = 0.05 is 2.183. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

Table 6.11 Rank Table of Friedman Test for the Proposed InceptionDepth-wiseYOLOv2

Methods	**INRIA**		**PASCAL VOC 2012**		**Caltech**		**Mean Rank**	**Rank**
	Miss Rate	***r***	**Miss Rate**	***r***	**Miss Rate**	***r***		
FasterRCNN	34.78	3	55.58	3	90.96	8	4.66	3
YOLOv2(AlexNet)	77.83	13	76.61	12	97.73	12	12.33	10
YOLOv2(ResNet18)	51.14	8	57.26	5	85.35	5	6.00	5
YOLOv2(ResNet50)	72.69	12	62.31	6	86.39	7	8.33	6
YOLOv2(Inceptionv3)	66.82	10	67.82	11	94.00	10	10.33	8
YOLOv2(Xception)	45.25	7	66.04	9	92.43	9	8.33	6
YOLOv2(SqueezeNet)	37.63	5	65.61	8	83.83	3	5.33	4
YOLOv2(MobileNetv2)	72.11	11	63.24	7	96.27	11	9.66	7
YOLOv2(DarkNet19)	44.86	6	56.16	4	85.82	6	5.33	4
YOLOv2(DarkNet53)	34.89	4	53.95	2	78.66	2	2.66	2
YOLOv3	30.65	2	67.61	10	83.91	4	5.33	4
SSD	62.19	9	90.54	13	99.07	13	11.66	9
Proposed InceptionDepth-wiseYOLOv2	**21.92**	**1**	**52.21**	**1**	**67.32**	**1**	**1.00**	**1**

The FireYOLOv2 and LightWeightFireYOLOv2 method:

The steps of statistical analysis, as described in Chapter-1, specifies to calculate the chi-square value, which is resulting to 31.920. The critical value of chi-square at a degree of freedom (=k-1) 13 is 22.362. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected. The calculated F-distribution is 9.017. The critical value of F-distribution with the degree of freedom k-1 and (k-1)(N-1) (13, 26) and α = 0.05 is 1.99. As the calculated F-distribution is greater than the critical F-distribution, the Null Hypothesis is rejected.

Table 6.12 Rank Table of Friedman Test for the Proposed FireYOLOv2 and LightWeightFireYOLOv2

Methods	**INRIA**		**PASCAL VOC 2012**		**Caltech**		**Mean Rank**	**Rank**
	Miss Rate	***r***	**Miss Rate**	***r***	**Miss Rate**	***r***		
FasterRCNN	34.78	4	55.58	4	90.96	9	5.66	3
YOLOv2(AlexNet)	77.83	14	76.61	13	97.73	13	13.33	10
YOLOv2(ResNet18)	51.14	9	57.26	6	85.35	6	7.00	5
YOLOv2(ResNet50)	72.69	13	62.31	7	86.39	8	9.33	6
YOLOv2(Inceptionv3)	66.82	11	67.82	12	94.00	11	11.33	8
YOLOv2(Xception)	45.25	8	66.04	10	92.43	10	9.33	6
YOLOv2(SqueezeNet)	37.63	6	65.61	9	83.83	4	6.33	4
YOLOv2(MobileNetv2)	72.11	12	63.24	8	96.27	12	10.66	7
YOLOv2(DarkNet19)	44.86	7	56.16	5	85.82	7	6.33	4
YOLOv2(DarkNet53)	34.89	5	53.95	3	78.66	3	3.66	2
YOLOv3	30.65	3	67.61	11	83.91	5	6.33	4
SSD	62.19	10	90.54	14	99.07	14	12.66	9
Proposed FireYOLOv2	**19.60**	**1**	**50.70**	**1**	**66.89**	**1**	**1.00**	**1**
LightWeight FireYOLOv2	**26.30**	**2**	**53.23**	**2**	**71.16**	**2**	**2.00**	**2**

Proposed YOLOv3 Method

In this work, there are 14 (k) independent variables, which are the detection methods and there are 2 (N) datasets. The proposed method MS-ML-SNYOLOv3 has the first rank when compared with FasterRCNN, YOLOv2(AlexNet), YOLOv2(ResNet18), YOLOv2(ResNet50), YOLOv2(Inceptionv3), YOLOv2(Xception), YOLOv2(SqueezeNet), YOLOv2(MobileNetv2), YOLOv2(DarkNet19), YOLOv2(DarkNet53), YOLOv3(ResNet18), YOLOv3(SqueezeNet) and SSD. The rank table for MS-ML-SNYOLOv3 is shown in Table

6.13. Following the steps involved in statistical analysis as discussed in Chapter-1, the chi-square is calculated to be 23.314. The critical value of chi-square at a degree of freedom 13 is 22.362. The degree of freedom, here is the number of detection methods(k) subtracted by 1. As the calculated value of chi-square is greater than the critical value of chi-square, the Null Hypothesis is rejected.

Table 6.13 Rank Table of Friedman Test for the Proposed MS-ML-SNYOLOv3

Methods	**INRIA**		**Caltech**		**Mean**	**Rank**
	Miss Rate	***r***	**Miss Rate**	***r***	**Rank**	
FasterRCNN	34.78	3	90.96	**9**	6	6
YOLOv2(AlexNet)	77.83	14	97.73	13	13.5	13
YOLOv2(ResNet18)	51.14	9	85.35	6	7.5	8
YOLOv2(ResNet50)	72.69	13	86.39	8	10.5	10
YOLOv2(Inceptionv3)	66.82	11	94.00	11	11	11
YOLOv2(Xception)	45.25	8	92.43	10	9	9
YOLOv2(SqueezeNet)	37.63	5	83.83	4	4.5	4
YOLOv2(MobileNetv2)	72.11	12	96.27	12	12	12
YOLOv2(DarkNet19)	44.86	7	85.82	7	7	7
YOLOv2(DarkNet53)	34.89	4	78.66	3	3.5	3
YOLOv3 (ResNet18)	41.26	6	84.44	5	5.5	5
YOLOv3 (SqueezeNet)	30.65	2	78.15	2	2	2
SSD	62.19	10	99.07	14	12	12
MS-ML-SNYOLOv3	**28.36**	**1**	**76.26**	**1**	**1**	**1**

6.4 Observations

In this contribution, as per the proposed YOLOv2 method: DSM-IDM-YOLO with base DarkNet19, InceptionDepth-wiseYOLOv2 with base DarkNet53 & FireYOLOv2 with base DarkNet53 and the proposed YOLOv3 method: MS-ML-SNYOLOv3 with base SqueezeNet, the following points are observed. Three benchmark pedestrian datasets are used to evaluate YOLOv2 proposed methods INRIA, PASCAL VOC 2012, and Caltech Pedestrian datasets. Two benchmark pedestrian datasets are used to evaluate YOLOv3 proposed method INRIA and Caltech Pedestrian datasets.

- The fused features from the proposed modules at different levels of the network amassed rich hierarchical feature information of the objects, i.e., pedestrians in the images.
- The proposed YOLOv2 methods have achieved the best performance i.e., least miss rate and highest precision in all the three datasets.
- The proposed YOLOv2 methods are performing better than the advanced YOLOv3 network also.
- The proposed YOLOv3 method have achieved the best performance i.e., least miss rate and highest precision in all the two datasets.

Chapter 7

Conclusion and Future Scope

The conclusion and future scope of the thesis is discussed in this chapter.

Conclusion

The pedestrian detection problem is addressed in this work by focusing on the feature development part of the process. To extract a detailed and dense feature representation, four detection methods are explored.

For Contribution-1, the hand-crafted features are employed. A scale-space pyramid-based shape feature-extraction method: SI-HOG is proposed. SI-HOG overcomes the shortcoming of HOG, i.e., that it is not applicable to multiresolution images, by considering gradient information from different scales of an image, making it resolution-independent. The addition of texture and color information to SI-HOG enables a more enhanced form of features. The performance of the proposed method is evaluated using three datasets, i.e., INRIA, NICTA, and Daimler, considering both single-resolution and multiresolution images wherein it has shown the least miss rate.

The features extracted by hand-crafted feature concentrate only one on particular aspect or feature of the image. This causes a significance performance gap. Whereas the features formed by deep CNN methods provides a comprehensive representation of the image.

For Contribution-2, the CNN features are used. A modified architecture for ResNet18 is proposed. The proposed method processes features from varying levels of the network. It is named MF2ResNet and is used in two ways: CNN features with SVM and End-to-End CNN network. The proposed methods are compared using three benchmark pedestrian datasets INRIA, NICTA and Daimler wherein it surpasses the handcrafted features performance.

In the first two contribution, the pedestrian detection is being done in per-window evaluation system. To further address the pedestrian detection problem, the state-of-the-art detection method is employed in the next contributions. The detection methods follow the per-image paradigm and generates the bounding box as well as the confidence score of the pedestrians.

For Contribution-3, the two stage Faster RCNN is used. Two modifications of CNN network ResNet18, which serves as a base for Faster RCNN is proposed. The proposed methods are named Faster RCNN (DCResNet) and Faster RCNN (MF2ResNet). The feature map is formed by the concatenation of the processed output feature map of the proposed networks. This enables a thorough feature extraction process as the resultant feature map gives a detailed hierarchical representation of the image. The proposed method is evaluated on two benchmark datasets INRIA and PASCAL VOC 2012 and has shown substantial improvement.

This work provided a reasonably better solution for pedestrian detection with best miss rate and precision. But the Faster RCNN methods, being the two-stage method lack speed. The next contribution addressed this with YOLOv2 and YOLOv3 detection methods, which proved to be faster and more accurate.

For Contribution-4, the single stage YOLO network is used. Three YOLOv2 modifications are proposed: DSM-IDM-YOLO with base DarkNet19, InceptionDepth-wiseYOLOv2 with base DarkNet53, and FireYOLOv2 with base DarkNet53. The proposed methods are evaluated on INRIA, PASCAL VOC 2012, and Caltech Pedestrian datasets, wherein it gives improved miss rate and precision.

For Contribution-5, a YOLOv3 modification is proposed with base SqueezeNet named as MS-ML-SNYOLOv3. It also achieves improved miss rate and precision when evaluated with INRIA and PASCAL VOC 2012 pedestrian dataset.

To summarize the work, the pedestrian detection problem is tackled with incremental improvement starting from hand-crafted features to deep learning CNN features. The two-stage and single-stage detection methods of Faster RCNN and YOLO respectively, provides sophisticated and advanced model to further improve the accuracy of the system. It can be established that the proposed method in single stage network YOLO has given the best performance in the work in terms of miss rate and precision and with lesser computational overhead.

Future Scope

- The pedestrian detection is limited to images in this work. It will be extended to videos so that it can be utilized in real world applications.
- The improvement in this work is done upon the feature extraction part. But to further improve the localization of the pedestrians, the region proposal part will be enhanced.
- Latest state-of-the-art detection methods will be employed wherein the base CNN architecture can be improved.
- As the detection method involves computational overhead, it will be shifted to a distributed environment to speed up the process.
- On the other hand, light weight detection methods will be proposed, to enable real time application without the requirement of a GPU.
- To aid in the detection of various scales of pedestrians in the image, methods such as super-resolution will be analyzed.

List of Publications

1. **Sweta Panigrahi** and U.S.N. Raju, "Scale-invariant Histogram of Oriented Gradients: Novel Approach for Pedestrian Detection in Multiresolution Image Dataset", Turkish Journal of Electrical Engineering Computer Sciences, Vol. 29, No. 7, pp. 3053-3073. (Journal: Accepted-2021). doi: 10.3906/elk-2009-27
2. **Sweta Panigrahi** and U.S.N. Raju, "Pedestrian Detection Based on Hand-crafted Features and Multi-layer Feature Fused-ResNet Model", International Journal on Artificial Intelligence Tools, Vol. 30, No. 05. (Journal: Accepted-2021). doi: 10.1142/S0218213021500287
3. **Sweta Panigrahi** and U.S.N. Raju, "An improved Faster RCNN for Pedestrian Detection", International Conference on Control, Automation, Power and Signal Processing (CAPS-2021), Organized by IIITDM, Jabalpur. (Conference: Accepted-2021; Date of Presentation: December 10-12, 2021).
4. **Sweta Panigrahi**, U.S.N. Raju, R.P. Raj, S. Namulamettu and V. Thanda, "Pedestrian Detection: Unification of Global and Local Features", International Conference on Innovations in Computational Intelligence and Computer Vision: Proceedings of ICICV, Vol. 1189, pp. 437-446. (Conference: Presented-January 17-19, 2020). doi: 10.1007/978-981-15-6067-5_49.
5. **Sweta Panigrahi** and U.S.N. Raju, "InceptionDepth-wiseYOLOv2: Improved implementation of YOLO Framework for Pedestrian Detection", International Journal of Multimedia Information Retrieval. (Journal: Under Review).
6. **Sweta Panigrahi** and U.S.N. Raju, "DSM-IDM-YOLO: Depth-wise Separable Module and Inception Depth-wise Module based YOLO for pedestrian detection", Journal of Electronic Imaging. (Journal: Under Review).
7. **Sweta Panigrahi** and U.S.N. Raju, "FireYOLOv2: New and Improved Framework for Pedestrian Detection", Neural Processing Letters. (Journal: Under Review).
8. **Sweta Panigrahi** and U.S.N. Raju, "MS-ML-SNYOLOv3: A Robust Modification of SqueezeNet based YOLOv3 for Pedestrian Detection", Optik. (Journal: Submitted).
9. U.S.N. Raju, Suresh Kumar Kanaparthi, Mahesh Kumar Murampudi, **Sweta Panigrahi**, Debanjan Pathak. "Big Image Data Processing: Methods, Technologies and Implementation Issues", Research Innovations and Trends on Computer Vision and Recognition System, Apple Academic press, CRC Press, a Taylor & Francis Group. (Book Chapter: In Press-2021).

References

1. N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, United States, June (2005), pp. 886–893.
2. A. Satpathy, X. Jiang and H.-L. Eng, Human detection by quadratic classification on subspace of extended histogram of gradients, IEEE Transactions on Image Processing 23(1) (2013), pp. 287-297.
3. Y. LeCun, Y. Bengio and G. Hinton, Deep learning, Nature 521(2015), pp. 436-444.
4. M. Kümmerer, L. Theis and M. Bethge, Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet, arXiv preprint arXiv:1411.1045 (2014).
5. G. Overett, L. Petersson, N. Brewer, et al., A new pedestrian dataset for supervised learning, in: 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, Netherlands, June (2008), pp. 373-378.
6. S. Munder and D.M. Gavrila, An Experimental Study on Pedestrian Classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 28(11) (2006), pp. 1863-1868.
7. M. Everingham, L. Van Gool, C.K. Williams, et al., The pascal visual object classes (voc) challenge, International Journal of Computer Vision 88(2) (2010), pp. 303-338.
8. P. Dollár, C. Wojek, B. Schiele B, et al., Pedestrian detection: A benchmark, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Florida, United States, June (2009), pp. 304-311.
9. G. Afzal, B. Roger, S. Will, et al., Performance metrics for evaluating object and human detection and tracking systems, National Institute of Standards and Technology (NIST) Interagency/Internal Report (NISTIR) - 7972, Maryland, United States, (2014).
10. T. Fawcett, An introduction to ROC analysis, Pattern recognition letters 27 (2006), pp. 861-874.
11. A. Martin, G. Doddington, T. Kamm, et al., The DET Curve in Assessment of Detection Task Performance, National Institute of Standards and Technology (NIST), Maryland, United States, (1997).

12. P. Dollar, C. Wojek, B. Schiele, et al., Pedestrian detection: An evaluation of the state of the art, IEEE Transactions on Pattern Analysis and Machine Intelligence 34(4) (2011), pp. 743-761.
13. D.M. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, Journal of Machine Learning Technologies, 2 (2011), pp. 37-63.
14. M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32(200) (1937), pp. 675-701.
15. R.L. Iman and J.M. Davenport, Approximations of the critical region of the fbietkan statistic, Communications in Statistics-Theory and Methods 9(6) (1980), pp. 571-595.
16. Q. Zhu, S. Avidan, M.-C. Yeh, et al., Fast human detection using a cascade of histograms of oriented gradients, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, United States, June (2006), pp. 1491-1498.
17. D. Wei, Y. Zhao, R. Cheng, et al., An enhanced histogram of oriented gradient for pedestrian detection, in: 2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP), Beijing, China, July (2013), pp. 459-463.
18. J. Huang, S.R. Kumar, M. Mitra, et al., Image indexing using color correlograms, in: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Puerto Rico, United States, June (1997), pp. 762-768.
19. A. K. Bhunia, A. Bhattacharyya, P. Banerjee, et al., A novel feature descriptor for image retrieval by combining modified color histogram and diagonally symmetric co-occurrence texture pattern, Pattern Analysis and Applications 23 (2020), pp. 703-723.
20. Y.D. Chun, N.C. Kim and I.H. Jang, Content based image retrieval using multiresolution color and texture features, IEEE Transactions on Multimedia 10(6) (2008), pp. 1073-1084.
21. T. Ojala, M. Pietikainenet and T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis & Machine Intelligence 24(7) (2002), pp. 971-987.
22. Z.Q. Zhao, P. Zheng, S.T. Xu, et al., Object detection with deep learning: A review, IEEE transactions on neural networks and learning systems 30 (2019), pp. 3212-3232.

23. M. Oquab, L. Bottou, I. Laptev, et al., Is object localization for free? - Weakly-supervised learning with convolutional neural networks, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Massachusetts, Unites States, June (2015), pp. 685-694.
24. M. Oquab, L. Bottou, I. Laptev, et al. Learning and transferring mid-level image representations using convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, Ohio, June (2014), pp.1717-1724.
25. J. Deng, W. Dong, R. Socher, et al., ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Florida, United States, June (2009), pp. 248-255.
26. A. Krizhevsky, I. Sutskever and E.H. Geoffrey, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, Nevada, United States, December (2012), pp. 1097-1105.
27. K. He, X. Zhang, S. Ren, et al., Deep Residual Learning for Image Recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Nevada, United States, June (2016), pp. 770-778.
28. C. Szegedy, V. Vanhoucke, S. Ioffe, et al., Rethinking the inception architecture for computer vision, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Nevada, United States, June (2016), pp. 2818-2826.
29. F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, Hawaii, United States, July (2017), pp. 1800-1807.
30. F.N. Iandola, S. Han, M.W. Moskewicz, et al., SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size, arXiv preprint arXiv:1602.07360 (2016).
31. M. Sandler, A. Howard, M. Zhu, et al., MobileNetV2: Inverted residuals and linear bottlenecks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Utah, United States, June (2018), pp. 4510-4520.
32. J. Redmon and A. Farhadi, YOLO9000: Better, faster, stronger, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Hawaii, United States, July (2017), pp. 6517-6525.

33. J. Redmon and A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767 (2018).
34. V.N. Vapnik, An overview of statistical learning theory, IEEE Transactions on Neural Networks 10(5) (1999), pp. 988-999.
35. A. Mohan, C. Papageorgiou and T. Poggio, Example-based object detection in images by components, IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(4) (2001), pp. 349-361.
36. S. Maji, A.C. Berg and J. Malik, Efficient classification for additive kernel SVMs, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(1) (2013), pp. 66-77.
37. R. Girshick, J. Donahue, T. Darrell, et al., Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, Ohio, June (2014), pp. 580–587.
38. J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, et al., Selective search for object recognition, International Journal of Computer Vision 104 (2013), pp. 154–171.
39. R. Girshick, Fast R-CNN, in: 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, December (2015), pp. 1440–1448.
40. S. Ren, K. He, R. Girshick, et al., Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (2017), pp. 1137-1149.
41. J. Redmon, S. Divvala, R. Girshick, et al., You only look once: unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Nevada, United States, June (2016), pp. 779-788.
42. M.M. Badža and M.C. Barjaktarović, Classification of brain tumors from MRI images using a convolutional neural network, Applied Sciences 10(6) (2020).
43. J. Ker, L. Wang, J. Rao and T. Lim, Deep learning applications in medical image analysis, IEEE Access 6 (2018), pp. 9375-9389.
44. S.S. Yadav and S.M. Jadhav, Deep convolutional neural network based medical image classification for disease diagnosis, Journal of Big Data 6(113) (2019).
45. Y. Liu, L.W.L. Yip, Y. Zheng and L. Wang, Glaucoma screening using an attention-guided stereo ensemble network, Methods, 2021.

46. S.P. Singh, L. Wang, S. Gupta, B. Gulyás and P. Padmanabhan, Shallow 3D CNN for detecting acute brain hemorrhage from medical imaging sensors, IEEE Sensors Journal 21(13) (2021), pp. 14290-14299.
47. J. Ker, S.P. Singh, Y. Bai, J. Rao, T. Lim and L. Wang, Image thresholding improves 3-dimensional convolutional neural network diagnosis of different acute brain hemorrhages on computed tomography scans, Sensors 19(9) (2019), pp. 2167.
48. S.P. Singh, L. Wang, S. Gupta, H. Goli, P. Padmanabhan and B. Gulyás, 3D deep learning on medical images: a review, Sensors 20 (2020), pp. 5097.
49. J. Ker, Y. Bai, H.Y. Lee, J. Rao and L. Wang, Automated brain histology classification using machine learning, Journal of Clinical Neuroscience 66 (2019), pp. 239-245.
50. A. Dhillon and G.K. Verma, Convolutional neural network: a review of models, methodologies and applications to object detection, Progress in Artificial Intelligence 9 (2020), pp. 85–112.
51. R. Li, L. Wang and O. Sourina, Subject matching for cross-subject EEG-based recognition of driver states related to situation awareness, Methods (2021).
52. M. Bilal, A. Khan, M.U.K. Khan, et al., A low complexity pedestrian detection framework for smart video surveillance systems, IEEE Transactions on Circuits and Systems for Video Technology 27 (2016), pp. 2260-2273.
53. J. Kim, J. Park and Y. Do, Pedestrian detection by fusing multiple techniques of a single color camera for a mobile robot, in: 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, Japan, April (2014), pp. 1253-1256.
54. E. Ohn-Bar and M. M. Trivedi, Looking at humans in the age of selfdriving and highly automated vehicles, IEEE Transactions on Intelligent Vehicles 1 (2016), pp. 90-104.
55. S. Seer, N. Brndle and C. Ratti, Kinects and human kinetics: A new approach for studying pedestrian behavior, Transportation Research Part C 48 (2014), pp. 212-228.
56. A. Ess, B. Leibe and L. Van Gool, Depth and appearance for mobile scene analysis, in: 2007 IEEE 11th International Conference on Computer Vision (CVPR), Rio de Janeiro, Brazil, October (2007), pp. 1-8.
57. W. Ouyang and X. Wang, Joint deep learning for pedestrian detection, in: 2013 IEEE International Conference on Computer Vision (CVPR), New South Wales, Australia, December (2013), pp. 2056-2063.

58. H. Li, Z. Wu and J. Zhang, Pedestrian detection based on deep learning model, in: 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Datong, China, October (2016), pp. 796-800.
59. K. He, G. Gkioxari, P. Dollár, et al., Mask R-CNN, in: 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, October (2017), pp. 2980-2988.
60. D.M. Gavrila, A bayesian, exemplar-based approach to hierarchical shape matching, IEEE Transactions on Pattern Analysis & Machine Intelligence 29(8) (2007), pp. 1408 – 1421.
61. C.P. Papageorgiou, A trainable system for object detection in images and video sequences, Ph.D. thesis, Massachusetts Institute of Technology, 1991.
62. O. Tuzel, F. Porikli and P. Meer, Pedestrian detection via classification on riemannian manifolds, IEEE Transactions on Pattern Analysis and Machine Intelligence 30(10) (2008), pp. 1713-1727.
63. B. Leibe, A. Leonardis and B. Schiele, Robust object detection with interleaved categorization and segmentation, International Journal of Computer Vision 77 (2008), pp. 259-289
64. D.G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60(2) (2004), pp. 91-110.
65. S. Nigam and A. Khare, Multiresolution approach for multiple human detection using moments and local binary patterns, Multimedia Tools and Applications 74(17) (2015), pp. 7037-7062.
66. J. Yan, X. Zhang, Z. Lei, et al., Robust multi-resolution pedestrian detection in traffic scenes, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, United States, June (2013), pp. 3033-3040.
67. P.F. Felzenszwalb, R.B. Girshick, D. McAllester, et al., Object detection with discriminatively trained part-based models, IEEE Transactions on Pattern Analysis and Machine Intelligence 32(9) (2009), pp. 1627-1645.
68. P. Hurney, P. Waldron, F. Morgan, et al., Night-time pedestrian classification with histograms of oriented gradients-local binary patterns vectors, IET Intelligent Transport Systems 9(1) (2014), pp. 75-85.

69. M. Bilal, M.S. Hanif, High performance real-time pedestrian detection using light weight features and fast cascaded kernel SVM classification, Journal of Signal Processing Systems 91(2) (2019), pp. 117-129.
70. R. Lahmyed, M. El Ansari, A. Ellahyani, A new thermal infrared and visible spectrum images-based pedestrian detection system, Multimedia Tools and Applications 78(12) (2019), pp. 15861-15885.
71. B.T. Bastian, C.V. Jiji, Integrated feature set using aggregate channel features and histogram of sparse codes for human detection, Multimedia Tools and Applications 79(3) (2020), pp. 2931-2944.
72. K. Kumar, R.K. Mishra, A heuristic SVM based pedestrian detection approach employing shape and texture descriptors, Multimedia Tools and Applications 79 (2020), pp. 21389-21408.
73. X. Zhanga, H. Shangguana, A. Ninga, et al., Pedestrian detection with edge features of color image and hog on depth images, Automatic Control and Computer Sciences 54(2) (2020), pp. 168-178.
74. Y. Xiang, W. Choi, Y. Lin, et al., Subcategory-aware convolutional neural networks for object proposals and detection, in: 2017 IEEE Winter Conference on Applications of Computer Vision, California, United States, March (2017), pp. 924-933.
75. Z.Q. Zhao, H. Bian, D. Hu, et al., Pedestrian detection based on fast r-cnn and batch normalization, in: Intelligent Computing Theories and Application, ICIC 2017, Lecture Notes in Computer Science 10361, (eds.) D.S. Huang, V. Bevilacqua, P. Premaratne, et al., Springer, Cham, 2017, pp. 735-746.
76. Z. Cai, M. Saberian and N. Vasconcelos, Learning complexity-aware cascades for deep pedestrian detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 42(9) (2020), pp. 2195-2211.
77. P. Sermanet, K. Kavukcuoglu, S. Chintala, et al., Pedestrian detection with unsupervised multi-stage feature learning, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Oregon, United States, June (2013), pp. 3626-3633.
78. C. Szegedy, W. Liu, Y. Jia, et al., Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Massachusetts, United States, June (2015), pp. 1-9.

79. D. Tomè, F. Monti, L. Baroffio, et al., Deep convolutional neural networks for pedestrian detection, Signal Processing: Image Communication 47 (2016), pp. 482-489.
80. W. Ouyang, X. Zeng and X. Wang, Partial occlusion handling in pedestrian detection with a deep model, IEEE Transactions on Circuits and Systems for Video Technology 26 (2015), pp. 2123-2137.
81. Q. Hu, P. Wang, C. Shen, et al., Pushing the limits of deep cnns for pedestrian detection, IEEE Transactions on Circuits and Systems for Video Technology 28 (2017), pp. 1358-1368.
82. L. Chunze, J. Lu and J. Zhou, Multi-grained deep feature learning for robust pedestrian detection, IEEE Transactions on Circuits and Systems for Video Technology 29 (2018), pp. 3608-3621.
83. L. Jianan, X. Liang, S. Shen, et al., Scale-aware fast R-CNN for pedestrian detection, IEEE Transactions on Multimedia 20 (2017), pp. 985-996.
84. L. Chunze, J. Lu, G. Wang, et al., Graininess-aware deep feature learning for robust pedestrian detection, IEEE Transactions on image processing 29 (2020), pp. 3820-3834.
85. J. Hosang, M. Omran, R. Benenson, et al., Taking a deeper look at pedestrians, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Massachusetts, United States, June (2015), pp. 4073-4082.
86. Y. Tian, P. Luo, X. Wang, et al., Pedestrian detection aided by deep learning semantic tasks, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Massachusetts, United States, June (2015), pp. 5079-5087.
87. W. Liu, D. Anguelov, D. Erhan, et al., Ssd: Single shot multibox detector, in: Computer Vision – ECCV 2016, Lecture Notes in Computer Science, (eds.) B. Leibe, J. Matas, N. Sebe, M. Welling, Springer, Cham, 2016, pp. 21-37.
88. Z. Yi, S. Yongliang and Z. Jun, An improved tiny-yolov3 pedestrian detection algorithm, Optik 183 (2019), pp. 17-23.
89. W. Lan, J. Dang, Y. Wang, et al., Pedestrian detection based on YOLO network model, in: 2018 IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, August (2018), pp. 1547-1551.
90. Z. Liu, Z. Chen, Z. Li, et al., An efficient pedestrian detection method based on YOLOv2, Mathematical Problems in Engineering 2018 (2018).

91. W.Y. Hsu and W.Y. Lin, Ratio-and-scale-aware YOLO for pedestrian detection, IEEE Transactions on Image Processing 30 (2020), pp. 934-947.
92. X. Yang, Y. Wang and R. Laganière, A scale-aware YOLO model for pedestrian detection, in: Advances in Visual Computing, ISVC 2020, Lecture Notes in Computer Science 12510, (eds.) G. Bebis, Z. Yin, E. Kim, et al., Springer, Cham, 2020, pp. 15-26.